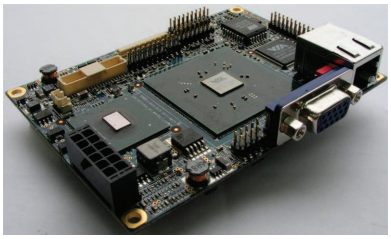
Hírek

Via: már pico-ITX



A *Via* bemutatta népszerű *Epia* sorozatának legújabb tagját, az *Epia PX*-et. 10-szer 7 centiméteres mérete ellenére 1 Ghz-es *C7*-es processzort és számos audió/videó csatlakozót kapott, igaz nagyrészük a megfelelő szalagkábel csatlakoztatása után érhető csak el. A többi *Via* eszközhöz hasonlóan ez is passzív hűtést kapott. A gyártó kétféle chippel szállítja: *CX700M(2)*, illetve *VX700*. A *CX700M*-en találunk videó bemenetet is, míg a *VX700*-al szerelt lapon ehelyett több *USB* bemenetet találunk. Az *Epia PX* ára még nem derült ki, de várhatóan 350 dollár körüli áron kerül majd a boltokba.

➲ http://www.linuxdevices.com/news/NS2154184680.html

PCIExpress kicsiben

Az *Evalue Technology* bemutatta *Via* alapú számítógépét, amely két okból kiemelkedő: 1 GHz-es processzor mellett passzív hűtéssel bír, illetve két darab gigabites hálózati csatolót (*Realtek RTL 8111B*) kapott, melyek *PCI Express* felületen kapcsolódnak a rendszerbe. További érdekesség, hogy akár két – egymástól független – megjelenítő is használható a *LVDS* interfésznek köszönhetően. A rendszer jumper nélkül állítható (*RS232* / *422* / *485*) soros porttal bír.

➲ http://www.linuxdevices.com/news/NS2196733155.html

Ipari POS

Az *Indesign* bemutatta ipari *POS (point-of-sales)* terminálján prototípusát, mely 8 és 32 bites *Freescale ColdFire* processzorokat tartalmaz. *uClinux*ot futtat más nyílt forrású szoftverekkel egyetemben, a cél pedig egy biztonságos terminál megépítése volt.
Az eszköz számbillentyűzettel és mágneskártya-olvasóval is rendelkezik, míg a felhasználót *LCD* kijelzőn keresztül vagy hanggal tájékoztatja. Az adatok titkosítását a hálózati csatoló hardveresen végzi.

➲ http://www.linuxdevices.com/news/NS3205048102.html

Ipari WiFi kártyák Linuxos támogatással

Mini-PCI szabványú *WiFi* kártyákat jelentett be az *Ubiquiti Networks*, melyeket elsősorban mostoha körülmények közé (-45 +95 *Celsius* fok) és nagy teljesítmény igény (600 mW) esetén ajánl. Az *ExtremeRange2* a 2.4 GHz-en üzemelő 802.11b/g, míg az *ExtremeRange5* az 5GHz-en üzemelő 802.11a hálózatokat támogatja. Mindkét kártya működik a nyílt forrású *Atheros MADWIFI* meghajtóprogrammal. Jelenleg 110 dollár körüli áron kapható.
A második negyedévre várhatóak olyan kártyák is, melyek szélesebb frekvencia tartományban (2.3-2.7 GHz és 5.0-6.1 GHz) üzemelnek majd.

➲ http://www.linuxdevices.com/news/NS7607606463.html

Linux: Meghódítva a Sony PSP is

Chris Mulhearn sikeresen portolta a *uClinux*ot a *Sony MIPS* alapú *PSP (Playstation Portable)* játékgépére. A megoldás pár helyen még hiányos, azonban az látszik, hogy nem lehetetlen a dolog.

➲ http://df38.dot5hosting.com/~remember/chris/

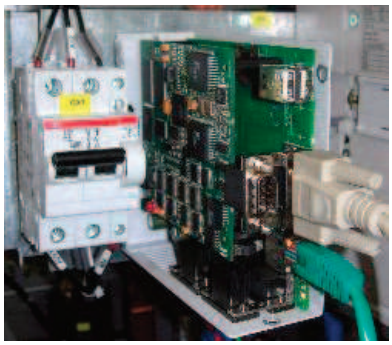
➲ http://www.linuxdevices.com/news/NS3690012215.html

Kínai UMPC GPS vevővel

A kínai *Shenzhen* kínálatában már megjelent legújabb *ARM* alapú *UMPC*-jük (*ultra-mobile PC*), mely beágyazott *Linux*szal (*H9*) vagy *Windows CE 5.0*-ával (*H10*) kérhető. Az eszköz 7 hüvelykes *LCD* kijelzővel érkezik, melynek felbontása azonban ismeretlen. Emellett 520 MHz-es *PXA270*-es processzor, 128 megabájt RAM, 20 gigabájtos merevlemez és *GPS* vevő lapul a burkolat alatt.
A híradás szerint nincs benne se *GSM* modul, se *Wifi*, azonban *SD* vagy *PCMCIA* foglalaton keresztül bővíthető a PC. Az eszköz méretei: 205 x 120 x 19 mm, míg a tömege fél kiló.

➲ http://www.linuxdevices.com/news/NS6848613546.html

Felvillanyozott pingvin



A *Koan* – amely egy olasz, elektromos rendszereket tervező cég – beágyazott *Linux*ot használ kulcsfontosságú rendszerekhez. Az eszköz jelenesetben távérzékelést, adatgyűjtést és irányítást is végez. Az eszköz lelke egy *ARM* processzor, míg az adatgyűjtés soros portokon (*RS232/485*) és digitális bemeneteken történik. Az esetleges áramszünet esetére pedig 512 kbyte-os akkumulátoros táplálású *SRAM*-ot használ a rendszer. Az eszköz 64 megabájt flash és 64 megabájt *SDRAM*-ot kapott. Két *USB* port, egy hálózati csatoló (számos protokollt támogat: *NTP, DHCP*, *FTP*, *PPP*, *SSH*, stb.) és egy digitális hőmérő is helyet kapott az eszközön, mely *Linux*ot (*KaeilOS*) futtat *2.6.17*-es kernellel.

➲ http://www.linuxdevices.com/news/NS2142593744.html

Debian telepítés Windows alól

Robert Millan bejelentette a *Debian-Installer Loader*t, mely lehetővé teszi, hogy *win32*-es rendszer alól telepíthesse a felhasználó a *Debian*t, amennyiben rendszergazdai jogokkal bír. A program induláskor ellenőrzi, hogy 32 vagy 64 bites processzorral van-e dolga, és ennek megfelelő, kernelt és hálózatos telepítést indít el. Ez a programocska az alábbi esetekben jöhet jól:

– *CD* meghajtó nélküli gépre kell telepíteni
– olyan felhasználónak kell telepítenie, aki nem tudja hogyan kell *CD*-t írni vagy a *BIOS*-t beállítani *CD*-ről történő indításhoz
– azonnal telepíteni anélkül, hogy *CD*-t kellene keresni

A windowsos kliens az alábbi linken érhető el:

➲ http://goodbye-microsoft.com/

Ingyen munka?

A *Linux* kernel fejlesztői felajánlották munkájukat: ingyen és bérmentve írnak bármely hardver eszközhöz meghajtóprogramot. Noha néhány hardver cég üzleti titkokra hivatkozva nem adja ki az eszközök leírását, viszont sok cég nem akarja kiadni a kezéből a meghajtóprogram fejlesztését, viszont anyagi eszközökkel és/vagy szakembergárdával sem rendelkezik a megírásához. *Greg Kroah-Hartman* és a kernelfejlesztő csapat ezen szeretne változtatni.

➲ http://www.linuxdevices.com/news/NS3585464168.html

Kisirodai szerver Debian ARM Linuxszal

A tajvani *Embedian ASR-7110*-es fedőnéven piacra dobott egy 400 Mhz-es *ARM9*-es processzorral szerelt *Debian*-t futtató miniszervert. Az eszköz 128 megabájt *RAM*-mal bír, háttértárként pedig laptop merevlemezt használ. (40 gigabájtos a legkisebb, amivel rendelhetjük) Ezenkívül helyet kapott négy darab *USB* 1.1-es host port, egy darab *USB* 1.1-es slave port. Érdekes ötlet: rendelkezik hang ki és bemenettel is. Alaptelepítésben *PHP5, Mysql5, Apache2, Postfix, Authsasld, Mailman, Samba, NFS, nyomtatószerver, Bind* található rajta.
Ezért jelenleg 320 dollárt kérnek.

➲ http://www.linuxdevices.com/news/NS7851725125.html

Podcast a beágyazott Linuxokról

A *TimeSys* cég elindította ingyenes *podcast* szolgáltatását, mely rendszeresen jelentkezik beágyazott *Linux* témában. Egy-egy rész félórás és kb. 20 megabájt. *Ogg* és *mp3* formátumokban tölthető le. A készítők szeretnék a lehető leginteraktívabbá tenni az adást, így szívesen fogadják a témában a telefonokat és az emaileket. A *podcast* angol nyelvű.

➲ http://lldn.timesys.com/learning_center/podcasts/
➲ http://lldn.timesys.com/webinars

Ipari automatizálás beágyazott Linuxon

A *United Electronics Industries* már szállítja *Linux* alapú programozható vezérlőit, melyeket elsősorban ipari felhasználásra vagy ember nélküli járművek vezérléséhez ajánl. Az eszköz roppant strapabíró: -40 és +85 *Celsius* fok között üzemképes, ellenáll az 50 g-s ütésnek és 22 km-es magasságig használható.
Az eszközt egy 400 Mhz-es *Freescale* processzor hajtja. Három vagy hat *PowerDNA* bővítőhellyel rendelhető. Jelenleg már több, mint húszféle ilyen bővítő modul létezik: soros port, analóg bemenet, analóg kimenet, számláló, időzítő, stb.

➲ http://www.linuxdevices.com/news/NS4539300882.html

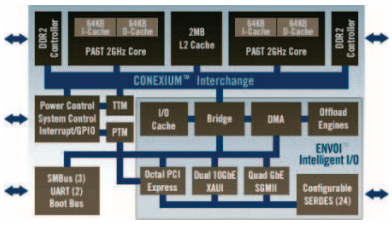
Linuxos videofon

Az *Iwatsu Electric* cég *Linux*ot használt *IP* alapú videótelefonjának megépítésekor. Az eszköz érdekessége: már a tárcsázás is weboldalak segítségével történik. Az *Opera* cég beágyazott eszközökre szánt böngészőjét találjuk rajta, de nem maradt ki az email kliens sem. Elsősorban az üzleti élet résztvevőit célozták meg a termékkel. A rendszer lelke egy *Freescale i.MX21*-es *ARM9* alapú processzor. A telefon támogatja a *H.263*-as és *MPEG4*-es videót, illetve a *G.711*-es tömörítést és a visszhang elnyomást is.

➲ http://www.linuxdevices.com/news/NS8298520771.html

Kétmagos PowerPC chip hamarosan

2007 negyedik negyedévére ígéri a *P.A.* a energiatakarékos 64 bites kétmagos *PowerPC* processzorát, mely várhatóan hálózati és adattárolási, valamint wifi alkalmazásokban fog előfordulni. A *PA6t-1682M* processzor 2 GHz-es, normál esetben 5-13 wattos fogyasztást ígér a gyártó, míg csúcsra járatva se haladja meg a 25 wattot. Ez várhatóan negyede lesz a hasonló képességű 65 nm-es processzoroknak.



A processzor két db *DDR2*-es memóriavezérlővel, két db 10 gigabites és négy db 1 gigabites hálózati kapcsolatot ad. Hardveres gyorsítást biztosít *TCP/IP*-hez, biztonsági algoritmusokhoz és *CRC* ellenőrző összegek kiszámításához. Minthogy még csak pár bemutató darab elérhető, így a piaci ár még nem alakult ki.

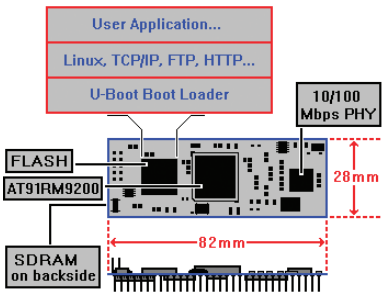
➲ http://www.linuxdevices.com/news/NS5899190894.html

Visual Basic Linuxon is

A *Mono Projekt* bejelentette, hogy létrehoztak egy *Visual Basic* fordítót, mely képes *Visual Basic* forráskódot lefordítani forráskód módosítás nélkül olyan platformokra, amelyeken fut a *Mono*. Eddig csupán *Microsoft Windows* alá lehetett lefordítani a *Visual Basic* forráskódokat, azonban ezután már akár *Linux*ra is, ezzel is növelve a két platform közti átjárhatóságot.
Miguel de Icaza – a *Mono Projekt* alapítója – hozzátette, hogy a fejlesztést semmilyen módon nem segítette vagy hátráltatta a *Microsoft-Novell* egyezség, hiszen az csupán a virtualizációról, a dokumentumformátumok együttműködéséről, valamint a szerverprogramok együttműködéséről szólt.

➲ http://www.linuxdevices.com/news/NS9725385854.html

Linuxos átjáró, pendrivenak álcázva



Tux/Stick néven kezdi hamarosan forgalmazni az *SSV Embedded Systems* legújabb *Linux*os átjáróját. Az eszköz első ránézésre pendrive-nak néz ki.

Az egyik végén – a kupak alatt – valóban egy *USB* csatlakozót találunk, míg a másik végén előre nem definiált csatlakozót, így az lehet bármi: *10/100 Ethernet, SPI, USB* vagy *soros port*.
Az eszköz a működéséhez szükséges energiát a *PC USB* portjáról nyeri, az alacsony energiaigényről egy 180 MHz-es *ARM9*-es processzor gondoskodik. A rendszer 16 megabájt flash memóriát kapott és 32 megabájt *SDRAM*-mal dolgozhat.
Az ára jelenleg még nem ismert.

➲ http://www.linuxdevices.com/news/NS4837503255.html

HP és a Debian

A 2006-os pénzügyi évben a *HP* 25 millió dolláros hardvereladást könyvelhetett el annak köszönhetően, hogy a *Novell* és a *Red Hat Linux*ok mellé a *Debian*t is felvette a támogatottak közé a cég. *Jeffrey Wade* – az nyíltforrású és *Linux*os megoldások igazgatója – elmondta, hogy ez a forgalom a legvadabb álmaikat is felülmúlta. Azt is elmondta, hogy egy új disztribúció bevezetése hatalmas befektetés és gondos mérlegelés után hozzák meg a döntést. Mindenesetre határozottan jó döntés volt, hiszen nagyon sok vevő azért választja a *Debian*t, mert nem jár egyéb kiadással (pl. előfizetés a frissítésekhez) az operációs rendszer.

➲ http://www.internetnews.com/dev-news/article.php/3661481

Elegáns Elive CD

Elive egy *Debian* alapú *Live CD*, mely *Enlightenment* ablakkezelőt használ. Fő célja az esztétikus megjelenés anélkül, hogy a funkcionalitás vagy a sebesség rovására menne. A *CD*-n számos program található: médialejátszó (*Mplayer, XMMS*), videószerkesztők (*Kino, Cinelerra*), hangszerkesztők (*ReZound, ZynAddSubFx, Hydrogen*), irodai alkalmazások (*Abiword, xpdf, Gnumeric*), grafikai alkalmazások (*Gimp, Blender*).

➲ http://distrocenter.linux.com/article.pl?sid=07/01/24/1424218&tid=127

Új Blender kívül-belül

Amellett, hogy a honlap jelentősen átalakult, megjelent a népszerű három dimenziós alkalmazás *2.43*-as verziója is. Számos újdonsággal került a programba.

➲http://www.blender.org/

➲http://www.blender.org/development/release-logs/blender-243/

Svéd haderő: Nekünk is Linux kell

A svéd haderő *Linux*ra váltás mellett döntött. A jelenlegi *Windows NT* szerverek helyett *Red Hat Enterprise* kerül telepítésre. Jelenleg mintegy 200 szerveren futtatnak *Red Hat Linux*ot. A döntést biztonsági okokra hivatkozva hozták meg, de szerepet játszott az erős támogatás és az átállás egyszerűsége is.

➲ http://linux.wordpress.com/2007/02/07/swedish-armed-forces-switches-to-linux/

Sony Playstation 3 előretelepített Linuxszal is rendelhető

Már vesz fel előrendeléseket a *TerraSoft* előretelepített *Linux*os *Sony Playstation 3*-akra, azonban nincs garantált szállítási határidő. 650 dollárért *Yellow Dog Linux*ot kapunk a *Playstation 3* mellé, amelynek a belsejében 60 gigabájtos merevlemez, 256 megabájt (nem bővíthető) *Rambus* memória, gigabites hálózati csatoló, *wifi*, 4 db *USB* és egy *BlueRay* meghajtó bújik meg. A csomagban sem billentyűzetet, sem egeret nem találunk, mint ahogy videójelkábelt is külön kell vásárolnunk.

➲ http://www.linuxdevices.com/news/NS7863481537.html

PDF is ISO szabvánnyá válhat?

Az *Adobe* szoftverevolúciós szempontból elérkezettnek látja az időt, hogy megnyissa a *PDF* szabványt. A *PDF* szabvány mellett szól, hogy noha számtalan módon fejlődött és számos webes technológia része, mégis megőrizte a kompatibilitást a korábbi verziókkal. Ha a *PDF*-et elfogadják *ISO* szabványként, akkor dokumentációmegőrzés szempontjából mindenképp jól járunk, hiszen lesz egy szerkeszthető (*ODF*) és egy „zárolható" (*PDF*) formátum, amelyet nem sajátíthat ki magának egyetlen cég sem.

➲ http://www.consortiuminfo.org/standardsblog/article.php?story=20070129060842557

Zenwalk 4.4

Megjelent a *Zenwalk* disztribúció legújabb verziója. A *Zenwalk* egy *Slackware* alapú *Linux*, mely *Xfce* ablakkezelőt használ, így régebbi gépeken is megfelelő teljesítményt ad. A jelenlegi verzióban az *Xfce 4.4*-es, illetve a kernel *2.6.20* verzióját találjuk. Természetesen letölthetjük a *KDE 3.5.6*, valamint az *OpenOffice.org 2.1.0*-ás verzióját is.

➲ http://www.zenwalk.org/modules/news/article.php?storyid=41

Internetes rádió – akár a konyhában is

A francia *Com One bluetooth* kiegészítőket gyártó cég elkészített egy *Wifi*s rádiót, amely *Familiar Linux*ot futtat. A *Phoenix IP* rádió számos *http* alapú folyamot lejátszik, így az *Icecast*/*Shoutcast* protokollt éppúgy, mint a *Microsoft Media Streaming Protokoll*t. Természetesen *USB*-s eszközökről is hallgathatunk zenét. A lejátszási listák közül az *M3U*, a *PLS* és az *ASX* formátumot támogatja, azonban a *podcast*-ok kedvelőinek jó hír: támogatja az *RSS*-t is.

A *Phoenix* rádió 802.11b/g-s hálózatokat, valamint *WEP* és *WPA* titkosításokat kezel. A jelenlegi verzió nem tartalmaz *Bluetooth*-t, de a későbbiekben várhatóan az is lesz benne.

A rádió gyomrában egy 300Mhz-es *Intel PXA270*-es processzor, 8 megabájt flash és 32 megabájt RAM memória bújik meg, míg az adatok kijelzéséről egy 128x64 képpontos *fekete-fehér LCD* kijelző gondoskodik.

Két darab 2 Wattos hangszóró kapott helyet – 170 Hz és 20 kHz közötti átvitellel –, de hagyományos 3.5 mm-es fejhallgatón is hallgathatjuk az adást. A 250 dolláros rádió természetesen hálózati táplálás mellett a mellékelt négy darab ceruza akkumulátorról is használható, azonban az üzemidejéről nem szól a híradás.

➲ http://www.linuxdevices.com/articles/AT3411564713.html

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?

A *Promise* kiadta a *sata_promise.c* meghajtó által támogatott valamennyi lapkakészlet teljes műszaki leírását. Ez nagyszerű hír, különösen ha azt is hozzá tesszük, hogy a *Promise* eleddig semmiféle dokumentációt nem volt hajlandó átadni a nyílt forrású közösségnek. *Jeff Garzik* egy ideje már tárgyalt a cég képviselőivel, és amint ez a hír is mutatja, nem minden siker nélkül. Talán bátran állíthatom, hogy valamennyiünk örömére szolgál a *Promise* ilyetén pálfordulása.

Karel Zak azon dolgozik, hogy vagy egy új szálat (*fork*) kezdjen a *util-linux* projektből, vagy átvegye annak vezetését *Adrian Bunktól*. *Karel* eddig csak a *Red Hat* csomagot tartotta karban, *Adrian* pedig nem mutatott az üggyel kapcsolatban akkor aktivitást, amekkorát *Karel* titkon szeretett volna. Éppen ezért *Karel* nemrég küldött egy levelet a fejlesztői listára, amiben leírta a terveit, valamint hogy nyitni szeretne egy új *git* tárhelyet és weblapot a projektnek, majd elkezdené egyesíteni a beérkezett hibajavításokat a fejlesztői fával. A célja természetesen az „átadás-átvétel" minél zökkenőmentesebb lebonyolítása, és ebben természetesen *Adrian* is támogatja. Amint pedig arra *H. Peter Anvin* is rámutatott, ennek a legjobb módja az, ha *Karel* egy új fejlesztési szálat nyit meg. Szerinte ugyanis ha *Adrian* már látja, hogy *Karel* valóban képes megbirkózni a feladattal, sokkal nyugodtabban adhatja át a vezetést neki, és hagyhatja, hogy a projekt haladjon a maga útján.

Mikulas Patocka kiadta a *SpadFS* első változatát. Ezt a fájlrendszer doktori disszertációjának részeként készítette, célja pedig az, hogy jobb és egyszerűbb megoldást nyújtson a hirtelen újraindulások kezelésére, mint a naplózó fájlrendszerek. *Mikulas* úgy gondolja, hogy a naplózás módszere nem csak túlságosan összetett, hanem – talán épp ebből következőleg – a hibákra is sokkal érzékenyebb. Szerinte a megoldás az úgynevezett *„baleset-számlálás"* (*crash counting*) jelenti. E módszer lényege, hogy a fájlrendszer meghajtója folyamatosan követi a becstolásokat és leválasztásokat, és ezt az információt hozzáfűzi minden friss adathoz egészen addig, amíg azokat nem sikerül kiírni egy permanens helyre. Ha a számítógép hirtelen leáll, majd újraindul, a fájlrendszer képes érzékelni, hogy a jelenlegi „becsatolási állapota" nem azonos az előzővel, melynek hatására azonnal visszaállhat a legutóbbi, konzisztens állapotba.

A *SpadFS* a legjobb úton halad afelé, hogy bekerüljön a hivatalos kernelbe, olyannyira, hogy még maga *Linus Torvalds* is támogatja. Igaz persze, hogy egyelőre csak annyit mondott róla: *„nem tűnik valami szörnyűnek"*.

A vadonatúj *ext4* fájlrendszer immár hivatalosan is része a rendszermagnak. A jelenlegi állapotában persze az *ext4* inkább csak egy olyan *ext3* fájlrendszernek tekinthető, amit számos extra szolgáltatással illetve folttal is felszereltek. Ugyanakkor az ext4 bebocsáttatása a hivatalos kernelfába sokkal simább folyamat volt, mint más fájlrendszereké, különösen pedig a *ReiserFS*-é. Az *ext3*-tól eltérően azonban az *ext4* továbbra is „mozgásban van", új szolgáltatásokkal fog a jövőben bővülni, és más, jelentős változások is lesznek a kódjában. Pontosan ezek a nagy változások azok, amelyek miatt *Linus Torvalds* azt kérte a fejlesztőktől, hogy adjanak a kódnak külön nevet. Szerinte egy fájlrendszernek teljesen megbízhatónak kell lennie. Ha pedig teljesen stabilizálódik a kód, akkor az adott rendszer – az adott névvel – készen van. Kisebb hibajavítások persze beleférnek ebbe a szabályba, de olyasmi, ami jelentősen veszélyeztetheti a tárolt adatok épségét, vagy egyén, ehhez hasonló mértékű problémákat okozhat, semmiképpen. Egy stabil fájlrendszernek az ilyenek a fejlesztési szakaszban át kell esnie. Pontosan ezért jelent meg az *ext2* új fejlesztési szálaként annak idején az *ext3*, és valószínűleg ugyanez fog bekövetkezni néhány év múlva, amikor az *ext4* teljesen stabilizálódik, és az új fejlesztéseket egyszer csak már *ext5*-nek fogják hívni.

A kernelfejlesztők egészen idáig nem fektettek különösebb hangsúlyt arra, hogy a számítógép energiaforrásának állapota valamiféle konzisztens felületen keresztül jelenjen meg a felhasználó számára. Minden akkumulátortípusnak más-más felülete volt, ami aztán persze jó nagy káoszt eredményezett. *David Woodhouse* azonban a közelmúltban bejelentett egy általános célú akkumulátorkezelési meghajtót, ami egy csapásra szabványosítja az összes eddig használt felületet.

A szerző mellesleg azt tervezi, hogy egy a váltóáramú táplálással kapcsolatos felülettel is kiegészíti a meghajtót, bár ezzel kapcsolatban volt néhány ellenvélemény is. *Richard Hughes* például meg van róla győződve, hogy az akkumulátorok és a váltóáramú tápegységek kellően sok részletben különböznek egymástól ahhoz, hogy külön meghajtót kapjanak. *David* meglátása szerint ugyanakkor a két felület annyira hasonlít egymásra, hogy semmi értelme nem lenne két meghajtót írni hozzájuk. A vita egyelőre folyik, eredmény később várható. Annyi ugyanakkor már most is világos, hogy akár egy, akár két az energiaellátással kapcsolatos meghajtó lesz is, a rendszermag fejlesztői magát az ötletet nagyban támogatják.

Linux Journal 2007., 155. szám

Zack Brown

A tolvaj szent

A legtöbben valószínűleg azt gondolják, hogy a kultúra szabadsága és a szerzői jogok között feszülő ellentét a huszonegyedik század szülötte. Pedig dehogy. Ki hinné például, hogy a szerzői jogok megsértésének első dokumentált eseténél a vádlott egy később szentté avatott szerzetes volt...

Manapság a legkülönbözőbb helyeken hallhatunk a szerzői jogok tiszteletben tartásának szükségességéről, a vélt vagy valós jogsértések társadalmi-gazdasági hatásairól, letartóztatott kalózokról, lefoglalt szerverekről, működő és működésképtelen védelmi rendszerekről, nyert és vesztett perekről. A témáról természetesen könyvek is születtek. Ezek egyikében (***John Gantz, Jack B. Rochester: Pirates of the Digital Millennium – Financial Times Press-Prentice Hall, 2005***) bukkantam arra a nem mindennapi történetre, amiről ez a cikk szól.

A könyvbarát...

Kolumba – egyes megnevezések szerint ***Comcille*** – ír szerzetes volt, aki igen aktív életet élt. Egy tehetős család anyagai támogatásával egymaga három kolostort alapított, és különös hangsúlyt fektetett arra, hogy ezek a kor legjobb vallási témájú könyveivel legyenek ellátva. A gyűjtemény gyarapításából „szerzőként" is kivette a részét, mégpedig két módon is. Egyrészt élete során maga is több mint háromszáz könyvről készített másolatot, amelyek közül némelyek túlélték az elmúlt évszázadokat, és ma is megtalálhatók különböző múzeumokban és könyvtárakban. De voltak más módszerei is... 557-ben például elutazott egy közeli kolostorba, ahol – sajnos a tulajdonos tudta és beleegyezése nélkül – másolatot készített az apát egyik zsoltároskönyvéről és azt magával vitte, gyűjteményét gyarapítandó. A másolás a kor technikájával ugyan napokat vett igénybe, de az akció sikerült. Az apát, ***Finnian of Moville*** – aki különben ***Kolumba*** egyik tanítómestere volt – mikor megneszelte a jogbitorlást, visszakövetelte a másolatot, de nem járt sikerrel. Ezért aztán jogorvoslat végett a királyhoz, ***Diarmaithoz*** fordult, aki el is rendelte a könyv visszaszolgáltatását.
A történetnek itt akár vége is lehetne, de az élet ennél sokkal érdekesebb dolgokat képes kreálni...

Lehetsz király, hiába vagy...

Azt mondják, minden törvény annyit ér, amennyit betartanak belőle. Manapság is gyakori jelenség, hogy az írott törvényeket ki így, ki meg amúgy értelmezi, amiből aztán végeláthatatlan viták, vagy értelmezhetetlen – de minimálisan is meglepő – bírói döntések születnek. Amint ***Kolumba*** példája mutatja, ez a fajta „jogbizonytalanság" sem nevezhető újnak.
Diarmait is király volt, vagyis elvileg maga a törvény, de sajnos nem számolt ***Kolumba*** „nehéz természetével". És ezt rosszul tette. Elkövette ugyanis azt a hibát, hogy nem sokkal az ominózus „könyves eset" előtt megölte ***Kolumba*** egyik rokonát. Ezek után talán nem meglepő, hogy kettejük viszonya nem volt épp felhőtlen, mondhatni perben és haragban álltak egymással.
Kolumbának tehát esze ágában sem volt eleget tenni a királyi ítéletnek, ellenben csatlakozott egy északi lázadó csapathoz, amelynek célja ***Diarmait*** hatalmának megdöntése volt.

A sorsfordító malőr...

És a lázadás sikerrel járt. Már amennyiben ezt sikernek lehet nevezni... A lázadók és a király csapatai a ***Cooldrumman*** melletti csatában – amit ***„A Könyves Csata" (The Battle of Book)*** néven is emlegetnek – ütköztek meg egymással, ahol is több mint háromezer ember vesztette életét, köztük maga a király.
Kolumba nem sokkal ezután elhagyta ***Írországot***, és ***Skóciába*** ment. Hogy ezt jószántából tette, vagy „megkérték rá", nem tudni pontosan. Azt viszont igen, hogy hősünk életében éppen ez volt az a fordulat, amely megnyitotta számára az utat az örökkévalóság felé.

Szolgálat a világ végén...

Manapság egy hittérítő és egy porszívóügynök között túl sok különbség nincs. Mindkettő becsönget, vagy leszólít bennünket az utcán, hülyesége-

ket beszél, majd – esetleges szóbeli áldásainkkal eltöltekezve – békében távozik. Nem volt ez azonban mindig így. A történelem sötétebb korszakaiból számos olyan esetről tudunk, amikor a hittérítő egyáltalán nem úszta meg ennyivel, s attól függően, hogy a világ mely területén végezte szolgálatát, igen gyorsan a sírban, esetleg vacsora gyanánt egy kondérban végezte. Nyilvánvaló, hogy *Kolumba* sajnos nem csak a királynak okozott – utóbbi számára tragikus kimenetelű – fejfájást, hanem a keresztény egyháznak is. Mert igaz ugyan, hogy az ő korában az emberélet nem volt különösebben drága, meg a királyok is jöttekmentek, na de háromezer jó keresztény bestiális lemészárlása – bármilyen okkal történt is – már akkor sem sem volt összeegyeztethető a felebaráti szeretettel, meg a tízparancsolattal. *Kolumba* tehát minden bizonnyal vezekelni ment *Skóciába*, *Iona* szigetére, azzal a céllal, hogy legalább háromezer új hívet szerezzen egyházának. A hely érdekessége, hogy az ötszázas években ez a sziget volt nyugat felé az utolsó ismert szárazföld, vagyis *Kolumba* a szó szoros értelmében a világ végén teljesített szolgálatot. És nem is rosszul, mert bár építeni sokkal nehezebb, mint rombolni, a tervet maximálisan túlteljesítette: gyakorlatilag teljes *Észak-Skóciát* áttérítette a keresztény hitre.

Az alkotás öröme

Kolumba egész életében szenvedélyesen szerette a könyveket. *Iona* szigetén egyetlen órát sem töltött tétlenül: rendtársaival együtt szorgalmasan készítették a másolatokat a legkülönbözőbb kéziratokról. Elsősorban az ő tevékenységének köszönhető, hogy a sokszorosításnak ez a módszere olyan fontos szerepet töltött be az elkövetkező évszázadokban egész *Európában*.
A legenda szerint élete utolsó napján is könyvet másolt. Az utolsó mondat, amit 597. június 9-én, 76 éves korában papírra vetett, így hangzott: *„akik az Urat keresik, nem szűkölködnek”*. Aztán letette a tollat, meghagyta, hogy *Baithene* nevű tanítványa folytassa majd a munkát, egy magaslatról még egyszer utoljára körbetekintett *Iona* szigetén, majd este meghalt.
A furcsa fordulatok pedig még ezzel sem értek véget. Szentté avatásához három szemtanúnak kellett nyilatkoznia arról, hogy látta, amint *Kolumba* csodát tett. A sors kifürkészhetetlen kegyéből ezek egyike éppen az az apát volt, akitől annak idején a könyvet lopta... Így lett *Szent Kolumba Írország*, *Skócia*, és a könyvkötők védőszentje.
És hogy mindebből mi a tanulság? Nos, azt hiszem az lesz a leghelyesebb, ha ennek levonását a nyájas olvasóra bízom...

Büki András
buki@linuxvilag.hu
A Linuxvilág magazin szerkesztője

KAPCSOLÓDÓ CÍMEK

Pirates of the Digital Millennium:
➲ http://www.idc.com/research/pirates_book.jsp

Szent Kolumba élettörténete:
➲ http://www.katolikus.hu/szentek/0609-359.html

A Könyves Csata (The Battle of Book):
➲ http://www.maristmessenger.co.nz/?p=21

Hogyan törjünk fel webhelyeket

Avagy hogyan ne írjuk webalkalmazásokat

Cím: Hogyan törjünk fel webhelyeket
Szerzők: James A. Whittaker, Mike Andrews
Kiadó: Kiskapu (Addison-Wesley)
Oldalszám: 256
Ár: 3980 Ft

Az utóbbi években az internet rohamos terjedésének köszönhetően a programfejlesztés is új irányt vett. Míg 10-12 évvel ezelőtt csupán statikus weblapok voltak jelen a hálón, a programok nagyobb része pedig helyben futott a számítógépen, addig ma már szinte minden áthelyeződött a világhálóra. Legyen szó valamilyen rendelésről (könyv, virág, pizza, aszpirin, viagra stb.), banki műveletről, aukciós oldalról, közösségi webhelyről vagy bármi másról, nyitjuk a böngészőt, és a világ legtermészetesebb módján használjuk, amit a webfejlesztő elénk rakott.
Az oldalak dinamikussá válásával azonban képbe került egy új probléma is: hogyan írjunk olyan webalkalmazásokat, amelyekkel mind a felhasználó, mind az üzemeltető biztonságban lehet anélkül, hogy az túlzottan megbonyolítaná a napi ügymenetet.
Manapság már számtalan nagyon jónak mondható webprogramozásról szóló könyv kapható itthon is.
A probléma az, hogy ezekből – akárcsak a többi, klasszikus programozási tankönyvből (*C*, *C++* stb.) – gyakran kifelejtettek egy fontos részt a szerzők.
A programozás elméletének és szintaxisának, meg az alkalmazási területeknek a bemutatásával sajnos nem érhet véget egy tankönyv. Nem csak a világos oldalt kell megvizsgálni ugyanis, amikor a felhasználó a tankönyvi példa szerint viselkedik, hanem a sötétet is, amikor – akár szándékosan, akár véletlenül – megpróbálja kihozni az alkalmazásból azt, amit mi egyáltalán nem terveztünk bele.
E tekintetben ***Whittaker*** és ***Andrews*** könyvét amolyan hiánypótló műnek tekinthetjük, hiszen a két szerző – tapasztalt biztonságtechnikai szakemberek – 24 lehetséges feltörési módszer tárgyal hét logikai részre tagolva. Minden betörési módszernél részletes leírást kapunk magáról a támadásról – mi is valójában és mikor kell rá odafigyelni –, a támadás végrehajtásáról és persze a támadás elleni védekezésről.
A három további fejezetben a web természetéről, az adatvédelemről és a webszolgáltatásokról kapunk áttekintést. A függelékben helyet kapott ezen kívül egy rövid – tíz oldalas – összefoglaló a szoftverfejlesztés történetéről a kezdetektől napjainkig, valamint a könyvben bemutatott virágboltos weboldal hibáinak rövid összefoglalása.
Apropó virágbolt... A könyvhöz tartozik egy ***CD*** melléklet is, melyen a könyvben említett segédprogramok és egy „állatorvosi ló” gyanánt használható webhely forrása található. Az eszközök elsősorban windowsosak, de számos alkalmazás – mint például az ***Ethereal*** – megtalálható a legtöbb ***Linux*** terjesztésben is. Emellett három, a témával kapcsolatos ***Firefox*** bővítmény is helyet kapott a lemezen.
A legfontosabb tanács talán az, hogy hibát követünk el, ha a felhasználót alapból jóindulatúnak tekintjük. Ha ezerből csak egy van, aki kihasználja az általunk ottfelejtett biztonsági rést, az eset már akkor is romba döntheti a cégünket. Anyagilag és erkölcsileg egyaránt.
Teljes védelem persze szintén nincs, mint ahogy örökké tartó sem. Újra és újra felül kell vizsgálnunk a kész alkalmazásokat is. A könyv ehhez és további támadási módok kigondolásához is segítséget nyújt.
A hátsó borítón ugyan „Haladó” szint van megjelölve, én azért mégis ajánlanám ezt a művet bárkinek, aki valamilyen szinten webprogramozással foglalkozik, akár a szerver akár a kliens oldalán. A hibás programozói magatartást ugyanis sokkal nehezebb levetkőzni, mint körültekintően programozni már a kezdetektől fogva.

Medve Zoltán
e-medve@e-medve.hu

Korábban elképzelhetetlen szintű nyílt forráskódú fejlesztések Magyarországon és a régióban

Stratégiai nyílt forráskódú együttműködést kötött a Novell és a Synergon

Szittya Tamás, a ***Novell Magyarország*** ügyvezető igazgatója és ***Ákos György***, a ***Synergon Informatika Nyrt.*** vezérigazgatója február elején közös sajtótájékoztatón jelentették be régió szintű stratégiai együttműködésüket a nyílt forráskódú technológia támogatására. A bejelentés értelmében mostantól a ***Linux*** – a nyílt forráskódú technológia – a ***Synergon*** egyik fontos stratégiai platformjává válik, mint a jövőbeni fejlesztések egyik alapja, és mindehhez stratégiai partnere a ***Novell*** és annak vezető ***Linux*** technológiája a ***SUSE Linux Enterprise***. A vezető elemző társaságok előrejelzése szerint a ***Linux*** jövője, robbanásszerű fejlődése elkerülhetetlen és ezt már a ***Microsoft*** és a ***Novell*** által aláírt megállapodás is tovább segíti. Ezért a ***Synergon*** fejlesztéseit az ügyfél-igényeknek megfelelően nyílt forráskódon is piacra viszi.

Ákos György, a ***Synergon Informatika Nyrt.*** vezérigazgatója elmondta, hogy a vezető rendszerintegrátorok körében ***Magyarországon*** elsőként a ***Synergon*** döntött stratégiai szinten a nyílt forráskódú fejlesztések előtérbe helyezéséről, hogy valódi alternatívát biztosítson a felhasználók költségcsökkentési, hatékonyságnövekedési és egyre növekvő biztonsági elvárásaira, melyek alapvető követelmények a cégek és intézmények számára egyaránt. A ***Microsoft-Novell*** együttműködés jó lehetőség egy olyan rendszerintegrátornak, mint a ***Synergon*** Informatika, amely érdekelt több, egymás mellett létező technológia együttes szállításában, beleértve a kapcsolódó szolgáltatásokat is. A mostani megállapodás nagyban segíti a ***Synergon*** üzleti aktivitását, hiszen a korábbiakhoz képest sokkal szélesebb ügyfélkört és igényeket tud kiszolgálni. Ez a megállapodás nem befolyásolja a ***Synergon*** és a ***Microsoft*** között továbbra is fennálló együttműködést, amelyet a Társaság változatlanul fontos fejlesztési platformnak tekint. A vállalat nagyra értékeli, hogy a ***Novell*** is a ***Synergont*** választotta partneréül.


Szittya Tamás


Ákos György

Szittya Tamás, a ***Novell Magyarország*** ügyvezető igazgatója szerint a vezető szoftver és hardvergyártók termékfejlesztési és support támogatása, valamint a felhasználók egyre növekvő nyílt forráskódú igényei mellett a ***Synergon*** belépésével a ***Linux*** kör bezárult ***Magyarországon*** is. ***Magyarország*** vezető rendszerintegrátorainak bekapcsolódásával minden adott a Linux és a további nyílt forráskódú megoldások robbanásszerű elterjedéséhez az államigazgatás és a versenyszféra számára egyaránt. A ***Novell SUSE Linux*** már számos intézménynél bizonyított a magyar kormányzati piacon és a versenyszférában egyaránt, de a ***Synergon*** támogatásával a ***Novell*** reményei szerint hamarosan egyre több ágazatban válik stratégiai platformmá.

Régiós szerep

A ***Novell Magyarország*** és a ***Synergon Informatika*** együttműködése ***Magyarország*** mellett a régióra is kiterjed, hiszen mindkét társaság regionális irányító szerepet is ellát. Ennek köszönhetően a két társaság a nyílt forráskódú fejlesztések magas megtérülésére számít, hiszen

A nyílt forráskódú szoftverek, és a ráépülő megoldások jelentős európai térnyerését mutatja, hogy az elmúlt években az Európai Unió tagállamainak meghatározó vállalatai és kormányzati intézményei tértek át a nyílt forráskódú megoldások használatára. A ***Linuxra*** történő átállást bejelentők közül a legjelentősebbek vállalatok az ***Audi***, a ***PSA Peugeot Citroen***, a ***Siemens Medical Solutions***, illetve ***Európa*** legnagyobb vasúttársasága, a ***Deutsche Bahn***. Az államigazgatásból a ***Linux*** mellett döntött többek között ***Bergen*** város önkormányzata, a svájci ***Szövetségi Kormány***, a spanyol ***Miniszterelnöki Hivatal***, az ***Egyesült Királyság Egészségügyi Minisztériumának*** vezető ügynöksége, a finn ***Védelmi Minisztérium***, a németországi ***Földnyilvántartó Hivatal***, a németországi ***Központi Információvédelmi Hivatal***, az angliai ***Kent*** rendőrsége, a holland ***EPE*** közigazgatási terület törvényhozása, az olaszországi ***Velence*** tartomány, illetve ***Prága*** város önkormányzata is. A hazai példák közül az ***Országos Igazságszolgáltatási Tanács Hivatala***, a ***Közlekedési Főfelügyelet***, illetve a ***Magyar Szabadalmi Hivatal Linux*** bevezetései is kiemelkedő méretű projektek.

a fejlesztések könnyen lokalizálhatók és implementálhatók a környező országokban is.

A Linux fejlődése töretlen – világviszonylatban és Magyarországon is

A felhasználóknak szükségük van megbízható, költséghatékony és biztonságos megoldásokra az informatikai rendszereik fejlesztésében, ezért évről-évre folyamatosan nő a ***Linux*** rendszert telepítők és használók száma mind szerver, mind pedig desktop oldalon. A szoftverfejlesztők és felhasználók, valamint a független elemzők véleménye megegyezik abban, hogy a nyílt forráskódú és ezen belül a ***Linux*** alapú megoldások alkotják ma az ***IT***-iparág legdinamikusabban növekvő területét, és a közeljövőben a linuxos technológiák további előretörése várható. Az ***IDC*** független elemző cég prognózisa szerint 2008-ra a teljes ***Linux***-piac értéke – beleértve a szervereket, asztali gépeket és a szoftvereket is – meghaladja majd a 35 milliárd dollárt, amelyből tisztán a szoftverek várhatóan több mint 14 milliárd dollárt tesznek majd ki.

A Linux térnyerése az Európai Unióban

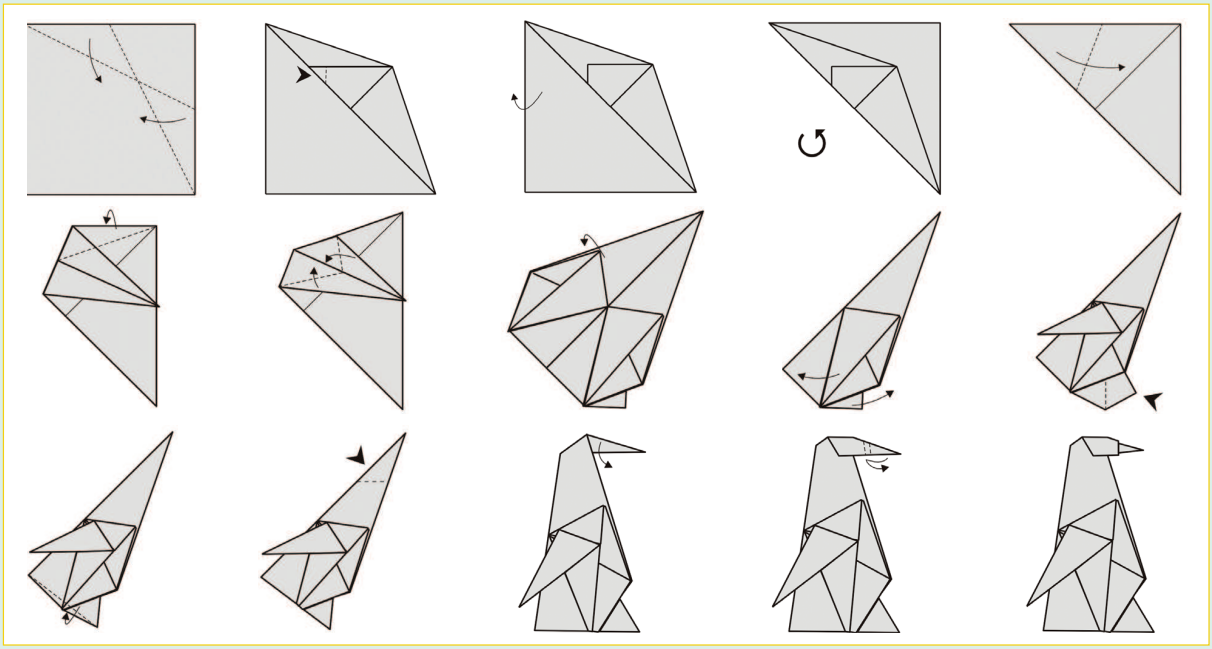
Az ***Európai Bizottság*** januárban kiadott jelentése szerint 2010-re az ***Európai Unió*** összes ***IT*** szolgáltatásának 32%-a nyílt forráskódú szoftverekre fog épülni. A jelentés készítői szerint a nyílt forráskódú szoftverek használata jó esélyt biztosít arra, hogy Európa elérje politikai céljait, miszerint az európai gazdaságnak 2010-re világszinten az egyik legversenyképesebb gazdasággá kell válnia. A kutatók úgy gondolják, hogy az ***Európai Unió*** ingyenesnyílt forráskódú szoftverekre (***FLOSS*** – ***Free/Libre and Open Source Software***) fordított beruházásainak megduplázása a teljes ***EU GDP*** 0,1 %-os éves növekedését eredményezi. Ez több mint 10 milliárd euró megtakarítást jelent évente az ***Európai Unió*** számára.

Pingvin a négyzeten

Egy régi vicc szerint Trabant gyártásához két ember szükségeltetik: az egyik hajtogat, a másik ragaszt. A japánban született origami ezt a gyártást elavulttá tette; így ha akár egy Trabanthoz, akár egy pingvinhez szeretnénk jutni (olcsón és büntetlenül), megspórolható a ragasztóanyag.

A papírhajtogatás művészetén elmerengve ***Jane Austen*** papírra vetett sorai jutnak eszembe, miszerint „a világ egyik fele értetlenül nézi a másik felének szórakozásait”… Hogy miért jó papirosból különféle formákat hajtogatni? Mindenesetre, mióta olvastam, hogy ***Ausztráliában*** kengurutrágyából is készítenek papírt, ***Afrikában*** pedig elefántürülékből, meggondolom, érdemes-e ezt a hobbit választanom. Ha azt vesszük, hogy 100 db A4-es laphoz körülbelül 6 kilogramm trágyát használnak fel – szinte kizártnak tartom, hogy valaha is megnyaljak még egy borítékot.
No de hogy mégis értelmet találjak ebben a játékban (bár talán nem véletlen, hogy a tudósok értelmes életet is csak a ***Földön*** kívül keresgélnek), el kell ismernem, hogy sokaknak már a nadrághajtogatás is bonyolult művelet, ezért meglehet, számukra kiváltképp előnyös a gyakorlást papíron elkezdeni. Első feladatként (nadrágvasalást követően) megpróbáltam rácáfolni ama állításra, mely szerint semmilyen papírt, legyen az bármilyen vékony vagy nagyalakú, nem lehet hétnél többször félbehajtani. Kísérletem sikertelennek bizonyult... Ekkor úgy döntöttem, új hobbiként áttérnék inkább a húszezresek hajtogatására (még akkor is, ha ehhez galandférgek tömkelegét használnák fel) – de mivel a bankok már zárva voltak, fekete-fehérben kezdtem neki.

Halusz Léna



Bővítmények fejlesztése Eclipse-hez

Bevezetés az Eclipse nézeteinek, szerkesztőinek és beépülő moduljainak világába.

A következőkben bemutatjuk az *Eclipse* platformra épülő alkalmazásfejlesztési környezetekhez tartozó beépülő modulok (***pluginek***) fejlesztésének praktikáit. A cikkben található általános alapelvek a letölthető verzión túl számos *Eclipse* alapú fejlesztési környezetre érvényesek. A fejlesztési folyamat több szempontját is érintjük, például a nézet kontra szerkesztő vitát, a belső vagy külső mód kiválasztását, a szabványos ***widget*** eszközkészlet (***Standard Widget Toolkit – SWT***) alapjait, az *Eclipse* beépülőmodul-varázslójának (***Plugin Wizard***) hasznosságát. Az *Eclipse* beépülő modulok *Eclips*-szel való fejlesztésének előnyeit is tárgyaljuk. Egy egyszerű beépülő alkalmazásmodul fejlesztésének lépéseit is végigkísérjük, nem tévesztve szem elől a más modulokban való újrahasznosítás lehetőségét.

A nézet kontra szerkesztő vita

Az *Eclipse*-ben kétféleképpen tálalhatjuk az információt a felhasználó számára: nézetben vagy szerkesztőben. Mindkettő lehetővé teszi, hogy a beépülő modul egyes műveleteit végrehajtsuk az elemeken, szimpla vagy dupla kattintással, illetve a jobb kattintásra előugró menüből vagy a főmenüből való kiválasztással.

A szerkesztő osztály gyakorlatilag mindent tud, amit a nézet, sőt, sokkal többet. Az extra funkciók azonban nincsenek ingyen, sem a rendszer, sem a kód bonyolultságát tekintve. A szerkesztő osztály fejlesztése általában több munkát igényel, mint a nézet, így a munka megkezdése előtt nem árt elgondolkodni azon, érdemes-e.

Ha az információk egyszerű megjelenítése és néhány beépített tulajdonság kihasználása a cél, akkor a nézet elegendő. A nézetbe egyszerűen vihetünk be adatokat, többnyire az *SWT* widgetjeivel, például táblázatokkal és szövegablakokkal. Mi van azonban akkor, ha egy sokkal szabadabb formájú kapcsolatot kívánunk megvalósítani a felhasználóval? Hogy oldjuk meg az *Eclipse* többszöri indításánál is fennmaradó felhasználói adatbeviteleket?

A vitában az egyik jó érv a perszisztencia kérdésének felvetése. Bár van rá lehetőség, hogy egy nézetből adatokat nyerjünk ki, valamiféle perszisztens tárolóból, ez többnyire némi fájlokkal, vagy fájl-szerű kontextusokkal való munkát jelent. Ebben az esetben gyakran egyszerűbb szerkesztőt készíteni.

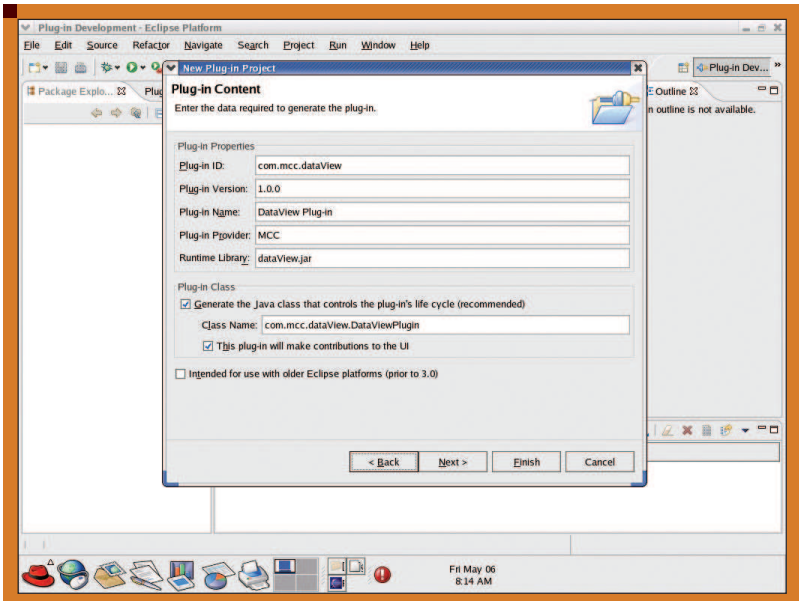
A második leggyakoribb szempont a megjelenítendő adat. Ha a felhasználónak többféle adat kiválasztására szeretnénk lehetőséget adni úgy, hogy ezeken egyesével végezhessen műveleteket, akkor a műveletek megvalósítása rendszerint külön nézetben egyszerűbb.

Néhány sorral később implementálni fogunk egy nagyon egyszerű *Eclipse* beépülő modult, amelynek egyetlen célja, hogy alkalmazásszintű adatokat biztosítson a felhasználónak. Az adatokat karakterláncként jelenítjük meg, de ezt lényegében bármivel helyettesíthetnénk. Engedélyezzük a szokásos bal, jobb és dupla kattintást, de csak ez utóbbit módosítjuk a többi művelet implementálásában való újrahasznosítás szemléltetésére.

Mivel nincs szükségünk perszisztens erőforrásra, és az adatokat egyesével akarjuk kiválasztani, ugyanazt a beépülőmodul-funkciót nézetként valósítjuk meg, amelyet egyszerűen *DataView*-nak hívunk.

A belső vagy külső mód kiválasztása

Mielőtt eldöntjük, hogy nézetet vagy szerkesztőt fejlesztünk, még



1. ábra Új beépülő modul készítéséhez indítsuk el a varázslót a File > New > Plug-in Project menüpontban

egy kérdésre kell felelnünk: a felhasználó számára megjeleníteni kívánt adatokat az *Eclipse* környezeten belül vagy kívül tároljuk? Az *SWT*-ben lévő *Form* osztályok lehetővé teszik, hogy az alkalmazás adatait külsővé tesszük („externalizáljuk”).

A szerkesztők lehetnek külsők és belsők egyaránt, azonban a külső szerkesztőnél nehéz hozzáférni a beépülő modulhoz. Egyes beépülő modulok meglepő módon épp ezt a funkcionalitást támogatják. A legtöbb esetben ennek az az oka, hogy a gyártók a felhasználókat kizárják a beépülő modulok bizo nyos szintű *Eclipse*-funkcióinak használatából. Nagy általánosságban – az eszközökben lévő nyitottságról szóló felhasználói viták ellenére is – azt mondhatjuk, hogy az *Eclipse* szerkesztőket belsőként kell megvalósítani. Egyszerűen nincs értelme lemondani a beépülő modul egyéb funkcióiról, ha nem muszáj. De mi a helyzet a nézetekkel?

A szerkesztőkhöz hasonlóan a nézet is megvalósítható külsőként, külön *Form* osztállyal, és belsőként is, mint egy szokványos nézet, járulékos widgetekkel. Nincsenek kőbe vésett szabályok, de a választást megkönnyíthetjük néhány alapelv észben tartásával. Általában két dolgot kell figyelembe vennünk: tekinthetjük-e a nézet adatait olyan, egyedi, egymástól elkülönült elemeknek, amelyek mezői vagy műveletei az egyes adatokra jellemzők, illetve úgy kilencnél kevesebb van-e ezekből az elemekből? Ha igen, táblázattal, esetleg elemenként külön füllel, nézetként megvalósítható.

Ha azonban az adatelemek és a rajtuk végezhető műveletek tényleges száma változó vagy ismeretlen – például mert a fejlesztőnek nincs előzetes ismerete arról, hogy hány elem várható, mennyi különféle műveletet kell támogatnia, illetve engedélyeznie a jövőben, – a legjobb, ha a nézetet külső *Form* osztályként valósítjuk meg.

A mintául szolgáló beépülő modul egy egyszerű, 100 elemű implementáció, amely mindegyik adatelemet két mezővel, névvel és értékkel jelenít meg. Bár a kilencnél több művelethez nincsenek előre definiált rendszerkövetelmények, nincs több művelet sincs megadva. Ennek megfelelően azt is feltételezhetjük, hogy ilyenekre nem kifejezetten lesz szükség – elvégre egyszerű példát szeretnénk bemutatni. A *DataView* beépülő modult tehát belső nézetként implementáljuk.

Ismerkedés az Eclipse környezettel

A beépülő modul fejlesztéséhez először az *Eclipse*-et kell telepíteni. A példa kedvéért az *Eclipse* weboldaláról letöltöttük az *Eclipse* legfrissebb változatát, amely a cikk írásának idején 3.0.2 volt (Az *Eclipse SDK* 2007. január 22-i verziója 3.2.1. – a *ford.*)). Mivel saját munkahelyemen a *CDT*-t használom a *C* és *C++* fejlesztésekhez, a *CDT* projekt 2.1-es változatát is letöltöttem, ami az *Eclipse* weboldalának *projects* ugrópontjáról elérhető *Eclipse Tools Projectben* található meg. Mindkettő elérhető *.zip* fájlként, amelyek kicsomagoláskor a */eclipse* könyvtárba kerülnek, ezért figyeljünk arra, hogy először az *Eclipse* csomagot telepítsük. Én *Red Hat Linux 9.0*-át használok az *Eclipse* keretrendszer és a *CDT GTK*-s verziójával, de a *Motifos* verziók is ugyanolyan jól működnek. Ezután már elindíthatjuk az *Eclipse*-t a *./eclipse* paranccsal, és a *Windows > Open Perspective* menüpontból kiválaszthatjuk a beépülő modul fejlesztésére szolgáló perspektívát (*Plugin Development Environment – PDE*).

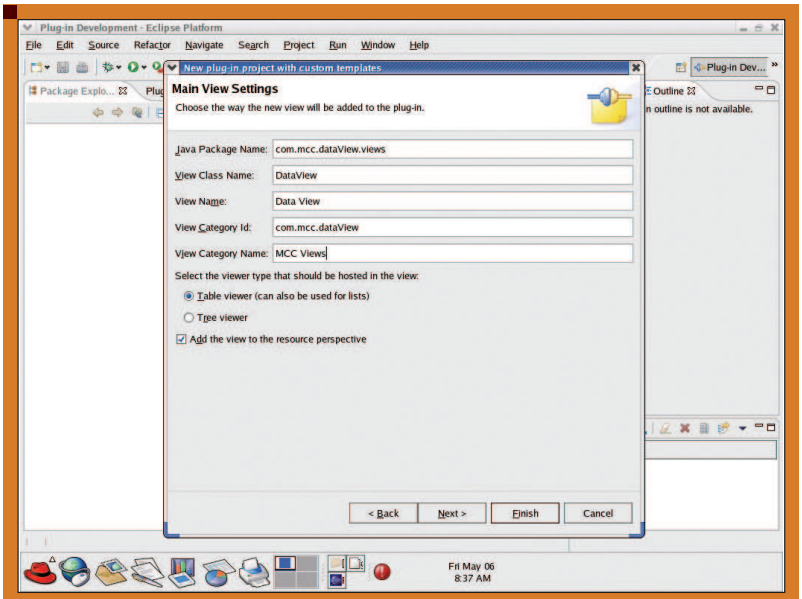
Az Eclipse beépülőmodul-varázslójának használata

Az *Eclipse* beépülő moduljainak fejlesztéséről szóló írások közül sok csak a „Helló Világ” típusú példán vezeti végig a felhasználókat. Nem kizárt, hogy ez jó kiindulási alap lehet kezdő fejlesztőknek, de hitem szerint, nincs ennél rosszabb módja, hogy a gyakorlott szoftverfejlesztők megismerkedjenek az *Eclipse*-szel. Túl sokáig tart, és ami ennél is kellemetlenebb, szinte az egészet elölről kell kezdeni, ha egy valódi beépülő modult kell készíteni. Ehelyett azt javaslom, hogy amennyire csak lehet, teljes beépülő modult készítsünk, a meglévő sablonokból annyit felhasználva, amennyit csak a környezet megenged. Ha így teszünk, rögtön számos funkció áll majd a rendelkezésünkre, amelyek testreszabása anélkül elvégezhető, hogy a normál beépülőmodul-környezethez való megfelelő kapcsolódás miatt aggódni kellene. A *PDE* beépülőmodul-varázslójával egyszerűen készíthetünk

beépülőmodulprojekt-mintát, a ***File > New > Plug-in Project*** menüpontban. A névválasztásnál a más gyártók által is használt konvenciót követjük: ***com.vállalatNeve.termékNeve***, ami a jelen példában ***com.mcc.dataView***, ahogy az *1. ábra* is mutatja.
Ha már a testreszabásnál tartunk, meglehetősen egyszerű feladat egyes funkciók eltávolítása, így a következő két képernyőt átugorjuk, és a sablonokkal folytatjuk. Kiválasztjuk a ***Create a plugin***-t az egyik sablonnal, és a ***Custom Plugin Wizard***-dal, majd a ***Next*** gomb megnyomására már láthatjuk is a létrehozni kívánt sablonokat.
Már ezen a ponton is eltávolíthatnánk funkciókat, de ebben a példában ezt nem tesszük. Ehelyett addig nyomkodjuk a ***Next*** gombot, míg a ***Main View Settings*** panelhez nem érünk, ahol a nézetmintát átnevezzük ***DataView***-ra, a *2. ábra* szerint.
Ha mindent megfelelően beállítottunk, a ***Finish***-sel befejezhetjük a műveletsort, de végiglépkedhetünk a testreszabás utolsó lépésein is (***View Features***). Minden képernyőnél szabadon léphetünk egy előző vagy következő stádiumba, úgyhogy nyugodtan próbálgassuk a lehetőségeket.
A ***Finish*** gomb megnyomásáig semmiféle változtatás nem történik a környezetben.
Ha az első próbálkozás nem sikerül, ahogy például nekem sem sikerült, ne keseredjünk el. Töröljük az egész projektet, a könyvtár tartalmával együtt, és csináljuk mindaddig újra, amíg nem sikerül. Mihelyt a beépülő modul elkészül a követelményeink szerint, máris futtathatjuk, amihez a futásidejű munkapadot használjuk.

A beépülő modul tesztelése a futásidejű munkapadon

Az ***Eclipse*** keretrendszer egyik legnagyszerűbb tulajdonsága, hogy a beépülő modulokat a saját futásidejű munkapadján fejleszthetjük, tesztelhetjük és javíthatjuk. Nagyon kevés fejlesztői környezet kínálja ugyanezeket a lehetőségeket, ilyen könnyen és intuitívan kezelhető módon. Más eszközökkel a fejlesztők gyakran sok időt pocsékolnak el a fordítás, szerkesztés és hibakeresés ciklusának hosszadalmas ismétlésével.

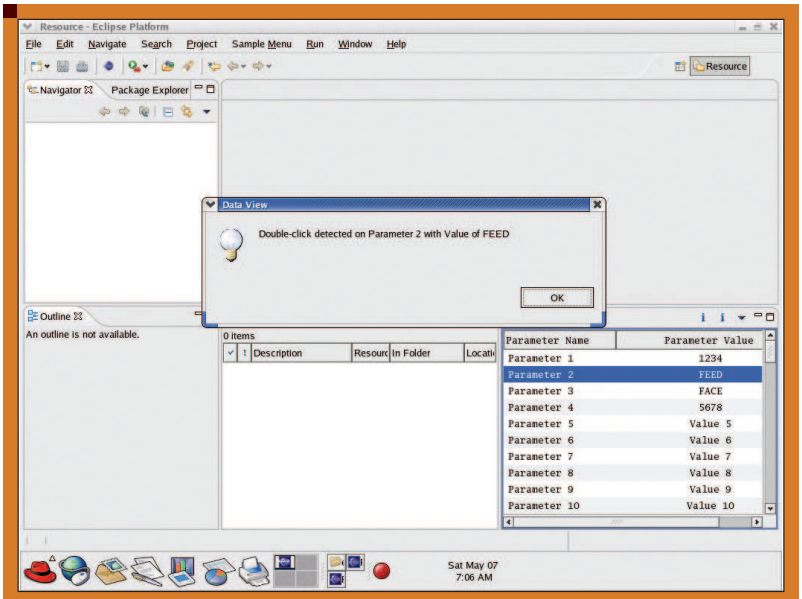


2. ábra Adjunk nevet a beépülő modulnak a Main View Settings ablakban

A ***DataView*** beépülő modul végrehajtásához egyszerűen válasszuk a ***Run > Run As > Run-time workbench*** menüpontot a ***PDE*** perspektívából. A ***PDE*** erre létrehoz egy teljesen elkülönült munkaterületet, amit futásidejű munkaterületnek is hívunk, és végrehajtja a ***DataView*** beépülő modult. Az első futtatáskor a ***Window > Show View > Other*** menüpontra is szükségünk lesz, ahol a beépülő modul létrehozásakor kiválasztott nézetek alatt lévő ***DataView***-t is ki kell jelölnünk.
A futásidejű munkapad a szokásos munkaterülethez hasonlóan működik, így megőrzi a nézet tulajdonságait az egyes indítások között. Ez nagymértékben leegyszerűsíti a tesztelést, mivel az ismételt ellenőrzéshez mindössze újra el kell indítani a futásidejű munkapadot.
A futásidejű munkapad modelljének kevés hátránya közül az egyik az, hogy közel megduplázza a memóriafelhasználást, mintha két ***Eclipse***-t használnánk ugyanazon a számítógépen. Kevés memóriával ellátott eszközökön, például hordozható számítógépeken ez a sebesség nagymértékű csökkenése miatt zavaró lehet. A Java virtuális gépek folyamatos javulásával azonban ez is egyre kisebb gond lesz.
Kísérletezzünk kicsit a beépülő modul menüivel, legördülő menüivel, hogy lássuk, milyen funkciókat hoztunk létre. Bár a szerkesztők testreszabását nem tárgyaljuk, érdemes lehet időt szakítani egy ***.mpe*** kiterjesztésű fájl elkészítésére is, egy egyszerű ***Eclipse*** projekt keretében. Ennek segítségével a kíváncsi olvasó megismerkedhet a többoldalas szerkesztők világával, amilyen pl. az új beépülőmodul-projektben található ***plugin.xml*** fájl megjelenítésénél is látható.

A beépülőmodul-nézet testreszabása

A ***DataView*** tesztre szabásának első lépéseként egy új ***ViewLabelProvider*** osztályt kell hozzáadnunk a ***Views*** könyvtárban lévő beépülőmodul-projekthez. Ezzel adatokat adhatunk a táblához, amit a beépülő modul a ***DataView*** ablakban futás közben megjelenít. A ***ViewLabelProvider*** osztály a tárolt adatot, egy nevet és egy értéket, átadja a ***DataView***-nak, a ***ParameterControl*** osztály segítségével. Ezt az osztályt teljes terjedelmében a projekt tar állományában találjuk.
A következő lépés a ***ViewLabelProvider*** osztály által hivatkozott ***ParameterControl*** osztály hozzáadása a ***Views*** könyvtárban lévő beépülőmodul-projekthez, amely



3. ábra Az elkészült beépülő modul reakciója a felhasználói inputra

a *DataView* táblájában megjelenítendő tényleges paraméterek nevének és értékének kezeléséért felelős. Bár ez egy viszonylag egyszerű implementáció, szükség esetén könnyedén kiterjeszthető nagy számú mezőre is. Ennek az osztálynak a leírását is a projekt tar állományában találjuk. Harmadik lépésként hozzáadunk a *DataView* osztályhoz egy táblát a megfelelő beállításokkal (lásd az 1. tételt a *Linux Journal FTP*-helyén), módosítjuk magát a *Plugin* osztályt a *UserParameter* változó támogatásához és a *DoubleClick* eseményhez hozzárendeljük a táblában lévő adatok megjelenítését. Vegyük észre, hogy egy új függvényt, az *UpdateTheTable*-t is létre kellett hozni, hogy a táblában mindig a legfrissebb adatok szerepeljenek. Ha új alkalmazásadatokhoz kell hozzáférnünk a fájlrendszeren, a hálózaton keresztül vagy valahogy máshogy, ezt a függvényt kell módosítani. Példánkban az első négy paramétert változtattuk meg. A beépülő modul teljes kódja a *Linux Journal FTP*-helyéről letölthető (lásd a kapcsolódó anyagokat). Az utolsó lépésben a beépülő modulban létrehozzuk a `ParameterControl` változót és kezdőértékkel látjuk el. Ezt a *DataViewPlugin.java* fájlban, közvetlenül a `resourceBundle` deklarációja után tegyük meg, a következő módon:

```
//User Parameter functionality
public ParameterControl
➥userParameters[] =
    new ParameterControl[100];
```

Végül a kezdeti értékadás következik a Plugin konstruktora után:

```
// Additions for User Parameter
➥functionality
int index;
for (index = 0; index < 100;
➥index++)
{
    userParameters[index] =
        new ParameterControl
        ➥("Parameter "
       + (index + 1), "Value "
       ➥+ (index + 1));
}
```

A *3. ábrán* a beépülő modul végrehajtása látható a futásidejű munkapadon. Az ábrán azt látjuk, amikor a felhasználó duplán kattintott egy elemre, aminek hatására a táblában kiválasztott elem adatai megjelennek egy egyszerű párbeszédablakban.

Összefoglalás

A cikkben áttekintettük azokat a főbb alapelveket, amelyeket az alkalmazásadatok megjelenítésére szolgáló, nézetként megvalósított *Eclipse* beépülő modul fejlesztésénél figyelembe kell vennünk. Felhasználtuk az *Eclipse* beépülő-modul-varázslóját az inicializációs kód jelentős részének automatikus létrehozásához, tesztelhető és újrahasznosítható módon. Az *SWT* felhasználásának néhány példáját is áttekintettük, például *Table*, *Viewer* és *LabelProvider*, illetve ezek alkalmazását a felhasználói nézetben. Végül rámutattunk az *Eclipse* futásidejű munkapadjának egyes előnyeire és hátrányaira. Mindeközben létrehoztunk egy egyszerű, mintául szolgáló beépülő modult, amely újra és újra felhasználható az új modulok kiindulási alapjaként. A jövőbeli igények kielégítésére bekapcsolható az összes jellemző funkció, például a többoldalas szerkesztők, tulajdonságok, varázslók és referenciakiterjesztések. A további képességek már iteratív módon is fejleszthetők, ami nem csak az adott beépülő modul jövőbeli növekedését teszi lehetővé, de a különféle modulok fejlesztésében való újrahasznosítását is. Az *Eclipse*-t egyáltalán nem vagy csak kevéssé ismerő fejlesztők számára is jó kiindulópont lehet. Az újrahasznosítás az *Eclipse* keretrendszer egyik legfigyelemreméltóbb sajátossága.

Linux Journal 2006., 143. szám

Mike McCullough az MCC Systems elnöke és ügyvezető igazgatója. A Bostoni Egyetemen megszerezte a számítógépek tudományának baccalareusa és a rendszertervezés magisztere címeket. Számos posztot tud maga mögött a Wind River Systems, Lockheed Sanders, Stratus Computer és az Apollo Computer cégeknél, egy 20 éves múlttal rendelkező elektronikai veterán. Az MCC Systems Eclipse alapú szoftverfejlesztési eszközöket készít és a beágyazott rendszerek témakörében kínál oktatási és konzultációs szolgáltatásokat.

KAPCSOLÓDÓ CÍMEK

- www.linuxjournal.com/article/8789

Valódi konszolidáció virtuális adatbázissal

A konszolidáció kifejezéssel, mint az IT költségek racionalizálásának módszerével szinte minden üzemeltetéssel foglalkozó szakember találkozott már. A hardver- és szoftvergyártók marketinganyagaiban rendre felbukkan ez a kifejezést, amit persze mindenki a saját maga módján értelmez.

Abban azonban megegyeznek, hogy az erőforrások jobb kihasználását, a korábbi szigetszerű, elosztott megoldások egyfajta koncentrációját teszi lehetővé. A konszolidáció annál hatékonyabb, minél magasabb szintet érint. Ezért a közös központi tároló kialakítása után logikus lépés az adatbázisok fürtbe (***Real Application Clusters***-be) való szervezése. Ez egy vagy több fizikai adatbázis több szerverről való egyidejű elérhetőségét teszi lehetővé. Az adatbázis fürtözés egyszerre szolgálja az adatbázis folyamatos elérhetőségét és a szerverek párhuzamos működése révén a teljesítmény igényeknek való rugalmas megfelelést. A konszolidáció csökkenti a költségeket, de a korábban különálló fizikai adatbázisok egy adatbázisban való egyesítése biztonsági kockázatot jelent. Csökkentésének egyik lehetséges eszköze a virtuális adatbázisok létrehozása. A korábban azonos szerkezetű több példányban létező adatbázisok egyetlen adatbázisban való összefésülése azt eredményezi, hogy a felhasználók egymás adataihoz hozzáférhetnek. A probléma megoldására több módszer kínálkozik.

Az alkalmazást módosítjuk, és minden *SQL* utasítást kiegészítünk egy szűrőfeltétellel. Ez a védelem, azonban csak ebben az alkalmazásban fog működni, vagyis minden további az adatbázist elérő alkalmazás módosítása szükséges.

Következő, már alkalmazás független megoldás, adatbázis nézetek létrehozása. Vagyis minden felhasználónak, vagy felhasználói csoportnak létre kell hozni egy a szűrőfeltételt tartalmazó nézetet. A táblák átnevezésével és a nézetekre az eredeti táblanévvel megegyező szinonimák létrehozásával az alkalmazás elől elrejthetjük a változtatást. A megoldás hátránya, hogy nagyon nehézkessé válhat a rengeteg nézet adminisztrálása.

Az előző kettőnél elegánsabb és alkalmazás független megoldás az *Oracle* adatbázis kezelő ***Virtual Private Database (VPD)*** technológiájának a használata. A *VPD* segítségével elérhető, hogy minden felhasználó ugyan azt az alkalmazást, és annak beépített hozzáférés szabályozását használva, mégsem látja egymás adatait. A gyakorlatban ez azt jelenti, hogy a felhasználók ugyan azon utasítást kiadva az adatbázis táblák más-más soraihoz férnek hozzá, akár lekérdezésről, akár adat manipulációs (***insert***, ***update***, ***delete***) műveletről legyen szó.

A *VPD*-t emiatt szoktuk sorszintű hozzáférés szabályozást lehetővé tevő eszköznek mondani.

A *VPD* működésének alapja, hogy minden táblához és azon belül minden utasításhoz egy szűrőfeltételt lehet rendelni. Ez utóbbit egy *PL/SQL* nyelven írt tárolt eljárás állítja elő.

Példaprogram

Tegyük fel, hogy egy cég szeretné az adatainak egy részét a partnerei számára hozzáférhetővé tenni. Mindezt úgy, hogy minden partner a csak rá vonatkozó adatokat lássa. A példában két táblát fogunk használni. Mind kettő az RENDELÉSEK sémában található.

Az ÜGYFELEK tábla szerkezete:

```
Név             Üres?    Típus
--------------------- --------
ÜGYFÉL_AZ       NOT NULL NUMBER(6)
ÜGYFÉL_NÉV      VARCHAR2(45)
```

Az ÜGYFELEK tábla tartalma:

```
ÜGYFÉL_AZ ÜGYFÉL_NÉV
--------- -----------------
      102 SCOTT
      101 PALM
```

A RENDELÉSEK tábla szerkezete:

```
Név             Üres?    Típus
--------------------- --------
RENDELÉS_AZ     NOT NULL NUMBER(4)
RENDELÉS_DÁTUM  DATE
ÜGYFÉL_AZ       NOT NULL NUMBER(6)
```

A RENDELÉSEK tábla tartalma:

```
RENDELÉS_AZ RENDELÉS_DÁT
ÜGYFÉL_AZ
 ----------- ------------ -----
         100 07-FEBR. -26   102
         101 07-FEBR. -21   101
```

A hozzáférés szabályozásunknak az a célja, hogy minden ügyfél csak a saját rendelési adataihoz férjen hozzá. Jelen esetben a SCOTT felhasználó a RENDELÉSEK tábla azon sorához fog hozzáférni, amelynek az ÜGYFÉL_AZ mezőjének értéke 102.

A hozzáférés szabályozás tárolt eljárásai a SZABÁLY sémában találhatóak. Az ügyfél-azonosítót a felhasználók adatbázisba való bejelentkezésekor

a LOGON_TRIGGER segítségével rögtön meghatározzuk és egy szerver oldali környezeti változóban tároljuk, amely az egész bejelentkezés ideje alatt elérhető lesz.
Felhasználó ügyfél azonosítójának meghatározása a bejelentkezéskor:

```
CREATE OR REPLACE PACKAGE
➥"SZABÁLY"."RENDELÉSEK_BIZT_
➥CSOMAG" IS
  PROCEDURE "AZONOSÍTÓ";
end;
/

CREATE OR REPLACE PACKAGE BODY
➥"SZABÁLY"."RENDELÉSEK_BIZT_
➥CSOMAG" IS
  PROCEDURE "AZONOSÍTÓ"
IS
  "KÓD" NUMBER;
  BEGIN
    SELECT "ÜGYFÉL_AZ" INTO
    ➥"KÓD" FROM "RENDELÉSEK".
    ➥"ÜGYFELEK"
      WHERE "ÜGYFÉL_NÉV"=
      ➥SYS_CONTEXT('USERENV',
      ➥'SESSION_USER');
    DBMS_SESSION.SET_CONTEXT
    ➥('RENDELÉSEK_BIZT',
    ➥'ÜGYFÉL_KÓD',to_char
    ➥("KÓD"));
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_SESSION.SET_CONTEXT
      ➥('RENDELÉSEK_BIZT',
      ➥'ÜGYFÉL_KÓD','-1');
  END;
END "RENDELÉSEK_BIZT_CSOMAG"
/
```

A SYS_CONTEXT függvény a felhasználói munkafolyamat környezeti változóinak lekérdezését teszi lehetővé. A USERENV az alapértelmezett névtér. A SESSION_USER értéke az adatbázis felhasználó neve. A program tehát az ÜGYFELEK tábla alapján meghatározza az ügyfél azonosítót (ÜGYFÉL_AZ) és a RENDELÉSEK_BIZT névtérbe helyezi ÜGYFÉL_KÓD néven.
Ehhez a csomaghoz minden adatbázis felhasználónak végrehajtási jogot adunk:

```
grant execute on "RENDELÉSEK_
➥BIZT_CSOMAG" to public;
```

Ezután létrehozzuk az adatbázis LOGON_TRIGGER-t:

```
CREATE OR REPLACE TRIGGER
➥LOGON_TRIGGER
  AFTER LOGON
  ON DATABASE
BEGIN
  "SZABÁLY"."RENDELÉSEK_BIZT
  ➥_CSOMAG"."AZONOSÍTÓ";
END;
/
```

Következő lépés a szűrőfeltételt meghatározó *PL/SQL* program létrehozása:

```
CREATE OR REPLACE PACKAGE
➥"SZABÁLY"."ÜGYFÉL_BIZT"  AS
  FUNCTION  "FELTÉTEL"(D1
  ➥VARCHAR2,D2 VARCHAR2)
  ➥RETURN VARCHAR2;
  PRAGMA RESTRICT_REFERENCES
  ➥("FELTÉTEL",WNDS);
END;
/

CREATE OR REPLACE PACKAGE BODY
➥"SZABÁLY"."ÜGYFÉL_BIZT"  AS
  FUNCTION  "FELTÉTEL"(D1
  ➥VARCHAR2,D2 VARCHAR2)
  ➥RETURN VARCHAR2
  IS
    D VARCHAR2(2000);
  BEGIN
    IF RTRIM(USER)<>D1 AND
    ➥RTRIM(USER)<>'SYS' AND
    ➥RTRIM(USER)<>'SYSTEM' THEN
      D:='"ÜGYFÉL_AZ"=
      ➥SYS_CONTEXT
      ➥(''RENDELÉSEK_BIZT'',
      ➥''ÜGYFÉL_KÓD'')';
    ELSE
      D:='1=1';
    END IF;
    RETURN D;
  END "FELTÉTEL";
END;
/
```

A PRAGMA RESTRICT_REFERENCES direktívával jelezzük, hogy ez a tárolt eljárás nem fogja módosítani az adatbázist, tehát lekérdezésben kívánjuk használni. A rendszer két paraméterrel (a séma nevével, amelyben a tábla van, és a tábla nevével) fogja a FELTÉTEL függvényt meghívni. A USER változó tartalma a felhasználó neve. A FELTÉTEL függvény visszatérési értékével fog az *SQL* utasítás WHERE része kiegészülni. Látható, hogy a logikailag mindig igaz, 1=1 értéket adja vissza, ha az utasítást kiadó felhasználó megegyezik a tábla tulajdonossal, a SYS, vagy a SYSTEM felhasználóval.
A szabály elkészítése után már csak meg kell határoznunk, hogy mely táblára és milyen adatbázis művelet esetén kívánjuk alkalmazni:

```
BEGIN DBMS_RLS.ADD_POLICY
➥('RENDELÉSEK','RENDELÉSEK',
➥'RENDELÉS_SZABÁLY','SZABÁLY',
➥'"ÜGYFÉL_BIZT"."FELTÉTEL"',
➥'SELECT');
END;
/
```

A RENDELÉSEK séma RENDELÉSEK táblájára létrehozzuk a RENDELÉS_SZABÁLY-t, amelynek érvényesítését a SZABÁLY séma ÜGYFÉL_BIZT csomagjának FELTÉTEL nevű függvénye fogja elvégezni, minden SELECT művelet esetében.

A VPD működése

A SCOTT felhasználó bejelentkezése során az adatbázis LOGON_TRIGGER meghívja az AZONOSÍTÓ tárolt eljárást, amely az ÜGYFELEK tábla alapján meghatározza az ügyfél azonosítóját (102) és a RENDELÉSEK_BIZT névtér ÜGYFÉL_KÓD változójába tárolja. Ezután ha SCOTT lekérdezi a RENDELÉSEK táblát, akkor az FELTÉTEL függvény visszatérési értéke az "ÜGYFÉL_AZ"=102 lesz. Az eredeti

```
"select * from
➥RENDELÉSEK.RENDELÉSEK"

lekérdezés tehát a

"select * from
➥RENDELÉSEK.RENDELÉSEK where
➥ÜGYFÉL_AZ=102"
```

utasítássá alakul át. Az eredmény pedig ez lesz:

```
RENDELÉS_AZ RENDELÉS_DÁT
➥ÜGYFÉL_AZ
----------- ------------ ------
        100 07-FEBR. -26    102
```

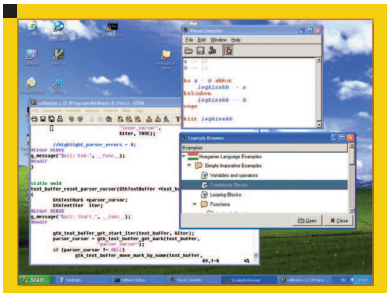
Összefoglalva a bemutatott példa jól szemlélteti, hogy hogyan lehet az *Oracle VPD* technológiájával hatékonnyá tenni a konszolidációt a biztonsági kockázatok kezelése mellett.

Mosolygó Ferenc

Alkalmazásfejlesztés a GTK+ programkönyvtárral (1. rész)

Napjainkban a legtöbb felhasználó olyan alkalmazásokat keres, amik grafikus felhasználói felülettel (GUI) is fel vannak szerelve. Éppen ezért a programozással hobbiból vagy hivatásszerűen foglalkozók körében igen keresettek a grafikus eszköztárak. Cikksorozatunkban a GTK+ programkönyvtár segítségével végzett alkalmazásfejlesztés fontosabb elemeit mutatjuk be azon olvasóink számára, akik a programozás alapjaival tisztában vannak és szeretnének modern, felhasználóbarát grafikus alkalmazásokat készíteni.

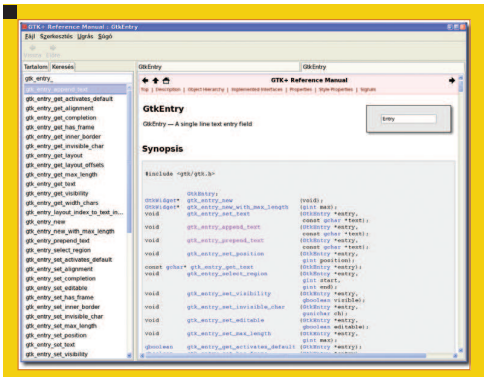
A *GTK+* (*Gimp Toolkit*) programkönyvtár – ahogyan neve is mutatja – grafikus eszköztárat biztosít a programozó számára *GUI* alkalmazások készítésére. Persze több programkönyvtár is létezik hasonló szolgáltatásokkal, amelyek közül érdemes megfontoltan választanunk. A programkönyvtárak napjainkban olyan sokrétűek, nagyok és bonyolultak, hogy az alapos megismerésükhöz évek kemény munkájára van szükség, ezért nem lehet túlbecsülni a választás jelentőségét. Mik tehát a *GTK+* programkönyvtár előnyei?
A programkönyvtár teljes egészében C programozási nyelven íródott, ezért hatékony, gyors és viszonylag kis memóriaigényű a többi, hasonló tudású eszköztárral összevetve. Ennek köszönhető, hogy a segítségével készített alkalmazások gördülékenyen futnak, nem akadoznak és nem idegesítik fel feleslegesen a felhasználókat. Abban, hogy a felhasználók jelentős része gyorsabbnak tartja a *Linuxot* a *Windows* különféle változatainál, valószínűleg komoly szerepe van, hogy a *Windows* grafikus eszköztára sokszor hosszú másodpercekre érzéketlenné válik a felhasználó ingereire, különösen ha sok adatot megjelenítő lista, ikonmező van a képernyőn.
A *GTK+ C nyelven* íródott, ennek ellenére támogatja a *C++*, *Objective-C*, *Guile/Scheme*, *Perl*, *Python*, *TOM*, *Ada95*, *Free Pascal*, és *Eiffel* nyelvű alkalmazásfejlesztést is, ráadásul objektumorientált programozási módszertan (*OOP*) alapján készült, ami a *GUI* készítésére különösen alkalmas. Az olvasók közül bizonyára sokan felkapják a fejüket e kijelentésre, hiszen nem szokás a C nyelvet az *OOP* módszertannal egy lapon emlegetni. Valóban, a *C nyelv* nem támogatja az *OOP* módszereket, viszont nem is teszi lehetetlenné. A *GTK+* alapjait biztosító *G* programkönyvtár teljes értékű keretrendszert ad az osztályok, az öröklődés és a többalakúság megvalósítására, a *GTK+* pedig ki is használja ezeket a lehetőségeket, így az objektumorientáltság ez esetben nem üres frázis, hanem következetesen végigvitt tervezési alapelv.
A *GTK+* másik nagy erőssége az, hogy többféle operációs rendszeren is futtatható. Ennek köszönhető az, hogy a *GTK+* és *G* programkönyvtárakkal készített alkalmazások nem csak *Linuxot* futtató számítógépeken, de például a *Windows* különféle változatain, a *Solaris* rendszereken és *MacOS* felett is használhatók.
Aki tehát *GTK+* segítségével fejleszt *Windows* programot is tud írni (*1. ábra*). Ne becsüljük alá a grafikus alkalmazások hordozhatóságát! Sokan próbáltak már olyan programfejlesztő környezetet létrehozni, ami *GUI* programok fejlesztésére alkalmas és több operációs rendszert is támogat, használható megoldást azonban annál kevesebbeknek sikerült készíteniük.



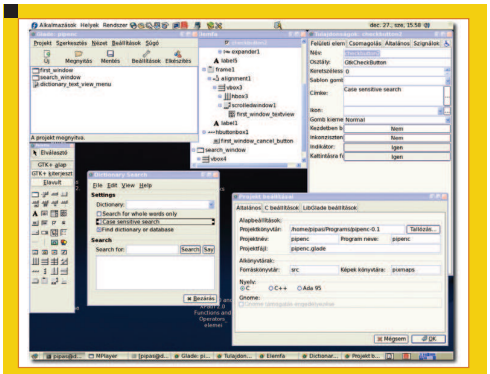
1. ábra GTK+ segítségével készített alkalmazások Microsoft Windows XP felett

A *GTK+* programkönyvtár legnagyobb előnye azonban az egyszerű és logikus felépítése, ami lehetővé teszi, hogy viszonylag rövid idő alatt elsajátítsuk a használatát. A programkönyvtárat profi fejlesztők írják és szabad szoftverként sokan tökéletesítik, akik következetesen betartják a fejlesztés irányelveit, ezért nem nehéz megtanulni a használatát. Nyilván sokan megütköznek e kijelentés hallatán. Egyszerűen megtanulható programkönyvtár *GUI* programok fejlesztésére? Több ezer függvénnyel? C nyelven megírt objektumhierarchiával?

Osztályok százaival? Az egyszerűség nyilván viszonylagos, a *GTK+* a programozó számára könnyebben kiismerhető, mint a hasonló tudású eszköztárak, ezért nyugodtan mondhatjuk, hogy egyszerű. A *GTK+* ráadásul szabad szoftver, ami lehetővé teszi, hogy elolvassuk, sőt akár módosítsuk is a forrását. Ha egy átlagos alkalmazás esetében ennek nem is tulajdonítunk nagy a jelentőséget – mondván, hogy az átlagos alkalmazást átlagos felhasználók használják – a programkönyvtárak esetében azonban különösen fontos a nyílt forrás. A programkönyvtárat nyilván programozók használják, akik szívesen vetnek néha egy-egy pillantást a forrásra, hiszen a legjobb dokumentáció maga a program.



2. ábra A devhelp gyorskereső szolgáltatása



3. ábra A glade ablakai

Az első lépések

A programozáshoz a programkönyvtárakon és a fejállományokon kívül okvetlenül szükségünk lesz a programkönyvtárak dokumentációjára, amelyet legkönnyebben a `devhelp` program segítségével böngészhetünk. Maga a `devhelp` elég primitív program, már néhány perc használat után is komoly késztetést érzünk arra, hogy megnyissuk a forrását és egy kicsit továbbfejlesszük, gyorskereső szolgáltatása azonban gyakorlatilag nélkülözhetetlenné teszi. A program ablakának bal oldalán található beviteli mezőbe (*2. ábra*) a keresett függvény vagy adatszerkezet nevének részletét is beírhatjuk, a találatok listájából pedig kiválaszthatjuk, hogy melyik találat részletes dokumentációjára vagyunk kíváncsiak. Ezt a lehetőséget a kezdő programozó igen intenzíven használja az első néhány hónapban, de a későbbiekben is hasznos lehet hiszen sokszor nem emlékezünk a szükséges függvény teljes és pontos nevére.

A munka során legtöbbet használt eszköze azonban a *Glade* lesz, a *GTK+* svájci bicskája, aminek használatát mindenképpen meg kell ismernünk. E programmal az alkalmazásunk grafikus felhasználói felületét grafikus felhasználói felületen szerkeszthetjük, ami meggyorsítja és megkönnyíti a *GUI* létrehozását és a későbbi módosítást is lehetővé teszi. A különféle képernyőelemek a *GTK+* megfelelő függvényeinek hívásával előállíthatók és a képernyőn megjeleníthetők, az ilyen program írása és karbantartása azonban igen unalmas és lassú, érdemes a *Gladedel* automatizálni.

A *Glade* programból jelenleg két – egymástól jelentősen különböző – változat is elérhető. A régebbi a `glade-2` paranccsal indítható (*3. ábra*). Ez a változat képes elmenteni a szerkesztett felhasználói felületet *XML* formátumban és képes az `autoconf` és az `automake` programokkal támogatott projektet létrehozni, fel van szerelve egy C nyelvű kódgenerátorral. Az újabb – `glade-3` paranccsal indítható – változatban nincsen kódgenerátor, csak az *XML* állomány létrehozására használható. Az első lépések megtételéhez a `glade-2` alkalmasabb, mert igen könnyen készíthetünk teljes értékű alkalmazást a segítségével, nem kell a projekt beállítóállományaival és az *XML* állománnyal bíbelődnünk. Valójában a `glade-2` olyan egyszerű, hogy néhány sorban leírható hogyan készíthetünk alkalmazást a segítségével.

A `glade-2` indítása után a menü segítségével hozzunk létre és állítsunk be egy új projektet. Adjuk meg a projekt nevét és a projektkönyvtárat, ahová az állományok kerülni fognak. A többi beállítással az első kísérletek során nem is kell foglalkozni.

Második lépésként a *paletta* ablakban válasszuk a legelső ikont, hogy egy új ablakot hozzunk létre. Az ablakba – szintén a paletta segítségével – ízlés szerint helyezzünk el vízszintes és függőleges dobozokat, azokba pedig szintén tetszés szerint válasszunk a paletta különféle képernyőelemei közül. Amikor úgy gondoljuk, hogy itt az ideje a programunk kipróbálásának, a *Glade* főablakának segítségével mentsük a projektet, majd az *elkészítés* gomb segítségével hozzuk létre a projekt automatikusan készített állományait.

A következő lépésként a *Glade* által létrehozott és a projekt főkönyvtárában elhelyezett `autogen.sh` héjprogramot kell futtatnunk. Ez a program az `automake` és `autoconf` programcsomagok segítségével létrehoz néhány állományt, majd szívességből futtatja az ugyanitt elhelyezett `configure` programot is.

Ezek után az alkalmazásunkat a szokásos módon, a `make` futtatásával lefordíthatjuk, a `make install` paranccsal pedig telepíthetjük. A forrásprogram terjesztésre alkalmas csomagolt változatát szintén a megszokott módon, a `make dist` parancs kiadásával hozhatjuk létre.

A *Glade* a projekt alapkönyvtárában létrehoz egy `.glade` névvégződésű *XML* formátumú állományt a felhasználói felület leírásával. Ezt az állományt a későbbiekben megnyithatjuk a *Glade* segítségével, módosíthatjuk, a forrásprogramot pedig a *létrehoz* gombbal frissíthetjük. A *Glade* az automatikusan módosított függvényeket és a programozó által készített programsorokat külön állományokban tárolja, így egy kis figyelemmel elkerülhetjük, hogy a munkánk kárba vesszen.

Pere László
(pipas@gamma.ttk.pte.hu)

gEvas: a GTK+2 és az Evas összekapcsolása

Ismerjük meg az Enlightenment néhány alapkönyvtárát és azok alkalmazását gEvas-szel, nagy teljesítményű GTK+2 alkalmazások készítéséhez.

Az ***Evas*** olyan függvénykönyvtár, amellyel a rasztergrafikák gyors megjelenítése megvalósítható, az áttetszőség támogatásával. Az ***Evas*** az ***Enlightenment*** alapkönyvtárainak (***Enlightenment Foundation Libraries – EFL***) egyike, valójában azonban függvénykönyvtárak gyűjteménye, melyeket eredetileg az ***Enlightenment DR 17*** támogatására készítettek.
Az ***Edje*** és az ***Embryo*** kombinációja, az ***Emotion*** további olyan ***EFL***-könyvtárak, amelyek az ***Evas***-t kiegészítik. Az ***Edje***-vel betűkészletek, grafikák és funkciók csomagolhatók témafájlokba. Az ***Embryo*** egy egyszerű, ugyanakkor ***Turing***-teljes szkriptnyelv, amivel egyszerű parancsokat ágyazhatunk az ***Edje***-fájlokba. Az ***Emotionnel*** egyidejűleg több videófolyamot használhatunk elsőosztályú keretobjektumként, ami annyit jelent, hogy mozgathatjuk, átméretezhetjük, rétegezhetjük vagy akár áttetszővé tehetjük ezeket.
A ***gEvas*** könyvtár lehetővé teszi, hogy az ***Evast*** felhasználhassuk ***GTK+2.x*** alkalmazásokban. Kedvcsinálónak elmesélem, hogy engem leginkább az egyszerű ***API*** és az áttetszőséggel biztosított gyors képalkotás vezetett az ***Evas*** használatáig, és végül a ***gEvas*** megírásáig.
Ahhoz, hogy a képalkotás sebességéről ilyen kijelentéseket tehessünk, egy kis kirándulást kell tennünk a benchmarking világába. Az ***Evas*** disztribúcióban található ***evas_bench*** alkalmazás sok kép- és szövegelemet tartalmaz, azok átméretezésének és elegyítésének a lehetőségével. A képátméretezés eredménye pl. annyira kiváló, hogy párja sem akad. Átírtam az ***evas_bench***-et, hogy ***GNOME Canvast*** használjon.

Az *1. ábrán* látható a ***GNOME Canvasra*** átírt ***evas_bench*** képernyőképe. Készítettem néhány egyszerűbb tesztet is a keretátméretezés algoritmusához, áttetsző és nem áttetsző keretekkel egyaránt. Utóbbi esetben egy levél képét méretezzük át ciklusban a keretnél nagyobb méretről 0x0-ra, majd vissza, a *2. ábrán* látható módon. Az áttetszőségnél egy, a levél képének méretével megegyező piros téglalapot használtam, ahol az alfa értéke 0 volt a bal felső sarokban és maximális az alsón. Ahogy már említettem, a ***gEvast*** jómagam írtam, és néhány olvasónak talán az is feltűnik, hogy az ***Enlightenment*** projekt fejlesztői csapatának tagja vagyok. Bár mindez igaz, mindent elkövettem annak érdekében, hogy a benchmark elfogulatlan eredményt adjon. A benchmark forráskódja letölthető (lásd az online anyagokat). Azoktól, akik belenéznek, előre is elnézést kérek a sebtében végzett kódolás közben gyakran alkalmazott, kevéssé optimális megoldások miatt.
Azok kedvéért, akik a ***GNOME Canvast*** kevéssé ismerik, elmondom, hogy két képalkotó back-endje létezik. A ***GNOME Canvas*** fejlesztői dokumentációja szerint (lásd a kapcsolódó anyagokat): A ***GNOME Canvas***-ben két képalkotó back-end közül lehet választani. Az egyik ***Xlib*** alapú és rendkívül gyors megjelenítést biztosít, a másik ***Libart*** alapú, kifinomult, élfinomítást és áttetszőséget támogató motorral. Az ***Evas*** megpróbálja a két világ előnyeit egyszerre nyújtani. Mindkettővel összehasonlítottam az ***Evast***.

1. ábra GNOME Canvas-ra átírt evas_bench alkalmazás. A keretben sok kép- és szövegelem mozog és átméreteződik.

Egyes esetekben természetesen az ***Evas*** is lehet rossz választás.
A ***GNOME Canvas***-ben találunk egy ***Bezier***-görbéket támogató elemet is, míg az ***Evas***-ban ilyen pillanatnyilag nem létezik. Ráadásul az is kevésbé valószínű, hogy az ***Evast*** és a ***gEvast*** előtelepítve találjuk, mint a ***GNOME Canvas*** esetében.
A ***Bezier***-görbék támogatása a jövőben megvalósulhat az ***Evasban*** is. Ezzel kapcsolatban ***Raster*** (becses nevén ***Carsten Haitzler***) a következő sommás

véleményt fogalmazta meg: *„Ha valaki vektorgrafikus szerkesztőt vagy valami hasonlót keres, használja a GNOME Canvast, mivel az abban jobb. Ha viszont bárhol, áttetszőséget is támogató, valósidejű megjelenítésre van szükség, az Evas a nyerő.”*
Az *Evas* önmagában is számos képalkotó back-endet támogat, köztük a *framebuffert*, az *XLib*-et és az *OpenGL*-t. A *gEvas* ezek közül egyelőre csak az *XLib*-et használja. Mivel a *GTK+2 framebufferen* is fut, a *gEvas*-nak is kellene, ezt azonban eddig még nem próbáltam.
A *GNOME Canvas* és az *Evas* adatmodellje hasonló. A *Qt QCanvas* adatmodellje azonban már eléggé különbözik ahhoz, hogy megnehezítse a benchmarkot. Az egyik legmarkánsabb különbség, ahogy a *QCanvas* a képeket kezeli. Úgy tehetünk képet a keretre, hogy létrehozunk egy *QCanvasSprite*-ot, egyetlen kerettel. A képet eztán a `QImage::scale()` és a `QImage::smoothScale()` metódusok egyikével méretezhetjük át, aminek eredményeképp egy képet kapunk, és ezzel frissíthetjük a `QCanvasSprite` tartalmát. A méretezés és a kapcsolódó gyorsítótár kezelése ezzel az ügyfélalkalmazásba kerül. Ezzel szemben mind a *GNOME Canvas*, mind az *Evas* közvetlenül támogatja a méretezést, és felelős az átméretezett képek gyorsítótárának kezeléséért is.
A másik fontos különbség, hogy a *Qt* lehetőséget ad a frissítendő terület méretének meghatározására. A *Qt* dokumentációja szerint: *„Ökölszabály szerint válasszunk az átlagos keretméretnél, mozgó objektumok esetén viszont csak azok átlagos méreténél valamivel kisebbet.”*

2. ábra Az evas_bench képátméretezési és -elrendezési benchmarktesztjének gEvas-os változata. A levelet fokozatosan kicsinyíti, amíg eléri a keret bal felső sarkát, majd nagyítja az eredeti méretre.

Az adatmodellek különbözősége miatt az *evas_bench*-nek még nem készítettem el a *Qt*-re írt változatát. Írtam viszont egy keretméretező és összhatást változtató programot, de néhány tervezési különbség miatt nem lehetséges a teljesen egyértelmű összehasonlítás.
Minthogy a *Qt* teljesen az ügyfélprogramba delegálja a gyorsítótár kezelését, úgy döntöttem, hogy az első iterációban az összes átméretezett képet a gyorsítótárba helyezem, és a benchmarkot alaphelyzetbe hozom a későbbi iterációkhoz, amelyek már csak a gyorsítótárban lévő képeket használják.
Ne felejtsük el tehát, hogy a *Qt* méretezési benchmarkja úgy készült, hogy az összes átméretezett kép előre a gyorsítótárba került, és a frissítendő területhez több méretet is használtunk. Emiatt a *QCanvas* eleve előnnyel indult a versenyben az *Evashoz* és a *GNOME Canvashoz* képest.

Induljon a verseny

A fajlagos eredmények a hardvertől függetlenül hasonlóak kellenek, hogy legyenek.
A teljesség kedvéért az általam használt gép paraméterei: *AMD XP-Mobile* 2.4 GHz, 200 MHz *FSB*, 1GB RAM 400 MHz kétcsatornás *cas222*-n és *NVIDIA 5900* videokártya, és a programok, amik a teljesítményt befolyásolhatták: *xorg-x11-6.8.2-1.FC3.13*, *GCC 4.0.0 20050308* (*Red Hat 4.0.0-0.32*) vagy *GCC 3.4.3.* Az *X11*-et *TwinView*-val használom, az egyik képernyő 1024x768, a másik 1600x1200 felbontású, mindkettő 85 Hz-es frissítéssel és 32 bites színmélységgel. A *TwinView* valószínűleg nem befolyásolja a mérést, mivel a keretek megjelenítése olyan szoftverágakon zajlik, amelyek sokkal érzékenyebbek a *CPU*/*RAM* sebességre.
A benchmarkok fordításához általában ezeket a `CFLAGS` és `CXXFLAGS` kapcsolókat használják:

```
-O3 -march=athlon-xp -fomit-
➥frame-pointer
```

Az *Evas* 2005. május 28-án *CVS*-ből letöltött verzióját ezért szintén ezekkel a `CFLAGS` kapcsolókkal fordítottam,

1. táblázat *A qt-canvas-resize program benchmarkja különféle területfrissítési értékekkel*

Alkalmazás	Területfrissítés mértéke	Levél képsebessége	Téglalap képsebessége
`qt-canvas-resize`	default	114	72
`qt-canvas-resize`	32	128	80
`qt-canvas-resize`	64	136	82
`qt-canvas-resize`	128	142	81

csakúgy mint a ***qt-3.3.4*** és ***libgnomecanvas-2.10.0*** könyvtárakat. A ***qt-canvas-resize*** program benchmarkját külön végeztem el az eltérő gyorsítótáras megoldás és a korábban említett területfrissítési optimalizáció miatt. Az eredmények az *1. táblázatban* láthatók. A fő ciklus ***Qt*** része lényegében a következő:

```
QCanvasSprite* leaf_sprite =
➥...;
QCanvasPixmapArray* leaf_tiles =
➥...;
while( running )
{
  while( app->has
  ➥PendingEvents() )
    app->processEvents();
  QImage im = ... from cache
  ➥...;
  QCanvasPixmap* qpix = new
  ➥QCanvasPixmap( im );
  leaf_tiles->setImage( 0,
  ➥qpix );
  leaf_sprite->setFrame(0);
  canvas->update();
}
```

A program működését számos parancssori kapcsoló befolyásolja: az `--alpha-blend-image` kapcsolóval adhatunk áttetszőséget a piros téglalapnak a levél helyett, a `--chunk-size` kapcsolóval bírálhatjuk felül a területfrissítés alapértékét. Az `--alpha-blend-image` kapcsolót a `qt-canvas-resize`, a `gnome-canvas-resize` és a `(g)evas-resize` programok mindegyike használhatja.
A ***Valgrind*** `callgrind`-jével néhány percig futtatva a `qt-canvas-resize` programot, az alapértelmezett területfrissítést használva, a levél esetében a `QCanvas::update()` a teljes futásidő 30 %-át emészti fel, míg 59 %-ért a `QCanvasPixmap::init()` metódus felel. Ezen feltétlenül lehet javítani, ha a gyorsítótárba kerülő képeket a `QCanvasSprite` számára `QCanvasPixmapArray`-ben tároljuk. Az előtárazásnak ezt a szintjét a `-Z` kapcsoló hozzáadásával teszteltem, aminek hatására a gyorsítótárban lévő összes kép egy `QCanvasPixmapArray`-be kerül a `QCanvasSprite` támogatására. Ezzel az optimalizációval 559-es képsebesség valósítható meg, miközben a futásidő 78 %-át viszi el a `QCanvas::update()` és 7 %-át a `QCanvasSprite::setFrame()`.

2. táblázat *A GNOME Canvas és a (g)Evas képátméretezési benchmarkja*

Alkalmazás	Levél képsebessége	Téglalap képsebessége
`gnome-canvas-resize`	21	21
`gnome-canvas-resize --aa`	149	127
`evas-resize`	190	184
`gevas-resize`	185	177

3. táblázat *Az evas_bench és a GNOME Canvas-ra átírt változatának viadala*

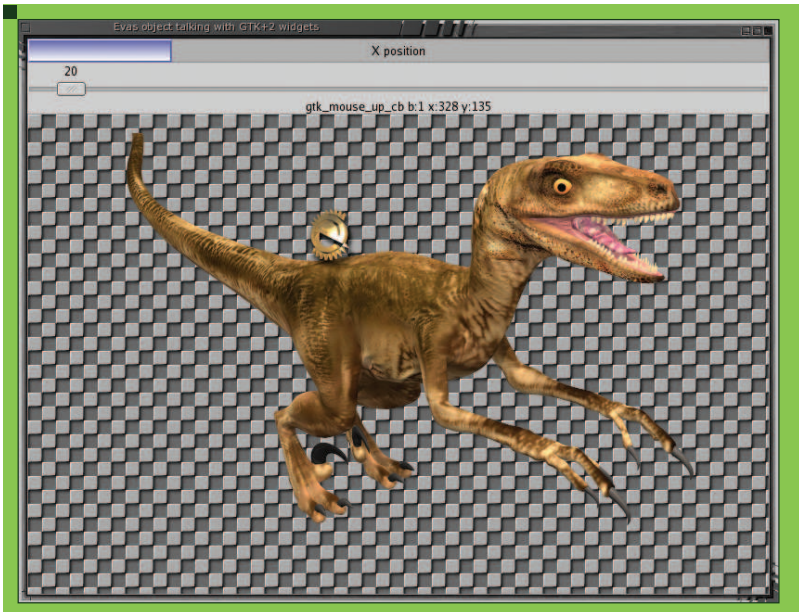
Alkalmazás	Képsebesség	EVAS_BENCH
`gnome-canvas-port-evas-bench --aa`	90	1.49
`evas_software_x11_main`	164	2.75
`evas_software_x11_main --smooth-off-for-some`	200	3.32
`evas_buffer_test`	290	4.83

Meg kell jegyezni, hogy az előtárazásnak ez a mértéke méltánytalan előnyhöz juttatja a ***QCanvast*** a képalkotás sebességében.
A képméretezés és összhatás ***GNOME Canvas*** programja a `gnome-canvas-resize`, melynek `--a` kapcsolójával választható ki a ***GNOME Canvas*** áttetszőséget megvalósító back-endje. Az `evas-resize` programnak nincs parancssori kapcsolója.
A `--a` kapcsoló hiányában a `gnome-canvas-resize` az idő 99 %-át a `gtk_widget_send_expose()` függvényben tölti, amelyet így vagy úgy a `g_main_context_iteration()` függvényből hívunk. Úgy tűnik, hogy a másik ***GTK+2*** motor számára nem előnyös a benchmark diktálta igénybevétel. Ha `callgrind`-del megnézzük a `--aa` ***GNOME Canvas*** back-endet, azt látjuk, hogy az idő 96 %-át a `gtk_widget_send_expose()`-ban tölti, ugyanakkor most az idő 66%-a telik el a `gdk_pixbuf_composite()`-ban, amelyet indirekt módon hívunk a `gtk_widget_send_expose()`-ból. Az `evas-resize` az idő 99 %-át az `evas_render_updates()`-ben tölti, és az ebből hívott függvények az idő 91 %-ában a méretezéssel foglalkoznak. Az `evas-resize` programot úgy is átírtam, hogy a ***gEvas API***-ját használja. Némi sebességcsökkenés észlelhető a ***GTK+2*** szignálok és a ***gEvas*** körítése miatt, de ez nem jelentős.

Az összehasonlítás szerint a közvetlen képátméretezésben a ***GNOME Canvas*** és a ***QCanvas*** hasonló eredményeket adnak, és mindkettő lassabb az ***Evas***-nál, amely ráadásul további előnyhöz jut, ha a képet áttetszővé kell tenni a háttér előtt.
Az `evas_bench` programból eltávolítottam azokat a funkciókat, amelyeket nehéz lett volna megvalósítani a ***GNOME Canvas***-ben. A többi funkció teljesítményre gyakorolt hatásának mérése opcionális.
A vágóterületek beállítása ***Evasból*** nehezen vihető át a ***GNOME Canvas***-be, így ezt az `evas_bench` programban nem engedélyeztem. Az egyenletes átméretezés nagy mértékben rontja a teljesítményt, így az ***Evas***-változathoz egy parancssori kapcsolót adtam, amellyel ez kikapcsolható.
Hadd jegyezzem meg, hogy az ***Evas*** pillanatnyilag nem rendelkezik gyorsítótárral az átméretezett képek tárolására, így az ***Evas*** benchmarkokban minden egyes keret végrehajtja az átméretezést és az áttetszőség biztosítását.
Az `evas_buffer_test` program is ezt teszi, de a képet egy, a memóriában lévő, 32 bites RGBA tárolóban állítja elő.

gEvas

A ***gEvas*** magja lényegében 5 dologból áll: megmondja az ***Evas***-nak, mikor rajzolja újra magát, segédkezik az ***Evas*** eseményeinek és a ***GTK+2***



3. ábra Az Evas és GTK+2 szignálok összekapcsolása. A dino közvetlenül vagy a csúszkák segítségével is mozgatható.

szignálok összerendelésében, kezeli az ***Edje*** időzítőit az animációk támogatásához, segít abban, hogy az ***Evas*** jól kezelje a ***GTK+2*** widgeteket, továbbá kódot nyújt az ***Evas*** helyes használatához. Mivel az ***Evas*** úgy készült, hogy beágyazott rendszerekhez is megfelelő legyen, az ***Evas*** magjának egy része kimaradt, hogy a kódméret kicsi maradjon. Mivel a ***gEvas*** desktop-orientált, néhány hasznos függvényt is tartalmaz a desktop-alkalmazásokhoz.
Az alábbi kód egy gördíthető területen hoz létre ***gEvas*** keretet, majd egy ***GTK+2*** ablakhoz csatolja. Minthogy nem minden gördíthető ***gEvas***, így például a ***GIMP*** sem, engedi a középső egérgombbal megváltoztatni a keret helyzetét, ezt a `gevas_new_gtkscrolledwindow()` függvényen kívül kell beállítani:

```
GtkWidget* window = 0;
GtkWidget* scw    = 0;
GtkWidget* gevas  = 0;
window = gtk_window_new
➥(GTK_WINDOW_TOPLEVEL);
gevas_new_gtkscrolledwindow(
   (GtkgEvas**)(&gevas), &scw );
gtk_container_add(GTK_CONTAINER
➥(window), scw);
gtk_scrolled_window_set_policy(
   GTK_SCROLLED_WINDOW(scw),
   GTK_POLICY_AUTOMATIC,
   ➥GTK_POLICY_AUTOMATIC);
gevas_set_middleb_scrolls(GTK_G
EVAS(gevas), 1,
  gtk_scrolled_window_get_
  ➥hadjustment(
     GTK_SCROLLED_WINDOW(scw)),
  gtk_scrolled_window_get_
  ➥vadjustment(
     GTK_SCROLLED_WINDOW
     ➥(scw)));
```

Az objektumok létrehozása ***gEvas***-ben és a ***GTK+***-ban egy tőről fakad, néhány metódusnak a `GtkgEvasObj` osztályhoz kapcsolásával. Ebből származik a többi speciális osztály, például a `GtkgEvasImage`. Sajnos, ez maga után vonja a típuskényszerítéssel terhelt ***ANSI C GTK+*** kódolást.
Az alábbi kód egy képet készít, amely eredeti méretben mutatja egy ***PNG***-fájl tartalmát. Ezután a képet mozgatjuk, és a keretben lévő rétegét is megváltoztatjuk:

```
GtkgEvasImage* gi;
GtkgEvasObj* go;
gi = gevasimage_new_from_
➥metadata(
        GTK_GEVAS(gevas), "/my/
        ➥path/foo.png" );
go = GTK_GEVASOBJ( gi );
int x = 100, y = 50;
gevasobj_move(      go, x, y );
gevasobj_set_layer( go, 1 );
```

Készítettem egy egyszerű programot, amely bemutatja, hogyan kapcsoljuk az ***Evas*** eseményeit egy ***GTK+2*** widgethez a ***gEvas*** widgeten kívül. A ***signalconnect.c*** a ***gEvas*** csomag ***demo*** könyvtárában található. Bonyolultabb példákkal, köztük. ***glib*** szignálokra irányított, egyszerű ***Evas*** és ***Evas***-vezérelt függvényvisszahívásokkal szolgál a testgevas program. A szignálkapcsolást a *3. ábra* mutatja be.
Az ***Evas*** eseményeit a `GtkgEvasEvHClass` alosztály segítségével kapcsoljuk. Az alábbi kódrészletben az evh az egér fel-le mozgásának ***Evas***-eseményeit ***glib***-szignálokra irányítja, azokat pedig eseménykezelő függvényekhez kapcsoljuk. Ha a felhasználó mozgatja a ***raptort***, a `raptor_moved()` függvény hívódik egy ***glib2***-szignál hatására, hogy különféle ***GTK+2*** widgeteket frissítsen a kép aktuális koordinátáival:

```
static gint raptor_moved(
    GtkgEvasObj* o,
    Evas_Coord* x, Evas_Coord*
    ➥y,
        gpointer user_data )
{
 gtk_progress_bar_set_fraction
 ➥( x_coord_tracker,
      (1.0 * (*x)) /
      ➥CANVAS_WIDTH );
 gtk_range_set_value(
      GTK_RANGE
      ➥(y_coord_tracker), *y );

 return GEVASOBJ_SIG_OK;
}
static gboolean
gtk_mouse_down_cb(GtkObject *
➥object,
 GtkObject * gevasobj, gint _b,
 ➥gint _x, gint _y,
 gpointer data)
{
 char buffer[1024];
 snprintf(buffer,1000,
 ➥"mouse_down b:%d x:%d y:%d",
          _b, _x, _y);
 gtk_label_set_text( e_logo
 ➥_label, buffer );
 return FALSE;
}
...
gi = gevasimage_new();
go = GTK_GEVASOBJ( gi );
gevasimage_set_image_name( gi,
➥"raptor.png" );
```

```
...
/** Let the user drag the
➥raptor around **/
GtkObject *evh = gevasevh_
➥drag_new();
gevasobj_add_evhandler( GTK_
➥GEVASOBJ( gi ), evh );
gtk_signal_connect( go,
➥"move_absolute",
   GTK_SIGNAL_FUNC( raptor_
   ➥moved ), go );
gi = gevasimage_new();
go = GTK_GEVASOBJ( gi );
gevasimage_set_image_name( gi,
➥"e_logo.png" );
...
evh = gevasevh_to_gtk_
➥signal_new();
gevasobj_add_evhandler( GTK_
➥GEVASOBJ( gi ), evh );
gtk_signal_connect(GTK_OBJECT
➥(evh), "mouse_down",
  GTK_SIGNAL_FUNC
  ➥(gtk_mouse_down_cb), NULL);
gtk_signal_connect(GTK_OBJECT
➥(evh), "mouse_up",
  GTK_SIGNAL_FUNC
➥(gtk_mouse_up_cb), NULL);
```

Íme néhány hasznosabbnak tűnő eseménykezelő, amelyeket szintén csatolhatunk:

```
/* Standard GTK+ popup menu
➥creation + handling */
static gboolean
gtk_popup_activate_cb(GtkObject
➥* object,
   GtkObject * gevasobj, gint
   ➥_b, gint _x, gint _y,
   gpointer data)
{
 static GtkMenu *menu = 0;
 ...
}
GtkgEvasObj* go  = ...;
GtkObject*   evh = 0;
/* Make the object throb when
➥mouse is over it */
GtkgEvasEvHThrob* evht =
➥gevasevh_throb_new( go );
/* Allow the user to drag the
➥object around  */
evh = gevasevh_drag_new();
gevasobj_add_evhandler( go,
➥evh );
/* Make a popup menu appear on
➥right mouse click */
evh = gevasevh_popup_new();
gevasobj_add_evhandler( go,
➥evh );
gtk_signal_connect(GTK_OBJECT
➥(evh),"popup_activate",
   GTK_SIGNAL_FUNC(gtk_popup_
   ➥activate_cb), NULL);
```

A keretben lévő kijelölés kezelése az előző eseménykezelőknél kicsit trükkösebb, mivel a kijelölésben több objektum vesz részt. Létre kell hoznunk a `GtkgEvasEvHGroupSelector` osztályból egy objektumot, amely a nem kiválasztható háttérobjektumhoz kapcsolódik. Úgy is gondolhatunk erre az objektumra, mint ahová a kijelölő téglalapot húzzuk, hogy megmutassuk, mely objektumokat szeretnénk kiválasztani. A kijelölő téglalapnak mindig a kiválasztandó objektumok feletti rétegben kell elhelyezkednie. Így a keretben mindegyik kiválasztható objektumhoz egy `GtkgEvasEvHSelectable` objektum csatlakozik, amely a `GtkgEvasEvHGroupSelector` objektummal kommunikál:

```
GtkWidget*     gevas = ...;
GtkObject*     evh_selector =
➥0;
GtkgEvasImage* gevas_image;
gevas_image = gevasimage_new();
gevasobj_set_gevas(gevas_image,
➥gevas);
gevasimage_set_image_name(gevas
➥_image,".../bg.png");
/* Make this a group_selector */
evh_selector = gevasevh_group_
➥selector_new();
gevasevh_group_selector_set_obj
➥ect(
  (GtkgEvasEvHGroup
  ➥Selector*)evh_selector,
  GTK_GEVASOBJ(gevas_image));
GtkgEvasObj* go  = ...;
make_selectable( gevas, go,
➥evh_selector );
...
/* lets make this object also
➥selectable */
void make_selectable( Gtkg
➥EvasObj* object,
            GtkObject* evhsel )
{
 GtkgEvasObj* ct  = 0;
 GtkObject* evh = gevasevh_
 ➥selectable_new( evhsel );
 gevasevh_selectable_
 ➥set_confine(
    GTK_GEVASEVH_SELECTABLE
    ➥(evh), 1 );
gevasobj_add_evhandler(object,
➥evh);
gevasevh_selectable_set_
➥normal_gevasobj(
   GTK_GEVASEVH_SELECTABLE
   ➥(evh),  object);
ct = (GtkgEvasObj*)gevasgrad_
➥new(
    gevasobj_get_gevas( GTK_
    ➥OBJECT(object)));
gevasobj_set_color( ct, 255,
➥200, 255, 200);
gevasgrad_add_color(ct, 120,
➥150, 170, 45, 8);
gevasgrad_add_color(ct, 200,
➥170, 90, 150, 16);
gevasgrad_set_angle(ct, 150);
gevasobj_resize( ct, 200,100);
gevasobj_set_layer(ct, 9999);
gevasevh_selectable_set_
➥selected_gevasobj(evh,ct);
}
```

Ezután már könnyen ellenőrizhető, hogy egy objektum kiválasztásra került-e, illetve létrehozhatunk egy gyűjtőobjektumot, amely műveleteket végez az összes kiválasztott objektumon:

```
GtkgEvasEvHGroupSelector* ev =
➥...;
GtkgEvasEvHSelectable*    o  =
➥...;
GtkgEvasObjCollection*    col =
➥0;
gboolean yn = gevasevh_group_
➥selector_isinsel(ev,o);
col = gevasevh_group_selector_
➥get_collection( ev );
gevas_obj_collection_move_
➥relative( col, 100, 200 );
```

Néhány objektumot, köztük a ***geTransAlphaWipe***-ot, továbbá azért hoztak létre, hogy képátmeneteket hajtsanak végre, még az ***Edje*** létezése előtt. Bár az ***Edje*** a jövő útja, az ***alphawipe*** kóddal egyszerű képátmenet valósítható meg az ***Edje*** bevonása nélkül. Ezt használja a ***gevasanim***, hogy foltszerű objektumot hozzon létre, amely keretről keretre változik az áttetszőség segítségével.
A `gEvas xxx_from_metadata()` függvényeivel egyetlen karakterfüzérrel beállítható egy új objektum helyzete, a képfájl neve, a láthatóság és más attribútumok. A `from_metadata()` és a képátmenettel kapcsolatos kód már megtalálhatók az ***Edje***-ben is:

```
sprite = gevas_sprite_new
➥( GTK_GEVAS(gevas) );
for( i=1; i<frame_count; ++i )
{
 gchar* md = g_strdup_printf(
   "cell%ld.png?x=120&y=
   ➥120&visible=0&fill_size=1"
   ,i);
 gi = gevasimage_new_from_
 ➥metadata( GTK_GEVAS(gevas),
 ➥md );
 g_free( md );
 gevas_sprite_add( sprite, GTK_
 ➥GEVASOBJ( gi ) );
}
gevas_sprite_set_default_frame_
➥delay( sprite,2000 );
gevas_sprite_play_forever(
➥sprite );
/* frame transitions */
geTransAlphaWipe* trans = 0;
trans = gevastrans_
➥alphawipe_new();
for( i=0; i<frame_count; ++i )
 gevas_sprite_set_transition_
 ➥function(
     sprite, i, trans );
```

Végezetül, a *gEvas* keretre egy *Edje* objektumot teszek. A cikkben is használt *gevasedje* bemutatóprogram a *gEvas* disztribúció *demo* könyvtárában található. A legérdekesebb részletek alább láthatók. Az *Edje*-nek egy forgó *Enlightenment* logója van, amely egérkattintásra lüktet:

```
/* init engines */
ecore_init();
edje_init();
gtk_init(&argc, &argv);
...
gevas = ...;
/* allow edje to update the
➥canvas as well */
gevas_setup_ecore( (GtkgEvas*)
➥gevas );
/* place an edje object */
GtkgEvasEdje* gedje
 = gevasedje_new_with_canvas
 ➥( gevas );
/* eet files can contain many
➥edje objects */
gevasedje_set_file( gedje,
➥"e_logo.eet", "test" );
go = GTK_GEVASOBJ(gedje);
gevasobj_move(      go, 300,
➥300 );
gevasobj_resize(    go, 370,
➥350 );
gevasobj_set_layer( go, 10 );
gevasobj_show(      go );
```

Zárszó

A memóriafogyasztással kapcsolatos eredményeket részben terjedelmi okokból mellőztem, részben pedig azért, mert a korszerű asztali számítógépekben ez a tényező kevésbé fontos. A *QTCanvas* és az *Edje* képátméretezésének igazságosabb összehasonlítását egy másik benchmarknak kell majd elvégeznie. Ha egy *GTK+2* alkalmazás készítéséhez fedőkönyvtárra lenne szükség, vegyük számításba a *GNOME Canvast* és a *gEvas-Edje* párost is, mielőtt nekiállnánk barkácsolni.

Linux Journal 2006., 141. szám

Ben Martin ideje javát virtuális fájlrendszerekkel és adatbányászattal tölti, bár úgy is jól ismert, mint aki szeret mások képeinek két- és háromdimenziós animációjával játszadozni.

KAPCSOLÓDÓ CÍMEK

- www.linuxjournal.com/article/8647

Új gép vásárlása helyett használjuk a Win4Lin Windows Virtual Desktop Servert

Az átállás néha nem is olyan nehéz. Különösen, ha a Win4Linx VDS segítségével Linuxot futtató klienseken és meg tudjuk jeleníteni a Windows szolgáltatásait...

A *Win4Lin Virtual Desktop Server (VDS*; ➲ *http://www.win4lin.com*) egy olyan kliens/szerver alapú virtualizációs megoldás, amellyel könnyedén és főként fájdalommentesen átállíthatunk egy szervezetet a drága és nehezen karbantartható ***Windows*** infrastruktúráról ***Linux*** használatára. Hogy hogyan? Nos ezzel a programmal egyetlen ***Windows*** szolgáltatásait akár több gép között is szétoszthatjuk. Nincs szüksége valamennyi felhasználónknak egy teljes Windows telepítésre? Nem gond! A ***Virtual Desktop Server (VDS)*** segítségével akár az is megoldható, hogy egy adott alkalmazás egy adott linuxos géphez rendelődjön hozzá. Látva ezeket a lehetőségeket talán az sem meglepő, hogy a tapasztalatok szerint a szervezetek a legkülönbözőbb okok miatt váltanak ***Linuxra*** a ***VDS*** segítségével.

Egy ***Linux*** kiszolgáló általában nem igényel akkor hardveres kapacitást, mint egy hasonló képességekkel rendelkező ***Windows***-t futtató szerver. Ez azt is jelenti, hogy a ***VDS*** segítségével – legalábbis egy időre – kiszállhatunk abból a mókuskerékből, amit az újabb és egyre erősebb gépek vásárlása iránti igény hajt. Ez pedig már csak azért is érdekes, mert a Vista megjelenésével számos szervezet lesz kénytelen komoly pénzeket költeni gépei lecserélésére az elkövetkező években. Ha a ***Linuxra*** történő váltásnak a ***TCO***-ra (***Total Cost of Ownership***) kifejtett hatását vizsgáljuk, pro és kontra mindenféle véleményt találhatunk. Akármi is azonban az igazság, az tény, hogy azok a cégek, amelyek gazdaságosabbnak vélik a nyílt forrású megoldásokat, és az átállás mellett döntenek, hamarosan szembesülnek ennek a döntésnek a logisztikai és infrastrukturális következményeivel. A ***VDS*** óriási előnye éppen az, hogy lehetővé teszi az alkalmazottaknak a régi ***Windows*** környezet és windowsos alkalmazások használatát is, miközben az alapszoftver már lecserélődött. A ***VDS*** ezzel csaknem végtelen kifutási lehetőséget biztosít a cég számára. Ha ugyanis a funkcionalitás megőrzése mellett az alapvető váltás már megtörtént, a vállalat vezetése dönthet úgy, hogy a továbbiakban is a kettős ***Linux/Windows*** virtualizációs rendszerrel folytatja az üzletmenetet, miközben nyitva bármikor áll az a lehetőség is, hogy a szükséges alkalmazásokat átírják az új operációs rendszerre, vagy kiváltják őket egy funkcióiban hasonló nyílt forrású megfelelővel.
A ***VDS*** akár egyetlen alkalmazást is képes ***Linux*** platformon megjeleníteni, ami azt jelenti, hogy ha a cégnek van egy küldetéskritikus programja, amitől semmiképpen nem tud „megválni”, nem kell azt átírnia más platformra. Az egyedi alkalmazásokat elindíthatjuk tehát úgy, hogy azok azonnal az ügyfél ***Linuxot*** futtató gépén jelenjenek meg.
Az infrastrukturális kérdések mellett a ***VDS*** lehetőséget teremt arra is, hogy központosítsuk a rendszerfelügyeletet, a szoftverfrissítéseket és a karbantartási feladatokat. Mivel valamennyi kliens ugyanazt a ***Windows*** képet használja, a kiszolgálón végzett valamennyi módosítás hatása azonnal érvényesül a hálózat valamennyi gépén.
A meglevő ***Windows*** licenszeket a vállalat tulajdonképpen változatlan feltételek mellett használhatja, csak a használat módja változik.
A ***VDS*** jelentősen eltér a piacon ma fellelhető kliens/szerver alapú virtuális géptől (***Virtual Machine***; ***VM***). A legtöbb ilyen termék egyszerűen csak

egy képernyőt nyújt a kliens számára, míg a *VDS* egy az *X Windows Systemen* alapuló valódi kliens/szerver rendszert képez. Ha az ügyfél nem támogatja az *X* protokollt, még mindig lehetőségünk van olyan alternatív display alapú megoldások használatára mint a *Virtual Network Connection* vagy a tarantella.
Ez megnyitja a lehetőséget gyakorlatilag minden olyan platform előtt, amelyen futtatni lehet egy viszonylag modern webböngészőt.
A *Windowst* futtató rendszerek rendszergazdáinak napjait javarészt az tölti ki, hogy mindenhova telepítsék a megfelelő foltokat és biztonsági javításokat. Csak így kerülhetik el az ezen a platformon általánosnak tekinthető kémprogramokat és egyéb kártevőket, amelyek rendszerint a kevésbé alaposan felügyelt gépek közvetítésével kerülnek be a rendszerbe.
Amint korábban említettem, a *VDS* egyetlen *Windows* képpel működik, aminek a karbantartása nem csak nagyságrendekkel kevesebb időt igényel, hanem nagyban egyszerűsíti is a rendszergazda életét, hiszen a továbbiakban nem kell azzal foglalkoznia, hogy a javítások tovaterjedjenek a teljes hálózati infrastruktúrán belül. Ráadásul mivel a végfelhasználó környezetét a központi Windows kép beállításai, és a helyi beállítások együttesen határozzák meg, ha az illető egyszerűen kijelentkezik, majd visszalép a rendszerbe, azzal automatikusan „kiölt” belőle minden, menet közben esetleg beszivárgott kártékony programot.

Ügyfelek

A *Win4Lin* a natív *Win4Lin Terminal Services Client* használatát javasolja, amely rendelkezik minden olyan szolgáltatással, amit egy *Win4Lin* kliens/szerver kapcsolat egyáltalán nyújtani tud. Ezzel együtt számtalan módja van annak, hogy egy *VDS* kiszolgálóhoz kapcsolódjunk és hacsak nem szeretnénk minden áron elkerülni valamennyi a nyomtatással kapcsolatos nehézséget, akkor gyakorlatilag bármilyen kliensprogram megteszi.
Egy *VDS* kiszolgálóhoz kapcsolódhatunk a *Telnet*, az *rlogin* vagy az *SSH* segítségével is (utóbbi esetben engedélyezni kell az *X11* forgalom továbbítását). A *Telnet* és az *rlogin* esetében megfelelően be kell állítanunk a `{$DISPLAY}` környezeti változó értékét. Általánosságban a távoli bejelentkezés csak akkor járható út, ha olyan megfelelő sebességű hálózati infrastruktúra áll rendelkezésre, mint amilyen például egy helyi *LAN*. A *WAN* illetve a fogyasztói kategóriába tartozó nagy sebességű hálózati kapcsolatok általában nem nyújtanak kellően nagy sávszélességet ezekhez a módszerekhez. Ugyanakkor kialakíthatunk kapcsolatokat *RealVNCkliens*, *NoMachine* ügyfél, vagy a *Tarantella* nevű csomag segítségével is.
Röviden talán azt lehetne mondani, hogy aki nem volt képes egy *VDS* kiszolgálóhoz kapcsolódni, az nem próbálta eléggé.

Licenszfeltételek

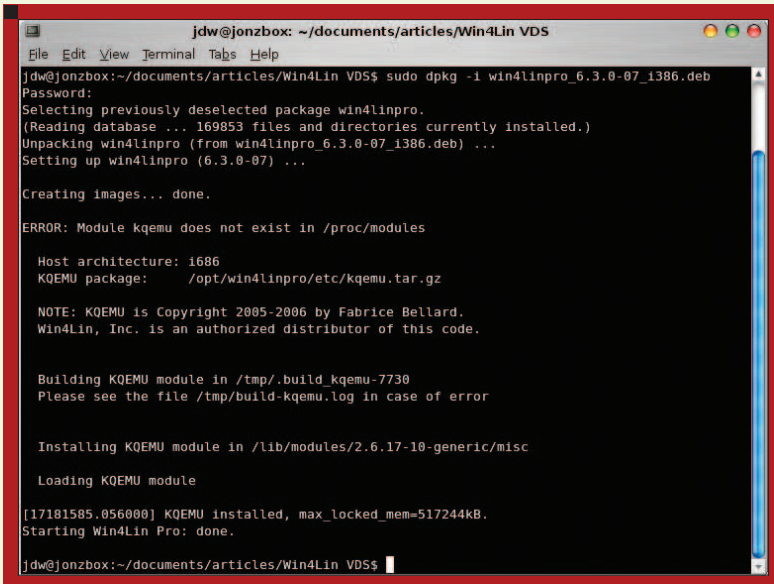
Egy alaplicensz 25 munkaállomáshoz 2500 dollárba kerül. Ezt aztán különböző lépésekben, különböző feltételekkel egészíthetjük úgy, hogy végül egy *VDS* szerver akár 1000 felhasználót is kiszolgálhat. Ehhez persze egy kellően erős gép is kell, hiszen 1000 felhasználó már valóságos tömeg.
Fontos hangsúlyozni, hogy a fenti adatok a *VDS* licenszére vonatkoznak, és nem a Windows operációs rendszer használatának jogára. A *VDS*-t használó szervezeteknek rendelkezniük kell a szükséges számú *Windows* licensszel, és természetesen a *Microsoft* által támasztott felhasználási kritériumok is érvényben maradnak.
Ugyanakkor a jelenlegi gyakorlat azt mutatja, hogy a szervezetek fölhasználhatják a már megvásárolt *Windows* használati jogokat.

Letöltés és telepítés

A *Win4Lin VDS* jelenlegi legfrissebb változata a 3.0-ás, de már a 3.5-ös is a bétatesztelés utolsó fázisában van, és mire ez a cikk megjelenik, már valószínűleg el is készül. Mi mindazonáltal a cikk írásakor stabil 3.0-ás változatot használtuk, hiszen nem láttuk különösebb értelmét annak, hogy „mozgó célpontra tüzeljünk”.
A *Win4Lin Pro* és *VDS* változatát ugyanaz a bináris állomány tartalmazza, de a különböző licenszekkel különböző funkciókat lehet aktiválni. Van *DEB* és *RPM* csomag, valamint 32 és 64 bites változat egyaránt.
A licenszhez tartozó kódot a telepítőprogram kéri be, majd – amint már mondtam – ez fogja eldönteni, hogy ami a folyamat végén a gépen keletkezik, az egy *VDS* vagy *Pro* változat lesz. Talán érdemes megjegyezni, hogy a *Pro* változatot magán a programon belül nem lehet *VDS*-sé frissíteni, vagyis ha van egy telepített *Pro* verziónk és magasabb osztályú licenszet vásárolunk, akkor kézzel kell futtatni az *ask_license.sh* szkriptet.
VDS ügyfelek léteznek *Linux*, *Solaris* és *Windows* operációs rendszerhez, de a forráskód is letölthető, ami azt jelenti, hogy egy közepesen képzett *Linux* rendszergazda gyakorlatilag bármilyen platformra lefordíthatja azt.

A kiszolgáló telepítése

Mi a telepítéshez használt csomagokat közvetlenül a *Win4Lin FTP* helyéről (➲ *ftp.win4lin.com/pub/releases/linux/pro/3.0/index.html*) töltöttük le. Ahogy korábban említette, léteznek 32 és 64 bites *RPM*, és ugyanilyen *DEB* csomagok is. Mindenki válassza azt, amelyik az általa használt környezetnek leginkább megfelel. Végső esetben létezik közönséges tarlabda is, és persze ebből is van 32 és 64 bites változat is.
Mi egy 1.83 GHz-es *Centrino Duo core* processzorral és 1 GB memóriával szerelt gépen próbáltuk ki a szoftvert, amelyen egy béta állapotú *Edgy Eft* futott. Ennek megfelelően az *RPM* csomagot töltöttük le. Végül aztán kénytelenek voltunk az alaprendszert lecserélni *SUSE 10.1*-re, mivel az *Edgy* saját rendszermagja valahogy nem jött ki a *Win4Lin* klienssel. Úgy gondoltuk, hogy a teszteléshez a *SUSE* terjesztés is tökéletesen megfelel majd, meg aztán eléggé izgatottak is voltunk. Mindezt csak azért említem meg, mert a szemfülesebb olvasók az ábrák között böngészve több különböző terjesztést fedezhetnek föl. Nos, ez volt az oka ennek a furcsa jelenségnek.
A *win4linpro_6.3.0-07_i386.deb* csomag mérete mindössze 3,7 MB volt, és egyetlen zokszó nélkül települt.
A *VDS* telepítési útmutatója szerint célszerű először azokat a segédeszközöket föltelepíteni, amelyek a támogatott terjesztésekhez szükségesek és csak aztán magát a *VDS*-t. Ennek pedig az az oka, hogy valamennyi *Win4Lin* terjesztés tartalmaz egy speciális *KQEMU* modult, ami nagyban



1. ábra A modulok lefordítása és telepítése

növeli az egész rendszer teljesítményét. Mármost ha a tanácsnak megfelelően előre feltelepítjük azokat az eszközöket, amelyek ennek a beüzemeléséhez szükségesek, akkor telepítésnél a folyamat gyakorlatilag láthatatlan módon végbemegy (*1. ábra*).
És ezzel el is érkeztünk ahhoz a ponthoz, amikor föl kell telepítenünk azt az egyetlen ***Windows*** példányt, amivel a rendszer működni fog. A próbához egy ***Windows XP Home*** változatot használtunk. Föltelepítettük, aztán hagytuk, hogy letöltse a ***Service Pack 2***-t, mielőtt létrehoztuk volna a felhasználói fiókokat. Mindezt úgy hajtottuk végre, hogy behelyeztük a telepítőlemezt a meghajtóban, aztán kiadtuk a következő parancsot:

```
sudo loadwinproCD
```

Ha még nem telepítettük volna a ***VDS*** licenszet, akkor a program ezen a ponton felszólít bennünket a megadására. Van ugyanakkor egy olyan lehetőségünk is, hogy nem adunk meg licenszet, hanem csupán folytatjuk a telepítést. Ekkor a rendszert 14 napig ugyan díjmentesen használhatjuk, de a kiszolgáló egyetlen funkciója sem lesz aktív, vagyis gyakorlatilag egy egyfelhasználós munkaállomás birtokába jutunk.
Amint az *1. ábrán* látható a vendégrendszert tartalmazó adathordozót a `-r` kapcsoló segítségével töltöttük be. Ennek az az oka, hogy a tesztrendszerünkön már volt egy korábban telepített ***Win4Lin Pro***, a `loadwinproCD` parancs pedig észrevette ezt. A -r kapcsoló egyszerűen csak arra utasítja a ***VDS***-t, hogy töltse be újra a vendégrendszert tartalmazó médiát.
A ***VDS*** telepítése gyakorlatilag két lépésből áll. Először a rendszergazdának be kell másolni a vendégrendszert tartalmazó médiát a merevlemezre. Ezt esetünkben a `loadwinproCD` parancs végzi el. A következő lépés a ***Windows*** tényleges telepítése, amit viszont már közönséges felhasználóként végezhetünk el. A ***Windows*** telepítésének és a mesterprofil létrehozásának a folyamata szorosan összefügg egymással. A mesterprofil azért lényeges, mert valamennyi további felhasználói fiókhoz tartozó ***Windows*** környezet ebből származik majd.
Ez a kitétel annyira fontos, hogy a most következő teljes szakaszt a folyamat megértésének fogjuk szentelni.

Profil alapú kiszolgálás

A ***Win4Lin VDS*** rendszer úgynevezett profil alapú kiszolgálást végez. Ez dióhéjban annyit tesz, hogy elő kell állítanunk egyetlen mesterprofilt, amit minden felhasználó örökölni fog. Amikor foltokat, javításokat telepítünk, egyéb karbantartási feladatokat végzünk, szoftvereket frissítünk, vagy alkalmazásokat telepítünk, az mindmind kizárólag ezt a mesterprofilt fogja érinteni. Amint rendelkezésünkre áll egy tökéletesen beállított mesterprofil, azt a rendszer minden egyéb felhasználója automatikusan örökli.

A mesterprofil létrehozása

A ***VDS*** rendszert egy közönséges felhasználói fiókba alá kell telepítenünk. Mint minden, a teljes infrastruktúrát érintő döntésnél, itt is igaz, hogy ha előzetesen egy kicsit eltöprengünk a dolgok mikéntjén, azzal sok-sok későbbi fejfájást spórolhatunk meg.
Én úgy döntöttem, hogy a mesterprofilt a saját felhasználói fiókomba (***jdw***) fogom telepíteni. Aztán azt is elhatároztam, hogy két másik felhasználói fiókot is ki fogok szolgálni. Ezek a ***jwatson*** és a ***dwatson*** nevet viselték. A legtöbb szervezetnél persze ez a dolog nem így fog működni, hiszen nyilván van egy már működő hálózati infrastruktúrájuk, amelyben eleve léteznek a szükséges felhasználói fiókok, ilyenkor célszerű egyetlen új fiókot létrehozni, amely kizárólag a ***VDS*** mesterprofilt fogja hordozni.
A mesterprofil létrehozásához először is bejelentkeztem egyszerű felhasználóként, majd kiadtam a következő parancsokat:

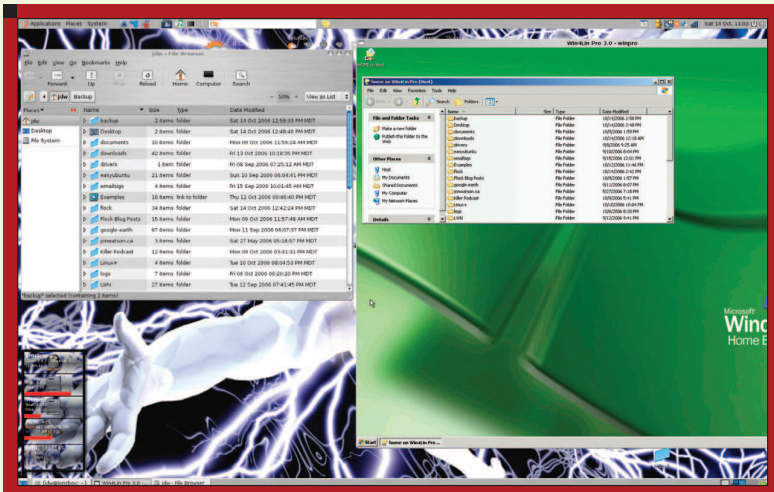
```
sudo adduser jwatson
sudo adduser dwatson
```

Miután az így létrehozott két felhasználói fiókhoz a megfelelő jelszavakat is hozzárendeltem, készen álltak a használatra. Most már csak azt a bizonyos mesterprofilt kellett létrehoznom. Ehhez telepítenem kell a ***Windows-t***, ki kell jelölnöm mesterpéldánynak, majd el kell végeznem rajta a globálisan érvényesíteni kívánt beállításokat.
Kijelentkeztem tehát, majd újra beléptem, de ezúttal már a mesterprofil hordozására szolgáló fiók (***jdw***) adataival, majd kiadtam az `installwinpro` parancsot.
Ennek a parancsnak számos kapcsolója van, amelyekkel részletesen meg lehet határozni annak a virtuális gépnek a tulajdonságait, amelyre a ***Windows*** települ. Én ezek használata helyett egyszerűen elfogadtam az alapértelmezéseket, ami például azt is jelenti, hogy a lemezkép legfeljebb 4 GB lehet, de ez számos esetben nem biztos, hogy elegendő.

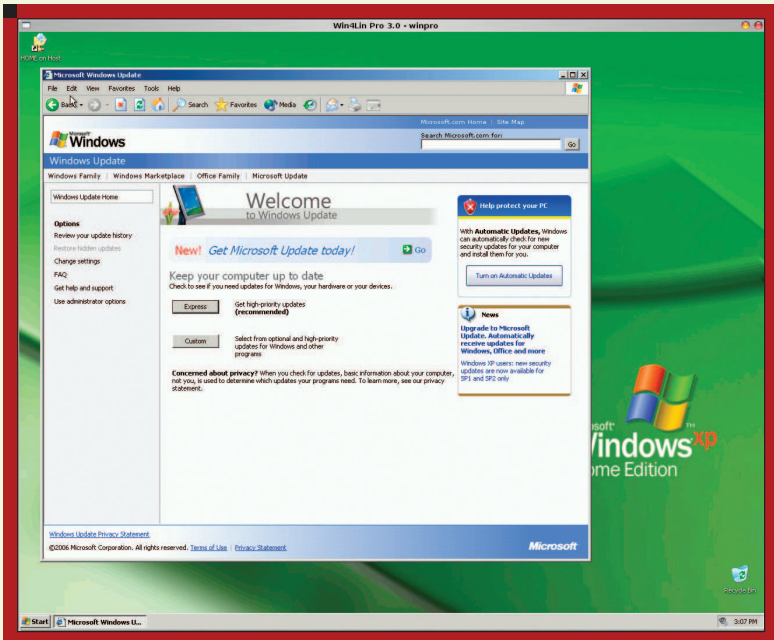
Ugyanakkor a számolásnál ne felejtsük el, hogy a felhasználók által használni kívánt területet nem kell hozzáadni a rendszer méretéhez, a felhasználói fájlok ugyanis a ***Linux*** fájlrendszerében tárolódnak a felhasználó saját könyvtárában, nem pedig a lemezképben (*2. ábra*).
A profil alapértelmezett neve ***winpro*** lesz, amit szükség esetén a `-d` kapcsoló segítségével változtathatunk meg. Ezen a ponton fontos megjegyezni, hogy ha megváltoztatjuk a konfiguráció nevét, akkor a következő lépésben ezt az információt exportálnunk kell a mesterprofilba.
Azt javaslom, mindenki fogadja meg a tanácsomat, és készítsen biztonsági másolatot a képfájlról ebben az állapotban. Szerintem egyetlen Windows rendszertől függni meglehetősen macerás dolog, nem gondolom, hogy bárkinek hiányozna egy teljes újratelepítés.
A kép mentése amúgy rendkívül egyszerű, hiszen csupán a ***GUEST.IMG*** fájlt kell átmásolnunk egy biztonságos helyre. Ha a telepítés során mindent az alapértelmezett értéken hagytunk, akkor ez a fájl a /home/jdw/winpro könyvtárban kell legyen (a felhasználó neve természetesen változhat).
Aki pedig változtatott menet közben a beállításokon, az nyilván emlékszik rá, mit hol kell keresnie.
Amint a telepítéssel elkészültünk, ki kell jelölnünk a kérdéses rendszert mesterprofilnak. Ezt a következő parancs lefuttatásával tehetjük meg:

```
/opt/win4linpro/bin/export-
➥profile <configuration name>
```

Ha az előző lépésben megváltoztattuk a konfiguráció nevét, akkor azt meg kell adnunk a `<configuration name>` szöveg helyén. Ha nem adunk meg nevet, akkor a program automatikusan a `winpro`-t fogja használni. Ilyenkor a mesterprofil kijelölésekor egyáltalán nem kell konfigurációnevet megadnunk.
Indítsuk el a telepített rendszert a lemezképből, és végezzük el a szükséges beállításokat. Nagyon fontos, hogy a mesterprofil előállításához legalább egyszer el kell indítanunk a rendszert, és tisztán le kell állítanunk, még akkor is, ha semmit nem csinálunk vele.
A mesterprofil futtatásához vagy kattintsunk a ***Win4Lin*** ikonjára az asztalfelületen, vagy parancssorból adjuk ki a `winpro` parancsot.



■ ***2. ábra*** A felhasználói dokumentumok és könyvtárak a Linux fájlrendszerében tárolódnak



■ ***3. ábra*** Frissítsük a Windows mesterképet tetszés szerint, de ha már van egy jól működő változatunk belőle, azt mindenképpen mentsük el...

A lemezkép beállítása

Mivel a ***VDS***-en keresztül kiszolgált felhasználók minden egyes bejelentkezéskor megkapják a mesterkép egy friss példányát, különösebben nagy jelentősége nincs annak, hogy aprólékosan beállítsunk mindent. Elméletileg azt is megtehetnénk, hogy ezen a ponton teljesen kihagyjuk a felhasználói fiókok beállítását, majd később visszatérve tovább finomítjuk a mesterkép tulajdonságait. Ugyanakkor van néhány olyan gyakorlati szempont, amelyek miatt célszerű a beállításokat most elvégezni:

- ***Az „ismerős jelleg" biztosítása:*** Egy szervezet általában azért vált a ***VDS*** használatára, mert minimálisra szeretné leszorítani a felhasználók rendszerre gyakorolt befolyását. Éppen ezért a józan ész azt diktálja, hogy a felhasználót már az első pillanatban is ismerős

képnek – tehát teljesen beállított rendszernek – kell fogadnia, ahol rögtön megtalálja az összes, számára fontos alkalmazást is.

- ***Biztonság:*** Mivel a felhasználók mindannyiszor egy új mesterképet kapnak, valahányszor bejelentkeznek a *VDS* rendszerbem az ember azt gondolhatná, hogy a biztonsági szakemberek ilyen kiszolgálás mellett nyugodtan hátradőlhetnek. Elvégre mennyi kért tehet egy kémprogram, vagy féreg egy olyan rendszerben, aminek a teljes tartalma naponta legalább egyszer felülíródik egy központi helyről, egy tiszta kópiával? Nos, ez igaz, de képzeljük el, mi van akkor, ha maga a központi lemezkép tartalmaz egy ilyen kártevőt. Egyszóval jobb a problémát megelőzni, és gondoskodni a megfelelő biztonsági beállításokról (*3. ábra*).

- ***Műszaki megfontolások:*** A mesterprofilt nem futtathatjuk, ha akár csak egyetlen felhasználói profil is fut. Ugyanakkor ahhoz, hogy bármit módosíthassunk, el kell indítani. Mármost ebből egyenesen következik, hogy vagy most azonnal elvégezzük az összes olyan beállítást, amit csak lehet, vagy számolnunk kell azzal, hogy a közeljövőben a rendszer sokat fog állni. Ha pedig munkaidőben próbálkozunk, akkor azt egyeztetni kell az összes felhasználóval, hiszen átmenetileg mindenkinek ki kell jelentkeznie a *VDS*-ből, és amíg nem végeztünk a mesterprofil módosításaival, addig nem is jelentkezhet vissza.

Talán ez az pillanat, amikor érdemes egy újabb mentést készíteni a mesterképről. Mostantól valahányszor valamit módosítunk a beállításokon, célszerű megőrizni az eredeti állapotot is, hátha valami rosszul sül el. Ne felejtsük el, hogy ha a központi lemezkép megsérül, akkor az összes felhasználónk munkaképtelenné válik, amíg le nem cseréljük azt egy működő változatra. Egy 4 GB-os lemezkép mentése vagy visszaállítása csak néhány percet vesz igénybe.
Ha ezzel szemben teljesen újra kell telepíteni magát a ***Windowst***, vagy egy meglehetősen régi lemezképet kell frissíteni, az sokkal hosszabb időbe telik.
A ***Win4Lin VDS*** szolgáltatásnak normális esetben magától el kell indulnia. Ha mégsem így lenne, vagy valamiért újra kell indítanunk, azt a

```
/opt/win4linpro/etc/mergepro_rc
➥start
```

vagy a

```
/opt/win4linpro/etc/mergepro_rc
➥restart
```

paranccsal tehetjük meg.

A felhasználói profilok létrehozása

A mesterprofilt csak az arra feljogosított felhasználók használhatják.
A rendszer azon felhasználóit, akiknek a munkájához nem szükséges a *VDS* használata, értelemszerűen nem kell felruházni ezzel a jogosultsággal. Ugyanakkor akinek ilyen jogot szeretnénk adni, annak a saját könyvtárában kell bizonyos módosításokat elvégeznünk. Ha csak néhány felhasználónk van, akkor a legegyszerűbb valószínűleg az lesz, ha egyenként átjelentkezünk valamennyiük fiókjába a su paranccsal, majd lefuttatjuk a megfelelő parancsot. Ha azonban a kiszolgált felhasználók száma átlépi a „kényelmetlenül sok" feliratú határt, akkor ügyes ***Bash***-szkriptekhez, vagy bejelentkezési szkriptekhez kell folyamodnunk a művelet elvégzéséhez.
Az én esetemben nincs szó különösebben nagy tömegről, tehát egyszerűen csak be kell jelentkeznek ***jwatson*** és ***dwatson*** néven is, majd lefuttatni a következő parancsot:

```
/opt/win4linpro/bin/import-
➥profile /home/jdw/winpro
```

Most ott tartunk, hogy létrehoztuk a mesterprofilt, a felhasználóknak pedig kiosztottuk a megfelelő jogosultságokat. Ami tehát a kiszolgálón elvégzendő munkát illeti, annak a nagy részével készen is vagyunk. Ha mostantól bármilyen módosítást végzünk a mesterprofilon (beleértve a foltok, biztonsági javítások, vagy új alkalmazások telepítését is) az automatikusan tovaterjed a felhasználói profilokra is, amint a tulajdonosuk bejelentkezik.
Na de hogyan fognak ezek a bizonyos felhasználók bejelentkezni? Természetesen erre való az ügyfélprogram, amit mindjárt telepíteni is fogunk.

Az ügyfél telepítése

Amint azt korábban is említettem, számos különböző módja van annak, hogy egy *VDS* kiszolgálóhoz csatlakozzunk. Én itt most csak a ***Win4Lin*** klienssel történő csatlakozást fogom bemutatni. Ez szabadon letölthető, és a kezelése is egyszerű.
Ha a *VDS* kiszolgáló elérésére a natív ***Win4Lin*** ügyfélprogramot használjuk, annak ráadásul az az előnye is megvan, hogy így érhetjük el a legnagyobb sebességet, illetve a legtöbb szolgáltatást. A ***Win4Lin*** ügyfelet a ***Win4Lin*** hivatalos webhelyéről (➲ ***www.win4lin.com/component/option,com_repository/Itemid,76/func,fileinfo/id,2***) lehet letölteni. Találunk itt lefordított programot ***Linux***, ***Solaris*** és ***Windows*** rendszerekhez, de letölthető akár a forráskód is. Kissé furcsa, hogy bár maga a *VDS* rendszer ***DEP*** és ***RPM*** csomag formájában egyaránt letölthető, addig a ***Win4Lin*** klienshez nem tartozik ***DEB*** csomag. Mivel a forrásból való telepítés az én esetemben általában azt jelenti, hogy kénytelen vagyok lemondani az ebédemről, inkább letöltöttem az ***RPM*** csomagot, amit aztán az ***Alien*** nevű alkalmazással ***DEB*** formátumúvá alakítottam. Végül a `dpkg` paranccsal telepítettem a korábban említett ***Debian***-alapú rendszeremre. A lépések részletesen a következők voltak:

1. Letöltöttem az ***RPM*** csomagot a ***Win4Lin*** webhelyéről.

2. Futtattam a `sudo alien wtsclient_1.0.0-4_i386.rpm` parancsot.

3. Telepítettem a keletkezett ***DEB*** csomagot a `sudo dpkg -i wtsclient_1.0.0-4_i386.deb` paranccsal.

4. A tsclient paranccsal csatlakoztam a *VDS* kiszolgálóhoz.

A ***Win4Lin VDS*** rendszert úgy is beállíthatjuk, hogy egyetlen alkalmazást közvetítsen egy teljes ***Windows*** desktopra. Mi a teszt során mindkét

említett konfigurációval csatlakoztunk a *Win4Lin* demókiszolgálóhoz, és az akcióról mellékeltünk két képernyőmentést. Az első egy teljes *Windows* desktop beállítást mutat, míg a második csak az *Internet Explorert*, amint éppen *Linux* munkaállomásunkon fut vidáman (*4. és 5. ábra*).

Egyedi alkalmazások közvetítése a Linux asztalfelületre

Egészen eddig csak arról esett szó, miként közvetíthetünk egy *Linuxot* futtató gépre egy teljes *Windows XP* asztalfelületet. Bár ez a megközelítés számos esetben a legmegfelelőbb, azért akadhatnak olyan helyzetek is, amikor a felhasználónak magára az operációs rendszerre nincs is szüksége, csak egy-egy konkrét alkalmazásra. A kérdés tehát az, miként vehetjük rá a *VDS* rendszert arra, hogy a teljes *Windows* telepítés helyett csupán egyetlen alkalmazást adjon át a bejelentkezett felhasználónak. Természetesen a *Windows* regisztrációs adatbázisának az ügyes átalakításával. Hogy ezt miként tehetjük meg, arról a legfrissebb információkat a *Win4Lin VDS* kézikönyvében találhatjuk, de ha valaki egyszerűen csak annyit szeretne, hogy ezzel a lapszámmal a kezében leül a számítógép elé, és beállítja, amit kell, nos annak röviden leírjuk a dolog menetét.

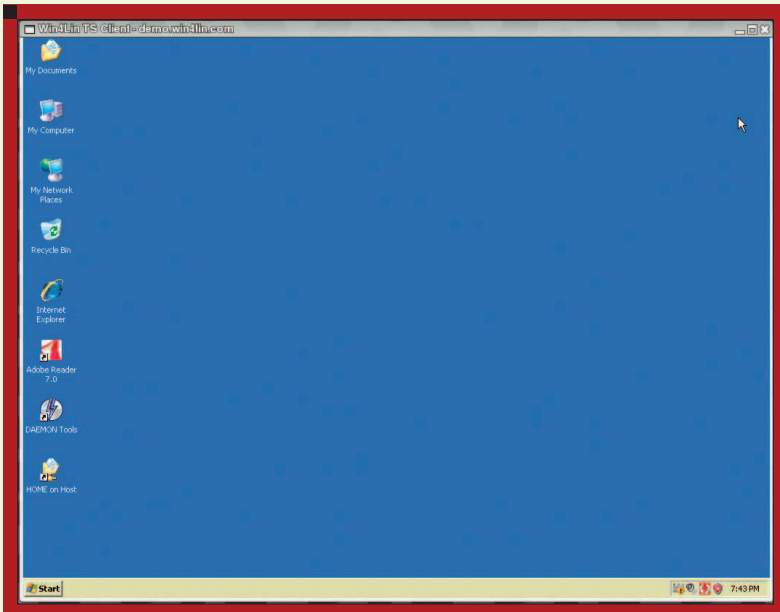
Győződjünk meg a megfelelő Win4Lin registry kulcs meglétéről (valamennyi Windows változatra érvényes)

Függetlenül attól, hogy milyen alkalmazást, vagy a *Windows* operációs rendszer mely változatát kívánjuk futtatni, van egy olyan, a *Win4Lin* rendszerre specifikus kulcs a regisztrációs adatbázisban (*registry*), amelynek meglétét ellenőriznünk kell:

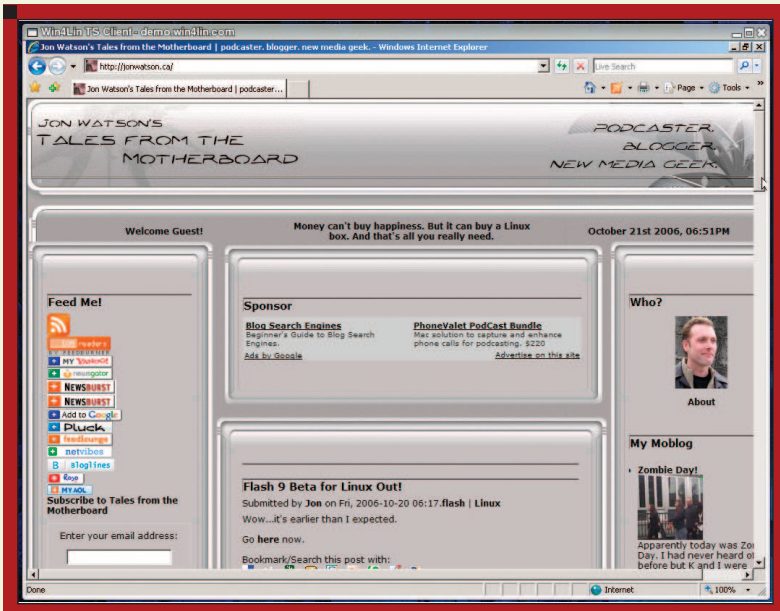
1. Indítsuk el a *Regedit* nevű alkalmazást.

2. Keressük ki a *HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon* kulcsot.

3. Győződjünk meg róla, hogy a *Userinit* nevű változó értéke B:\mrgpro32.exe. Ha nem így lenne, módosítsuk erre az ott szereplő értéket.



■ *4. ábra* A Win4Lin ügyfél, amint éppen egy teljes desktopot futtat



■ *5. ábra* A Win4Lin itt csupán egyetlen alkalmazást közvetít a Linux munkaállomásra egy teljes asztalfelület helyett

Annak ugyebár nem sok értelme van, hogy a felhasználó minden alkalommal szabályszerűen bejelentkezzen a *Windowsba*, ha mindannyiszor ugyanazt az egyetlen alkalmazást akarja futtatni. Éppen ezért az első lépés az – bár ez végül is nem kötelező – hogy a mesterprofilban automatikus bejelentkezést állítunk be.
Ez a *Windows* különböző változataiban eltérő módon érhető el.

Az automatikus bejelentkezés beállítása Windows 2000 alatt

1. Nyissuk meg a *Vezérlőpultot* (*Control Panel*).

2. Indítsuk el a *Felhasználók és Jelszavak* (*Users and Passwords*) kisalkalmazást.

3. Vegyük ki a jelölést a *„Felhasználóknak felhasználói*

nevet és jelszót kell megadniuk ennek a számítógépnek a használatához" („Users must enter a user name and password to use this computer") pont mellől, majd nyomjuk meg az *OK* gombot.

4. Amikor a rendszer kéri, adjuk meg annak a felhasználónak a nevét és jelszavát, akinek a fiókjába automatikusan be akarunk jelentkeztetni minden felhasználót.

Az automatikus bejelentkezés beállítása Windows XP alatt

1. Nyissuk meg a *Vezérlőpultot*.
2. Válasszuk a *Felhasználói fiókok* (*User accounts*) pontot.
3. Kattintsunk a *„Felhasználók be- és kijelentkezési módjának megváltoztatása"* (*Change the way users log on/off*) pontra.
4. Vegyük ki a jelet az *„Üdvözlő képernyő használata"* (*Use the Welcome screen*) jelölőmezőből.
5. Kattintsunk az *Alkalmaz* (*Apply*) gombra.
6. Indítsuk el az alternatív felhasználói fiókszerkesztőt úgy, hogy a *Start* menüből a *Futtatás* (*Run*) pontot választjuk, majd begépeljük a `control userpasswords2` parancsot, és az *OK* gombra kattintunk.
7. Vegyük ki a jelet az *„A felhasználóknak felhasználói nevet és jelszót kell megadniuk ennek a számítógépnek a használatához"* (*„Users must enter a user name and password to use this computer"*), majd kattintsunk az *OK* gombra.
8. Amikor a rendszer kéri, adjuk meg annak a felhasználónak a nevét és jelszavát, akinek a fiókjába automatikusan be akarunk jelentkeztetni minden felhasználót.

Az egyetlen futtatni kívánt alkalmazás kijelölése (valamennyi Windows változaton)

1. Indítsuk el a regisztrációs adatbázis szerkesztésére szolgáló alkalmazást úgy, hogy a *Start* menü *Futtatás* (*Run*) pontját választjuk, majd begépeljük a `regedit` parancsot, és az *OK* gombra kattintunk.
2. Keressük ki a *HKLM\Software\Win4Lin* kulcsot.
3. Kattintsunk a jobb egérgombbal a jobb oldali ablakszárny egy üres területén, és hozzunk létre egy új változót.
4. Válasszuk a karakterlánc (*String Variable*) típust.
5. A változó neve legyen *SingleAppStart* (ügyeljünk rá, hogy itt a kis- és nagybetűk különbözőnek számítanak).
6. Kattintsunk duplán az újonnan létrehozott *SingleAppStart* változón.
7. Az *Érték* (*Value*) mezőben adjuk meg annak az alkalmazásnak a nevét és teljes elérési útvonalát, amit automatikusan indítani szeretnénk. Az útvonal megadása nélkül a dolog nem működik, vagyis ha a futtatni kívánt alkalmazás például a *Microsoft Word*, akkor a `WORD.EXE` szöveg nem elegendő. Az elérési út ebben az esetben a `C:\Program Files\Microsoft Office\word.exe` (vagy ahova éppen a *Word* végrehajtható állománya telepítve van az adott rendszeren).
8. Lépjünk ki a *regeditből* és készen is vagyunk.

Mostantól amikor a felhasználók elindítják a kliensprogramot és bejelentkeznek a *VDS* rendszerbe, azonnal a *Microsoft Word* fog nekik elindulni. Magából a *Windowsból* gyakorlatilag nem is látnak semmit. Ez persze így nem tűnik valami nagy eredménynek, hiszen a *Microsoft Office* részét képező alkalmazásokat kiválóan támogatja a *Wine* és a *CrossOver Office* is, de cseréljük csak ki a *Word*-öt valami másra, és rögtön rájövünk, mi is a jelentősége a *Win4Lin VDS* által alkalmazott megoldásnak. Mivel esetünkben végső soron a *Windows* egy teljes példányát használjuk a kérdéses alkalmazás közvetítésére, gyakorlatilag bármi szóba jöhet, ami egy „igazi" *Windows* rendszeren is futna, és ehhez magán az alkalmazáson az égvilágon semmit nem kell módosítanunk. Ugyanez a másik két megoldásról – egyelőre – nem mondható el.

Záró gondolatok

A virtuaálizáció technológiája nem számít különösebben újnak. Ami azt illeti, az *IBM* már az 1960-as évek óta játszadozott a gondolattal. Ami a virtualizációt hirtelen ismét érdekessé tette, az a nagy sebességű hálózatok és erős kiszolgálók általános elterjedése. Ha *Windows* alkalmazásokat *Linux* operációs rendszer segítségével „juttatunk el" a felhasználó képernyőjére, az nem csak a beszerzendő hardver szempontjából lehet költséghatékony, hanem az emberek képzésére és a rendszer karbantartására is kevesebbet kell költeni. Az olyan megoldások pedig mint amilyen a most bemutatott *Win4Lin Virtual Desktop Server* ráadásul egészen könnyen telepíthetők és használhatók.
A virtualizáció tehát eljutott arra a pontra, amikor az igazi problémák nem nem műszaki, inkább csak logisztikai természetűek. Manapság nehezebb helyesen megtervezni egy virtualizációs rendszert, mint megvalósítani azt.

Linux Journal 2007., 154. szám

Jon Watson
(http://www.jonwatson.ca) kanadai GNU/Linux aktivista, aki rendszeresen közöl cikkeket a Linux közösség legkülönbözőbb fórumain. Amikor éppen nem ír, nem blogol, és nem is a szabad és nyílt forrású szoftvereket népszerűsítő podcastokat készít, akkor rendszerint a szobájában fényesítgeti azt a Linux+ tanúsítványát, ami rajta kívül valahogy nem sokakat hat meg.

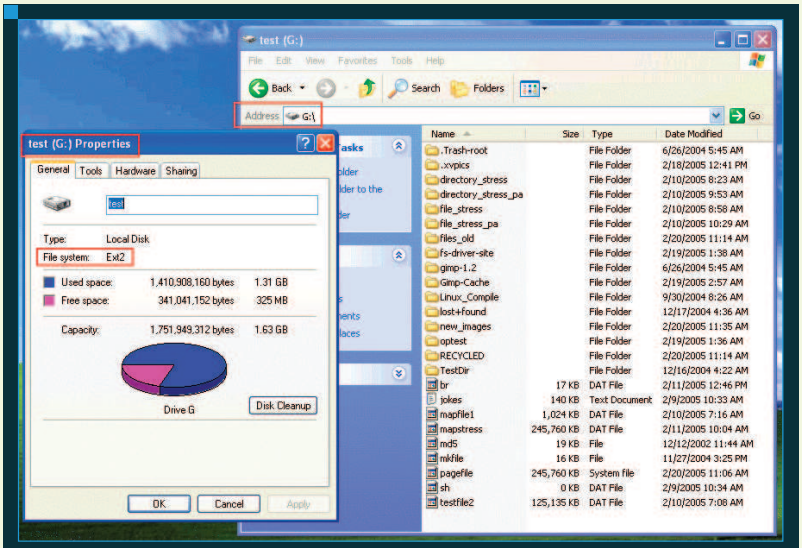
Linux fájlrendszerek kezelése Windows alól

Jó hír: nem kell a Microsoft fejlesztéseire várnunk, ha Windows alól is kezelni szeretnénk linuxos fájlrendszereinket...

Ha egy eddig használt operációs rendszerről egy másikra szeretnénk váltani, annak gyakorta a legnagyobb gátja az a tudat, hogy az új rendszer alól nem fogjuk tudni elérni a régi alatt létrehozott adatokat. És ez valahol érthető is. Senkit nem kárhoztathatunk azért, mert nem fűlik a foga ahhoz, hogy egyik pillanatról a másikra elveszítse az összes adatát. Lehet, hogy az új rendszer szebb és jobb, mint a régi, de az adatok nélkül gyakorlatilag használhatatlan.
Ami a ***Linuxot*** illeti, a fejlesztők mindent megtettek annak érdekében, hogy a ***Windowst*** is használók hozzáférjenek az ott létrehozott adataikhoz. A ***FAT16/32*** fájlrendszerek támogatása már régóta gyakorlatilag tökéletes, ami pedig az ***NTFS***-t illeti, nos az még mindig nem teljes, de nem áll messze tőle. Itt érdemes megjegyezni, hogy már olyan eszközök is vannak – ilyen például a ***Captive-NTFS*** – amelyek teljes mértékben támogatják az ***NTFS*** fájlrendszerek ***Linux*** alóli kezelését. Az adatokhoz való hozzáférés ráadásul ebben az esetben sem korlátozódik a helyi gépre. ***Samba*** segítségével a felhasználók a hálózat más, ***Windows*** futtató gépein megosztott könyvtáraikat is elérhetik. Bátran kijelenthetjük tehát, hogy ha valaki ***Windowsról*** szeretne ***Linuxra*** váltani, annak semmiféle problémára nem kell felkészülnie. Minden működni fog a migráció után is.
Ugyanakkor megfigyelhető egy új trend is. Ahogy a ***Linux*** egyre elterjedtebbé válik az asztali gépeken, úgy egyre több olyan felhasználóval találkozhatunk, akik bizonyos feladatokra ***Linuxot*** használnak ugyan, utána viszont újraindítják a gépüket, és ***Windows*** alatt folytatják a munkát. Erre általában azért van szükség, mert bizonyos népszerű alkalmazásoknak továbbra sincs ***Linux*** alatt futó változata, vagy megfelelő képességekkel rendelkező linuxos helyettesítője. Ilyenek például a modernebb játékok, vagy az ***Adobe Photoshop***, de más, speciális területekről is említhetnénk olyan példákat, amikor a nyílt forrású megfelelő egyszerűen nem létezik. Ha tehát a felhasználó kényszerűségből oda-vissza váltogat a két operációs rendszer között, akkor nyilván lesz olyan pillanat is, amikor ***Windows*** alól szeretné elérni a ***Linux*** partíciók tartalmát. Ezzel csak egy a gond: a ***Windows*** hivatalosan semmiféle támogatást nem nyújt a ***Linux*** fájlrendszerinek kezeléséhez. A több operációs

Rövid bevezetés a fájlrendszerek elméletébe

Olvasóink közül talán nem pontosan tudják, mi is az a fájlrendszer.
A fájlrendszer gyakorlatilag egy tárolási rendszer vagy módszer, amely meghatározza, miként kell a fájlokat a lemezre írni, illetve onnan visszaolvasni. Mármost ebből elég egyenes következik egy logikus kérdés: „Miért van annyiféle fájlrendszer? Miért nem tudják az emberek egyszer és mindenkorra eldönteni, hogy melyik a legjobb tárolási és visszaolvasási rendszer, és miért nem használja mindenki, minden operációs rendszer alatt azt?" Bár a kérdés a maga módján jogos, az igazsághoz az is hozzátartozik, hogy a különböző operációs rendszereknek eltérő a „célközönsége", és az eltérő igényekhez eltérő fájlrendszerek igazodnak jobban. A ***Windows NT***-t például üzleti felhasználóknak szánták, ezért ott nagy jelentősége volt a fájlrendszer biztonsági szolgáltatásainak. Így született meg az ***NTFS***. A ***Windows 9x*** ezzel szemben a „közönséges felhasználóknak" készült, akiknek nem a biztonság, hanem a teljesítmény volt a fontosabb. Éppen ezért ez az operációs rendszer ***FAT16/32*** fájlrendszerrel működik. A ***Windows XP*** azért támogatja mindkettőt, mert mindkét célcsoport számára megfelelő szolgáltatásokat kell nyújtania. Hasonlóan ***Linux*** alatt is többféle fájlrendszert használhatunk. A de facto szabvány itt az ***ext2/3*** fájlrendszer használata, de léteznek teljesen speciális célokra tervezett adattárolási rendszerek is. Ilyen például a nagy számítási feladatokhoz használt ***XFS***, vagy a nagyobb hibatűrést biztosító ***ext3*** és ***ReiserFS***.
Ha egy operációs rendszerbe be szeretnénk építeni egy adott fájlrendszer „támogatását", akkor létre kell hoznunk benne azokat az adatszerkezeteket, amelyek a kérdéses tárolási rendszer kezeléséhez szükségesek. Ilyenkor természetesen sokkal nehezebb dolgunk van egy zárt szabványokra épülő fájlrendszerrel – mint amilyen például az ***NTFS*** – hiszen ismeretlen, vagy kevéssé ismert a lemezen elhelyezkedő adatok szerkezete, a használt titkosítási algoritmusok, és az egyéb, ezekhez hasonló tényezők. Pont ezzel magyarázható, hogy az ***NTFS Linux*** alóli támogatása miért jelent olyan nagy kihívást a fejlesztőknek.



1. ábra Az Ext2fsd segítségével az ext2 fájlrendszerek ugyanolyannak tűnnek Windows alatt, mint bármely más partíció

rendszert telepítőknek tehát mindenképpen speciális eszközöket kell használniuk *Windows* alatt, ha boldogulni akarnak.

Linux partíciók kezelésére szolgáló eszközök Windows alatt

Amint korábban is említette, linuxos fájlrendszerekhez a *Windows* natív támogatást egyáltalán nem tartalmaz. Persze azért nincs minden veszve. A nyílt forrású közösség felvette a kesztyűt, és megalkotott néhány egészen kiváló eszközt, amelyekkel a probléma megoldható. Ebben a cikkben elsősorban az *LTOOLS* csomagra fogok koncentrálni. Ez egy egészen kifinomult eszköz, amelyhez többféle interfész is tartozik. Ezekkel a felhasználó több különböző, *Linux* alatt használatos fájlrendszertípust is kezelhet. Mielőtt azonban belekezdenék ennek a csomagnak a tárgyalásába, fussunk gyorsan végig az egyéb lehetőségeken is, amelyekkel a probléma szintén megoldható.

Ext2fsd

Az *Ext2fsd* az egyik legrégibb olyan projekt, amely céljául a fenti feladat megoldását tűzte ki. Segítségével *ext2* fájlrendszerek tartalmához férhetünk hozzá *Windows* alól. A csomagot a ➲ *http://sourceforge.net/projcts/ext2fsd* címről tölthetjük le. A program fájlrendszer-meghajtóként, és nem közönséges programként települ. Ennek a módszernek az az előnye, hogy az *Ext2fsd* teljesen transzparens módon teszi elérhetővé az *ext2* fájlrendszereket *Windows* alól, vagyis első közelítésben semmi különbséget nem látunk valódi windowsos és linuxos partíció között. Az *Ext2fsd* nem csak olvasni, hanem írni is tudja az *ext2* és vele rokon fájlrendszereket. A *Linux* első de facto szabványos fájlrendszere ugyan az *ext2* volt, mára azonban számos egyéb tárolási rendszer is megjelent. Ilyen például az *ext3* és az újabban felbukkant *ext4*, amelyek azonban mindketten visszafelé kompatibilisek az *ext2*-vel. Ennek megfelelően az *Ext2fsd*-vel ezek is kezelhetők, bár ami azt ext4-et illeti, a helyzet egyelőre kissé bizonytalan, hiszen egyelőre csak kísérleti jelleggel vették fel a kernelbe. Mindazonáltal az *Ext2fsd*-vel való kompatibilitás azért valószínű, mert amikor *Windows* alól kezelünk egy *ext3*/*4* partíciót, akkor a fájlrendszernek csak azokat a funkcióit használjuk, amelyek már az *ext2*-ben is megvoltak, vagyis elvileg nem lehet semmi gond. Ugyanakkor persze az is igaz, hogy az olyan kiterjesztett szolgáltatások, mint a naplózás így nem használhatók.

rfstool

Az utóbbi időben linuxos körökben egyre nagyobb népszerűségnek örvend a *ReiserFS* fájlrendszer. Ennek elsősorban az általa biztosított nagyfokú hibatűrés az oka. Ehhez is készült egy *Windows* alatt használható csomag, az *rfstool*, amely azonban csak olvasási műveleteket tud végezni, és a fejlesztők a hivatalos webhely tanúsága szerint nem is tervezik az írási műveletek megvalósítását. A csomagot a ➲ *http://fresmeat.net/projects/rfstool* címről szerezhetjük be.

LTOOLS

És ezzel meg is érkeztünk a cikk főtémájához, az *LTOOLS* csomaghoz. Az *Ext2fsd*-től és az *rfstool*-tól eltérően ez a fájlrendszereknek nem csupán egyetlen speciális osztályát tudja kezelni, sokkal általánosabb annál. Az *LTOOLS* egyszerre támogatja az *ext2*, *ext3* és *ReiserFS* fájlrendszereket. A csomag tulajdonképpen parancssori eszközök gyűjteménye, de tartozik hozzá grafikus felhasználói felület (*GUI*), sőt webes felület is. Telepíthető gyakorlatilag valamennyi *DOS* és *Windows* változat alá (*XP*, *2000*, *NT*, *ME*, *9x* és *3.x*). Ráadásul nem csak a helyi gépen levő fájlrendszereket tudja kezelni, hanem a hálózati megosztásokat is. Akár *DOS*, akár valamilyen *Windows* alatt dolgozunk is tehát, bizton számíthatunk az *LTOOLS* eszközeire, ha *Linux* fájlrendszeren levő adatokhoz kell hozzáférni.
Az *LTOOLS* amúgy akkor is jól jöhet, ha úgy kell megjavítanunk egy *Linux* rendszert, hogy éppen nincs kéznél egy megfelelő *Live CD*.
Amint az imént említettem, az *LTOOLS* csomag két különböző grafikus felületet is tartalmaz, amelyekkel a parancssori eszközök teljes funkciókészletét kihasználhatjuk. Van hozzá egy *LTOOLSgui* nevű, *Java* alapú interfész, illetve létezik egy *Microsoft .NET* alapú *LTOOLSnet* nevű alkalmazás is. Mindkettővel kezelhetünk helyi és távoli fájlrendszereket egyaránt.
Aki még ezzel sem elégedett, például mert kizárólag szabad szoftvereket szeretne használni (sem a *Java*, sem a *Microsoft .NET* nem az), az akár egy közönséges webböngészőn keresztül is elérheti az *LTOOLS* szolgáltatásait. Ez utóbbi azért lehetséges, mert a csomag tartalmaz egy *LREADsrv* nevű alkalmazást, ami nem más, mint egy egyszerű webkiszolgáló. Ez a *Linux* fájlrendszereit bármely webböngésző, akár az *Explorer* számára is láthatóvá

tudja tenni. Az *LREADsrv* segítségével ráadásul nem csak a *Linux*, hanem a *DOS*/*Windows* fájlrendszereit is elérhetővé tehetjük a hálózaton keresztül.

Az LTOOLS telepítése

Az *LTOOSL* egy szabványos *Windows* telepítőcsomag formájában érkezik, ami kifejezetten időszerűnek nevezhető. A telepítés folyamatában sincs semmi említésre méltó. A telepítő létrehoz a *Start* menüben egy *LTOOLS* nevű pontot, amely alatt aztán megtaláljuk az összes említett interfészt, amelyeken keresztül máris hozzáférhetünk linuxos fájlrendszereink tartalmához.
Bár – amint említettem – maga az *LTOOLS* támogatja ugyan az összes *Windows* változatot, nem minden szolgáltatása illetve interfésze működőképes valamennyi verzió alatt. A csomag eleve két különböző konzolos változatot tartalmaz a *Win9x*/*ME* illetve a *Windows NT*/*XP* rendszerekhez.

Az LTOOLS szolgáltatásainak áttekintése – A parancssori eszközök

A parancssori eszközökkel a legalapvetőbb fájl- és könyvtárkezelési feladatokat végezhetjük el. Mint említettem, a parancsok a linuxos fájlrendszerek írására és olvasására egyaránt használhatók. Minden parancshoz három fájl tartozik. Nézzünk talán egy példát. Az

```
ldir.bat
```

parancs például listázza a könyvtárak tartalmát, ez azonban – mint nevéből is látszik – nem egy program, hanem egy szkript. Ez a szkript – attól függően, hogy milyen operációs rendszer alatt futtatjuk – maga is meghív az *LTOOLS* csomag részét képező programokat. Jelen esetben ténylegesen vagy az *LdirDOS.exe*, vagy az *ldirNT.exe* fog lefutni. Az első a *Win9x*/*ME* rendszerekre vonatkozik, míg a második a *Windows NT*/*XP* operációs rendszerekhez való.
A legtöbb *LTOOLS* parancs szintaxisa teljesen logikus. A partíciókat például ugyanúgy kell megneveznünk, mint *Linux* alatt. Ha mondjuk a *vars.inc* nevű fájlt akarjuk átmásolni a */root* fájlrendszerről (ami fizikailag mondjuk a */dev*/*hda2* partíciót jelenti) a *C:* meghajtóra, akkor a következőt kell tennünk:

```
lread.bat --s/dev/hda2 /root/
➥vars.inc C:\vars.inc
```

Teljesen hasonlóan ha ezt a fájlt vissza akarjuk írni a *Linux* fájlrendszerére, a következő parancsot használhatjuk:

```
lwrite.bat --s/dev/hda2
➥C:\vars.inc /root/vars.inc
```

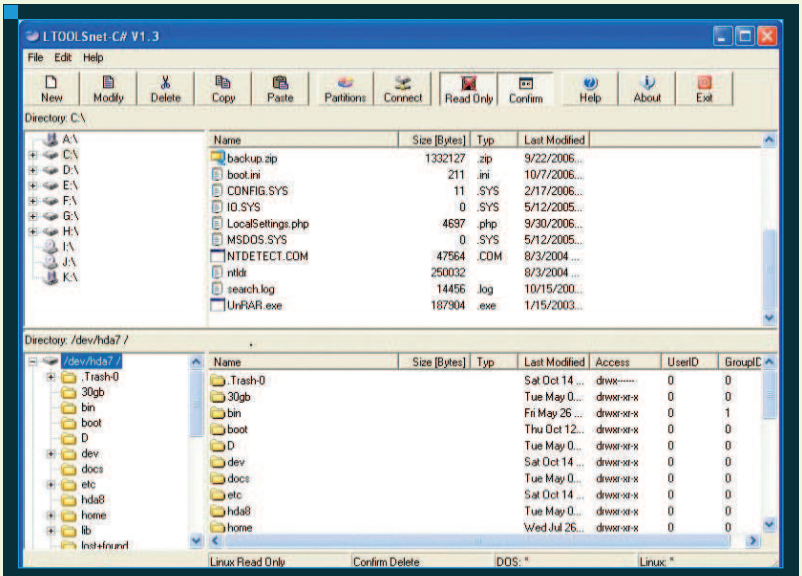
Az *LTOOLS* csomag parancsairól az *1. Táblázatban* találunk összefoglalót. A fent felsorolt eszközök a *ReiserFS* fájlrendszereket nem támogatják. Éppen ezért az *LTOOLS* csomag tartalmazza az *rfstools* programot is, amit kifejezetten erre a célra írtak. Ahhoz, hogy *Windows NT*/*2000*/*XP* vagy *UNIX*/*Linux* alatt a merevlemez tartalmát olvashassuk, rendszergazdai jogosultságokra van szükségünk. Ha az *LTOOLS* parancsait egy nem adminisztrátori felhasználói fiókból futtatjuk, nem fogunk tudni hozzáférni a lemez tartalmához.
Az *LTOOLS* emellett nem foglalkozik a *Linux* alatt érvényes „tulajdonviszonyokkal". Ez egyben azt is jelenti, hogy ha egy felhasználó képes hozzáférni a gyökér fájlrendszerhez, akkor azon azt alakít át, amit akar – beleértve persze a */etc*/*passwd* tartalmát is.

1. táblázat *Az LTOOLS parancsai*

`lread.bat`	Fájlok olvasása és másolása Linux partícióról DOS partícióra.Példa: `lread.bat --s/dev/hda2 /root/vars.inc C:\vars.inc`
`lwrite.bat`	Fájlok másolása DOS-ról Linuxra. Példa: `lwrite.bat --s/dev/hda2 C:\vars.inc /root/vars.inc`
`ldel.bat`	Fájlok vagy üres könyvtárak törlése Linux fájlrendszerről. Hatása tulajdonképpen azonos az `rm -f` illetve rmdir parancsokéval.Példa: `ldel.bat /root/vars.inc`
`lchange.bat`	Fájlok tulajdonságainak és tulajdonosának megváltoztatása Linux alatt (a `chmod` parancs megfelelője).Példa: `lchange.bat --s/dev/hda2 754 /root/vars.inc`
`lren.bat`	Linux fájlrendszeren található fájlok átnevezése (az mv parancs analógja).Példa: `lren.bat --s/devhda2 /root/vars.inc /root/var2.inc`
`lmkdir.bat`	Könyvtár létrehozása a Linux partíción (az `mkdir` megfelelője). Példa: `lmkdir.bat --s/dev/hda2 /root/newdir`
`lln.bat`	Szimbolikus lánc létrehozása (az `ln` megfelelője). Példa: `lln.bat --s/dev/hda8 /root/link /root/vars.inc`
`lcd.bat`	Könyvtárváltás (a cd megfelelője). Példa: `lcd.bat /home/`
`ldrive.bat`	Az alapértelmezett Linux lemezmeghajtó kiválasztása. Példa: `ldrive /dev/hda8`

Az LTOOLS .NET keretrendszerre támaszkodó ügyfélprogramja

A *.NET* keretrendszerre támaszkodó ügyfélprogram az *LTOOLS* szolgáltatásokban leginkább bővelkedő része. Futtatásához természetesen le kell töltenünk a *Microsoft .NET* keretrendszert a ➲ *http://www.microsoft.com/downloads* webhelyről.
Az ügyfél lehetővé teszi valamennyi *Windows* és *Linux* fájlrendszer böngészését. Szabadon átvihetünk köztük fájlokat, törölhetjük, szerkeszthetjük, vagy bármilyen más módon megváltoztathatjuk azokat. Arra is lehetőségünk van, hogy távoli fájlrendszereket becsatoljunk, és azok tartalmát kezeljük. Én magam ezt a lehetőséget



2. ábra A C# LTOOLS fájlkezelő

rendkívül hasznosnak találtam, amikor a webkiszolgálómmal volt némi problémám. *Windows* alól is minden további nélkül be tudtam csatolni a webes dokumentumokat tartalmazó fájlrendszert, és elvégezhettem a szükséges változtatásokat.

Az LTOOLS Java keretrendszerre támaszkodó ügyfélprogramja

Aki *Windows95/98/ME* operációs rendszert használ, az nem tudja telepíteni a *Microsoft .NET* keretrendszerét, mert annak nem készült ezekhez a rendszerekhez megfelelő változata. Ilyenkor jön jól a Java felület. Persze ehhez is le kell töltenünk egy *Java* futtatókörnyezetet (*Java Runtime Environment*), amit a ➲ *java.sun.com* címen találunk meg. A *Java* interfész funkciói mindenben megegyeznek a *.NET* kliens szolgáltatásaival.

Az LTOOLS web-alapú felülete

Tapasztalataim szerint az *LTOOLS* leghasznosabb kezelőfelülete a hozzá mellékelt webes szolgáltatás. Az *LTOOLS* csomag tartalmaz egy *LREADsrv.exe* nevű beépített webkiszolgálót, amit a felhasználó elindíthat, és segítségével egy közönséges webböngészőn keresztül kezelheti a fájlrendszerek tartalmát. Ennek óriási hasznát vehetjük akkor is, ha távoli felhasználókkal szeretnénk fájlokat megosztani. Ugyanakkor biztonsági szempontból nem javasolt az *LREADerv.exe* futtatása egy globálisan hozzáférhető kiszolgálón. Ilyenkor ugyanis esetleg olyanok is hozzáférhetnek a fájlrendszer tartalmához, akikkel nem akarunk adatokat cserélni. Az *LTOOLS* saját kiszolgálóját tehát célszerűen csak olyan környezetben futtassuk, ahol egyéb módon biztosított, hogy csak a megfelelő felhasználók férhetnek hozzá a megosztott tartalmakhoz. Ilyen környezet például egy virtuális magánhálózat. Az *LREADsrv.exe*-nek amúgy vannak eleve ismert hibái, amelyeket a fejlesztők a későbbi változatokban bizonyára ki fognak javítani. Ennek megfelelően a program a fejlesztők szerint is alfa állapotú, és vannak bizonyos képességbeli korlátai is. Ezek között egyesek éppen a *HTTP 1.1*-gyes webböngészőkkel való kompatibilitást érintik, amilyen az *Internet Explorer* is. Ilyenkor a kiszolgáló válaszideje megnövekszik. Egy másik, adott esetben kifejezetten zavaró korlát abból fakad, hogy az *LREADsrv* program jelen változatában egyszálú, ami azt jelenti, hogy ha egyszerre többen használják, akkor valamennyi változtatás hatása globális. Így versenyhelyzetek alakulhatnak ki, illetve bizonyos szituációkban elveszhetnek módosítások. Ráadásul az *LREADsrv* hibaellenőrzése is meglehetősen gyenge. A legtöbb felhasználói bemenetet (például a fájlneveket és ehhez hasonló információkat) egyáltalán nem ellenőrzi. Ha tehát a felhasználó elgépeli egy fájl nevét, vagy tévesen adja meg egy partíció nevét, a webkiszolgáló instabillá válhat. Az egyetlen jó hír, hogy ez adatvesztést biztosan nem fog okozni.

Összefoglalás

Összességében elmondható, hogy a *Linux* felhasználóknak egyre jobb eszközök állnak rendelkezésére a *Windows* fájlrendszerek kezeléséhez. Szerencsére létezik az ellenkező irányú hozzáférés is, hiszen mint láttuk, az *LTOOLS* számos különféle *Linux* fájlrendszerhez ad hozzáférést *Windows* aló, a legkülönfélébb felhasználói felületeken keresztül. Ugyanakkor az is igaz, hogy maradt még tér a továbbfejlődéshez is, hiszen bizonyos nem is olyan ritkán használt fájlrendszer-típusok *Windows* alatti támogatása ma még meglehetősen hiányos, vagy egyáltalán nem létezik. Ilyen például az *XFS*. Az olyan meghajtók fejlesztése, amelyek a *Linux* fájlrendszerek olyan speciális szolgáltatásait mint például az *ext3* vagy a *ReiserFS* naplózása is képesek kihasználni, egyelőre gyerekcipőben szintén jár. A *Linux* fájlrendszerek és a *Windows* operációs rendszer közötti átjárhatóság megvalósítása ugyanakkor rendkívül fontos feladat, hiszen ennek hiánya sokakat megakadályozhat a nyílt forrású rendszerekre való váltásban. Összességében tehát a nyílt forrású közösségnek tudomásul kell vennie, hogy a *Windows* egyelőre dominánsnak tekinthető az asztali gépeken, és a két rendszer közötti kölcsönös átjárhatóság biztosítása elemi érdekünk.

Linux Journal 2007., 154. szám

Irfan Habib egyetemi hallgató. Informatikát tanul a Pakisztáni Nemzeti Tudományos és Műszaki egyetemen (National University of Sciences and Technology, Pakistan). Évek óta nagyon komolyan foglalkoztatja a szabad és nyílt forrású szoftverek világa. Gyakran dolgozik heterogén számítógépes rendszerekkel. Éppen innen ered a különböző operációs rendszerek és fájlrendszerek közti átjárhatósággal kapcsolatos érdeklődése.

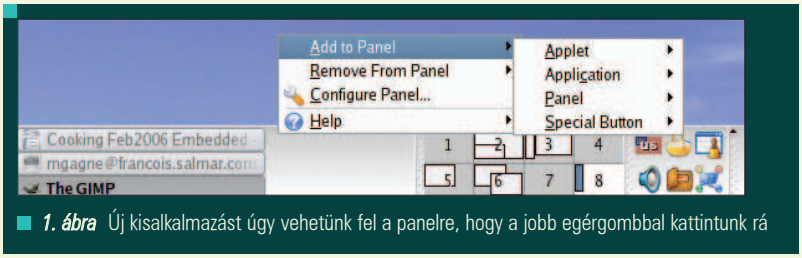
Icipici alkalmazások

Nos, kedves Francois, látom rengeteg apró segédeszköz van állandóan a kezed ügyében, de nem gondolod, hogy egy kicsit túlzásba viszed a dolgot? Természetesen örülök, hogy ilyen gyorsan rá tudtál hangolódni a témára, de ha a mütyürkéktől már egy webböngészőt vagy egy szövegszerkesztőt se tudsz kényelmesen elindítani úgy, hogy láss is valamit a képernyőn, akkor ugye átestünk a ló másik oldalára, nem gondolod? A kisalkalmazások lényege, hogy kicsik. Na de ha ennyit futtatsz belőlük, akkor nem marad helyed az érdemi munkára.

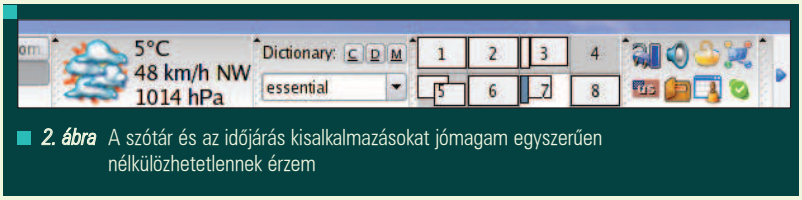
■ Na, nem baj ***Francois***, ezt majd később megbeszéljük. Ami azt illeti, az olyan beágyazott alkalmazások, mint amiket te is használsz igazán jól illenek a mai menühöz. Na de most gyorsan el kell készülnünk, mert mindjárt itt vannak a vendégeink. Hoppá, már el is késtünk ***Francois***, hiszen meg is jöttek. Szeretettel köszöntök mindenkit ***Marcel*** vendéglőjében, a kivételesen finom borok és a legjobb ***Linux*** alkalmazások hazájában. Kérem foglaljanak helyet és helyezzék magukat kényelembe. ***Francois***, te meg indulj a pincébe, és hozz föl néhány palackkal az 1999-es ***Toszkánai Brunello di Montacinoból***. Ott találod a délkeleti szárnyban közvetlenül a ***Chiantik*** után.

Amíg hűséges pincérem a bort keresi, had meséljek egy kicsit a mai összeállításról. Korábban meséltem már róla, hogy mi mindenre jók azok az apró alkalmazások, amelyeket a képernyő alján a panelbe építve találunk. Ezeket kisalkalmazásoknak (***applet***) hívják és a legtöbbjüket úgy tervezték meg, hogy kényelmesen elférjen a tálcán, miközben hasznos szolgáltatásokat nyújt. Ami azt illeti, sokan úgy hiszik, hogy ezek alkotják a tálcát, pedig az maga is egy önálló kisalkalmazás.

De ha már itt tartunk, akkor érdemes azt is hozzátenni, hogy a feladatváltó, a menü, az óra és maga a rendszertálca is ilyen kisalkalmazások.



■ ***1. ábra*** Új kisalkalmazást úgy vehetünk fel a panelre, hogy a jobb egérgombbal kattintunk rá



■ ***2. ábra*** A szótár és az időjárás kisalkalmazásokat jómagam egyszerűen nélkülözhetetlennek érzem

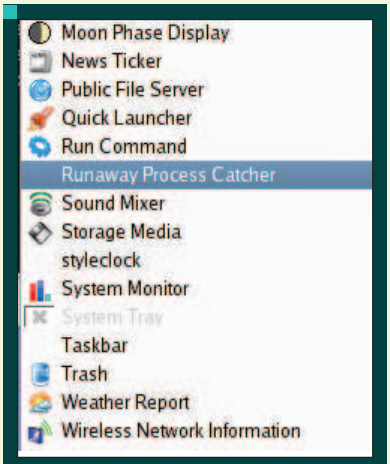
Mielőtt továbblépnénk, azt hiszem nem árt megjegyezni, hogy a ***KDE*** felhasználóknak rendszerint egy panel van a képernyőjén, mégpedig annak az alján, míg a ***GNOME*** híveinek rögtön kettő is jutott belőle, melyek közül egy alul, egy meg felül található.

Ma mindkét asztali környezetről esik majd szó, de előbb a ***KDE***-t kerítjük sorra. A lényeg, hogy az eddig felsorolt kisalkalmazásokon kívül általában még rengeteg van telepítve a gépünkre, csak fel kell fedezni őket.

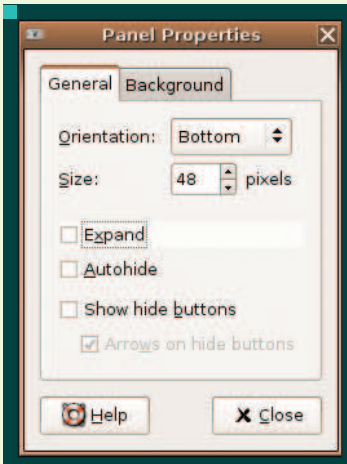
Ha új kisalkalmazást akarunk felvenni a ***KDE*** paneljére, a következőképpen kell eljárnunk. Kattintsunk a jobb egérgombbal a panelre, a felbukkanó helyi menüből válasszuk a ***„Hozzáadás a panelhez” (Add to Panel)*** pontot, majd abból ***„Kisalkalmazás” (Applet)*** alpontot (*1. ábra*).

Erre a kisalkalmazások egy meglehetősen nagy gyűjteménye jelenik meg. Ezek közül választhatjuk ki, melyeket szeretnénk viszontlátni a képernyőnkön. Az igazság az, hogy bármelyikbe könnyen beleszerethet az ember, de én most a számomra legkedvesebb darabra, a szótárra fogok koncentrálni (*2. ábra*). Valahányszor egy új rendszert telepítek az első dolgom az, hogy ezt kitegyem a tálcára. Egyszerűen el se tudom már képzelni a képernyőmet nélküle, hiszen döbbenetes hasznos kis programról van szó.

Az ember csak begépel egy szót, az



3. ábra Ez csupán töredéke annak a rengeteg kisalkalmazásnak, amelyek a KDE részeként érkeznek



4. ábra Ahhoz, hogy bizonyos kisalkalmazások tartalmát láthatóvá tegyük, meg kell növelnünk a panel méretét

alkalmazás pedig automatikusan rákeres a meghatározására több különböző, az interneten elérhető szótárban. Bár nekem ez az egy az abszolút kedvencem, azért számos más, hasznos is akad ebben a gyűjteményben. Sok más embertársamhoz hasonlóan engem sem „hagy hidegen" az időjárás. Így aztán az erről tájékoztató kisalkalmazás is ott virít a képernyőmön. (Ha valakit érdekelne, amikor ezt írom az idő kissé felhős, a hőmérséklet pedig 5 fok.) Kattintsunk a kisalkalmazás ikonjára a jobb egérgombbal, majd válasszuk a ***„KWeather beállítása" (Configure KWeather)*** pontot. Válasszuk ki a listából azt a várost, ahol élünk, és ezzel készen is vagyunk. Szintén hasznos eszköz a színérzékelő (***color picker***), amivel egy a képernyőn megjelenített dolog színkódját kérdezhetjük le. Ez a dolog természetesen bármi lehet, tehát akár egy weblap tartalma is. Van aztán egy olyan kisalkalmazás, amely a hold fázisait tudja megjeleníteni. Mit mondjak, mámorító érzés tudni, hogy már csak hatot kell aludni a legközelebbi holdtöltéig. A „kockák" kipróbálhatják a bináris órát. Nekik való. De hogy valami kifejezetten értelmesről is essen szó, van rendszermonitorozó program is a gyűjteményben. Szóval akit érdekelnek a kisalkalmazások, az kénytelen lesz némi időt szentelni a kipróbálásukra, ugyanis rengeteg van belőlük (*3. ábra*). Aztán ha az ember már elég sokat kísérletezett, akkor eljön az a pillanat, amikor kezdi úgy érezni, hogy már nem fér el a saját képernyőjén. Nemhogy alkalmazásoknak, hanem már az újabb ikonoknak sincs hely a tálcán. Mi a teendő ilyenkor? Az egyik megoldás természetesen azoknak a kisalkalmazásoknak az eltávolítása, amelyeket valójában nem is használunk. Ehhez kattintsunk a panelen a jobb egérgombbal, a helyi menüből válasszuk az ***„Eltávolítás" (Remove)*** menüpontot, majd a ***„Kisalkalmazás" (Applet)*** alpontot.

Persze megeshet, hogy úgy gondoljuk, mindarra, amit kiraktunk a tálcára valóban bármikor szükségünk lehet. Ilyenkor nem az eltávolítás a megoldás, hanem egy gyermekpanel létrehozása. Éppen erről beszélgettünk ***Francois***-val, amikor Önök megérkeztek. Neki is nagyon sok gyermekpanelje van kirakva, és ez finoman szólva a rendetlenség látszatát kelti. Nincs kétségem afelől, hogy a jelenlevők valamennyien rendelkeznek a szükséges mértékű önuralommal, ezért megmutatom, hogyan kell ezt a dolgot csinálni. Kattintsunk a jobb egérgombbal a panelen, válasszuk a már ismert ***„Hozzáadás a panelhez" (Add to Panel)*** pontot, majd a ***Panel*** alpontot. Ebben az almenüben több különböző lehetőséget találunk, melyek között akad egy Panel nevezetű is. Kattintsunk hát erre, mire egy teljesen üres, szürke panel jelenik meg közvetlenül a mostani fölött. Most aztán nekiláthatunk ezt is benépesíteni apró ikonokkal, de ami azt illeti mást is kezdhetünk vele. Vannak aztán más műveletek is, amelyek egy ilyen szép új panellel végezhetők. Ezekről akarok most néhány szót ejteni.

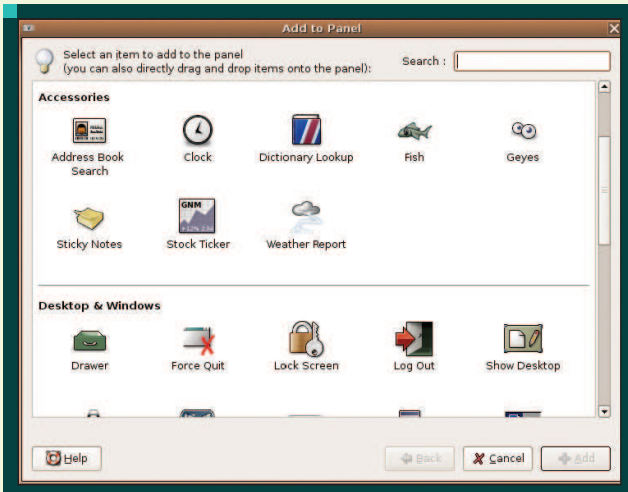
Először is meglehet, hogy nem igazán vagyunk megelégedve az új panel elhelyezkedésével. Ha így van, akkor egyszerűen a bal egérgomb nyomva tartása mellett húzzuk arra a helyre, ahol lennie kell. Ezzel a mozdulattal a képernyő bármelyik oldalsó élére, vagy a tetejére is áthelyezhetjük. Ugyanezt úgy is megtehetjük, hogy a panelen a jobb egérgombbal kattintva a ***„Beállítás" (Configure)*** menüpontot választjuk. Ez azért jó, mert itt rögtön a panel méretét is átállíthatjuk, ha szükséges.

A kisalkalmazásokkal kapcsolatos őrületből természetesen a ***GNOME*** rendszer használói sem maradhatnak ki. Amint azt korábban említettem, itt is létezik a Panel megfelelője, és ezen is lehetnek is apró, hasznos programok. Bár ***GNOME*** esetében az alsó és felső panel egyaránt a rendelkezésünkre áll, én a továbbiakban inkább csak az alsóra fogok koncentrálni.

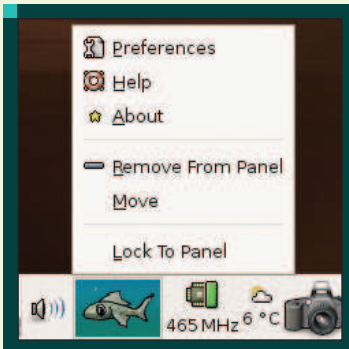
A panel alapértelmezett kinézete és mérete itt némiképp változik attól függően, hogy ki milyen terjesztést használ. Az én 2.12-es verziószámot viselő ***GNOME***-omnak például 24 pixel széles a panelje. Ez azért érdekes, mert attól függően, hogy milyen kisalkalmazást akarunk rajta elhelyezni, előfordulhat, hogy annak a tartalma nem igazán lesz látható. Ha tehát sikerült beleütköznünk ebbe a problémába, akkor kattintsunk a panel egy üres területén a jobb egérgombbal és válasszuk a ***„Tulajdonságok" (Properties)*** menüpontot. Ennek hatására megjelenik a ***„Panel Tulajdonságai" (Panel Properties)*** nevű dialógusablak (*4. ábra*).

Tapasztalataim alapján – na meg a képernyőm méreteiből adódóan – a 48 pixelnyi szélesség éppen megfelel a célnak. Persze az, hogy ki mekkorára veszi ezt az értéket, az teljesen az elhatározásaitól és egyéni igényeitől függ, meg persze attól, hogy mennyi bort ivott előtte. Na jó, csak viccelek, de azért ez az egész igazán kiváló ötlet, nemde? Apropó ***Francois***, kérlek tölts a vendégeinknek!

Most, hogy magát a panelt méretre szabtuk, ideje, hogy találjunk rá néhány szép kisalkalmazást. Kattintsunk tehát rajta a jobb egérgombbal és



5. ábra Kisalkalmazás felvétele a GNOME paneljére



6. ábra Valamennyi kisalkalmazást beállíthatunk, áthelyezhetjük vagy el is távolíthatjuk őket

szemléljük meg a felbukkanó menüt. Válasszuk a ***„Hozzáadás a panelhez" (Add to panel)*** pontot, melynek hatására a használható kisalkalmazások listája jelenik meg (*5. ábra*). Lehet, hogy ez a lista egyszerűen csak az alkalmazások felsorolása, némi leírással kiegészítve, de az is lehet, hogy mindenféle kategóriák vannak benne. Akárhogy is, válasszunk ki valami nekünk tetszőt belőle, majd kattintsunk a ***„Hozzáadás" (Add)*** gombra. Jómagam elsőként a ***Fish*** nevűt próbáltam ki. Ez egy kis animált halacska, aki úszkálás közben mindenféle véletlenszerűen kiválasztott bölcsességeket meg egyéb aranyköpéseket jelenít meg. Amúgy a ***fortune*** parancsra támaszkodik, tehát alapjában véve nem új találmány. És igen kérem, a hal neve ***Wanda***. Nem, hölgyeim és uraim, nem én találtam ki. Csak pár másodperc, és a kisalkalmazás máris megjelent a panelen.

Amikor egy ilyen program elindul, általában nem azon a helyen bukkan föl, ahova én helyezni szeretném. Szerencsére az áthelyezés sem különösebben bonyolult. Egyszerűen kattintsunk a jobb gombbal a képére, mire egy helyi menü jelenik meg (*6. ábra*). Az ebben található egyik pont neve ***„Mozgatás" (Move)***. Kattintsunk tehát ide, mire egy kis kezet formázó ikon jelenik meg, amivel valóban mozgatni lehet a kisalkalmazás ikonját a panelen belül. Vigyük tehát oda, ahova szerintünk való. Ezzel a módszerrel amúgy akár egy másik panelre is áthelyezhetjük (erről egy pillanat múlva még bővebben is szólok).
Most pedig vessünk még egy pillantást arra a bizonyos menüre, és máris láthatjuk, hogy van benne még néhány hasznos funkció. Például minden alkalmazásnak lehet saját beállításkészlete. Az általam telepített ***Fish*** alkalmazáshoz például jár számos kiegészítő grafikai elem, arra az esetre, ha valakinek nem tetszik a halacska. Ilyenkor kattintsunk a ***„Testreszabás" (Preference)*** pontra, és válasszuk ki a nekünk megfelelő megjelenést.
Aztán megeshet, hogy végül mégsem akarjuk ezt a kisalkalmazást a panelünkön tudni. Ezt az ***„Eltávolítás" (Remove)*** ponttal oldhatjuk meg.
Aztán ha elszabadult a kisalkalmazásmánia, hamar kifuthatunk a szabad helyből. Igen, ez az a pillanat, amikor szükségünk lesz egy új panelre, akárcsak korábban a ***KDE***-nél. Kattintsuk tehát a jobb gombbal a panel egy szabad területén, a helyi menüből pedig válasszuk az ***„Új panel" (New Panel)*** pontot. Egy üres, szürke panel jelenik meg, amit azonnal a képernyő megfelelő helyére vonszolhatunk. Akárcsak az eredeti panelnél, itt is rendelkezésünkre áll a ***„Tulajdonságok" (Properties)*** ablak, amelyben beállíthatjuk az új példány méretét.
Ami a ***GNOME*** kisalkalmazásait illeti, meglehetősen hasonló készletet kapunk, mint amiről korábban a ***KDE*** kapcsán volt szó, bár némelyik funkcionalitása eltérő lehet. A szótár és az időjárás természetesen most is kötelező jelleggel kikerül a képernyőre.
A ***Fish*** és az egérmutatót követő szemek csak a móka kedvéért futnak.
A képlopó kisalkalmazás (***Take Screenshot***) ugyanakkor különösen hasznos lehet dokumentációk készítése során. Mivel jómagam több rendszeren szoktam egyszerre dolgozni, nálam a ***Terminal Server Client*** és a ***Connect to Server*** nevű programok is mindennapi használatban vannak, tehát a tálcán tartom az ikonjukat.
Nos, kedves vendégeim, a falióra szerint megint eltelt egy napunk. Bárcsak olyan könnyű lenne néhány órányi idővel kiegészíteni a napjainkat, mint amilyen könnyen kisalkalmazások ikonjaival „felszerszámozni" a tálcát. Ez sajnos nem lehetséges, de kárpótlásul itt van nekünk hűséges pincérünk, ***Francois***, aki készen áll, hogy még egyszer utoljára teletöltse mindenki poharát. Emeljük tehát poharainkat és igyunk egymás egészségére!

Linux Journal 2006., 142. szám

Marcel Gagne díjnyertes író, aki Mississauga-ban, Ontario államban él. Ő a szerzője a „Válts Linuxra – Búcsú a kékhaláltól" (Kiskapu) című könyvnek is (Moving to Linux: Kiss The Blue Screen of Death Goodbye! 2nd edition, Addison-Wesley, ISBN 0-321-35640-3). Marcel gyakorta bukkan fel a tévében is mint a Linuxban járatos szaktanácsadó. Aki levelet szeretne küldeni neki, az az *mgagne@salmar.com* címre írhat, webhelye pedig a *www.marcelgagne.com* címen üzemel.

LightScribe – Linux alatt

A 2004 elején „feltalált” LightScribe Direct Disk Labelling néven illetett technológia méltán lett egy év alatt népszerű a CD vagy DVD írással kapcsolatba kerülő emberek körében. A LightScribe olyan egyszerű ötlet, mint a faék, mégis sok-sok év kellett ahhoz, hogy a nagy gyártók „feltalálják” és szabványt képezzenek rá.

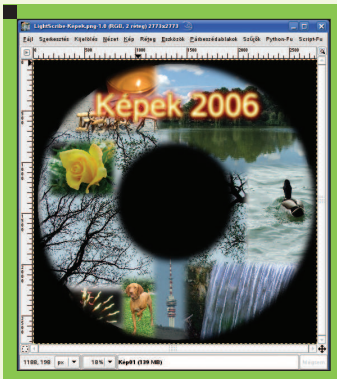
Az írási folyamatot leegyszerűsítve, a *CD* és a *DVD* a lemezre az író egy spirál mentén meghatározott távolságra világos vagy sötét pöttyöket képez. Ezek a pöttyök a spirál mentén kiolvashatók, és egy összetettebb kódolási folyamat után bitsorozatként kiolvashatók. A ***LightScribe*** tulajdonképpen ugyan ezt műveli, de a lemez másik oldalán, ahol a világos és a sötét pöttyök sokkal eltérőbb színűek (arany és sötétbarna), a kiolvasást pedig az emberi szem végzi el.
A ***LightScribe*** lemez címke oldalán ugyanis a spirálon egymás követő pöttyök egy nagyon nagy felbontású monokróm (kétszínű) kép részét képezik. Mivel a felírt kétszínű pöttyök (pixelek) nagyon kicsik, a lemezen elkészülő képen szürke-átmenetek is szépen ábrázolhatók, mégpedig közel fotóminőségben (*1.* és *8. ábra*)!

Mi kell a LightScribe használatához?

Hardver szempontjából egy ***LightScribe*** képes író és legalább egy ***LightScribe CD*** vagy ***DVD*** lemez. Mindkettő „alkatrész” kapható nagyobb vagy felkészültebb számítástechnikai üzletben. Azt kell mondjam, hogy a igencsak mellé kell nyúlni ahhoz, hogy egy újonnan vásárolt ***DVD*** író ne legyen felkészítve ***LightScribe*** lemez írására. A lemezek többsége azonban még nem ***LightScribe***, így itt többet kell válogatnunk. Természetesen a ***LighScribe*** lemez (még) drágább, mint a megszokott lemezek!
A technológia megjelenésekor Linux alá egyáltalán nem volt olyan program, amely lemezcímkét tudott volna írni, és ez egészen 2006 közepéig így is maradt. 2006 őszére viszont megérkezett egy nagyon egyszerű ***LightScribe*** program, amelyet a ➲ http://www.lightscribe.com oldalról lehet letölteni. Ezen az oldalon három ***RPM*** csomagot találunk, amelyek közül kettő mindenképpen szükséges, mégpedig a ***lightscribe-1.4.124.1-linux-2.6-intel.rpm***, amely az alapvető hardverkezelésért felel; illetve a ***lightscribeApplications-1.4.119.1-linux-2.6-intel.rpm***, amely egy nagyon egyszerű címkéző programot rejt (***SimpleLabeler***). A harmadik csomag segítségével tudunk saját ***LightScribe*** programot írni (ehhez kell az ***SDK***).
A csomagokat feltelepítve a címkéző program ***még*** nem épül be az ablakkezelő programok menüjébe, így csak parancssorból tudjuk indítani, vagy készítünk rá egy alkalmazása mutató linket a munkaasztalra (vagy hozzáadjuk a menühöz). A program jelenleg a ***/opt/lightscribeApplications/Simple Labeler/*** könyvtárba települ, és ott a ***SimpleLabeler*** programot tudjuk elindítani.

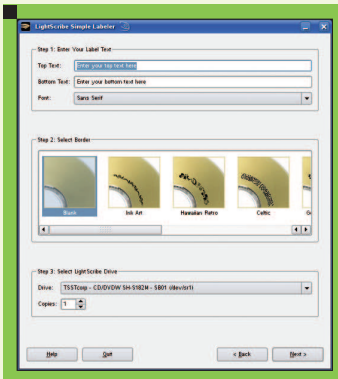
Egyszerű címkézés

A program egy bemutatkozó oldallal indul (*2. ábra*), és az immár klasszikus „varázsló” stílusban készíthetjük el a lemezeink borítóját.
A ***Next*** gomb megnyomásával tovább tudunk ugrani a program fő részéhez (*3. ábra*), amelyen kiválaszthatjuk a kívánt lemezmintázatot, és két szöveget, amelyet a program ráír a lemezre. Ennél többet sajnos nem tud a program, bár sok esetben is is bőven elég lehet. Az ablak alján kiválaszthatjuk, hogy melyik íróval és példányszámmal írjuk meg a lemezcímkét. Ha nincs ***LightScribe*** képes íróeszközünk, akkor ezen a ponton nem jelenik meg egy eszköz sem.



1. ábra Egy GIMP segítségével készített LightScribe DVD címketerv

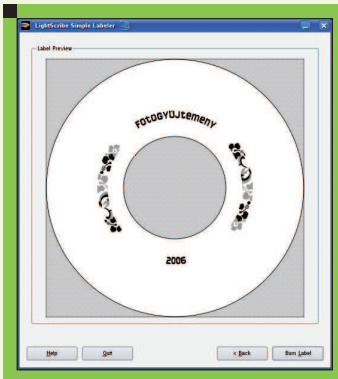
2. ábra Simple Labeler főoldal



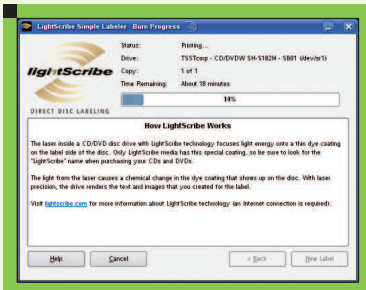
3. ábra Lemezmintázatok és feliratok

7. ábra Saját címke előnézete

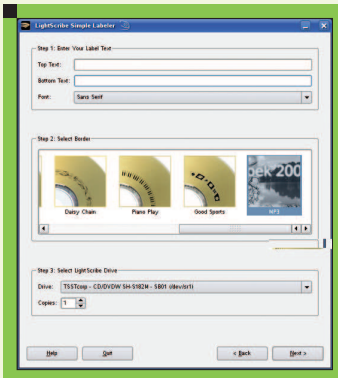
8. ábra Lemezcímke la GIMP



4. ábra Előnézet



5. ábra Lemezcímke írása



6. ábra Saját minta

Ha tovább lépünk, akkor a program készít egy előnézetet a lemezcímkéről (*4. ábra*), ha nem tetszik, vissza tudunk lépni az előző ablakba, és változtatásokat eszközölni.
Ezek után a program elkezdi felírni a címkét a lemez hátoldalára (*5. ábra*), azért lehetőleg pont fordítva tegyük be az íróba a ***LightScribe*** lemezt, mint azt íráskor megszoktuk. Írás közben a ***Simple Labeler*** különféle reklámokkal szórakoztat, s ideje van rá, ugyanis 20 percig is eltart a címke elkészítése.

Haladóknak

Kis utánajárással a ***Simple Labeler*** teljes körű címkéző programmá alakítható, ugyanis a ***content/images/borders/*** könyvtárban találhatjuk meg azokat a háttérképeket, amelyek a programban – mint választható minták – jelennek meg. Három könyvtár rejti ezeket a mintákat: ***fullsize***, ***metadata***, és ***thumbnails***; ezek rendre a teljes méretű képeket, a képek nevét, illetve az előnézetüket tartalmazzák.
Egy ***LightScribe*** háttérkép könnyedén elkészíthető ***GIMP*** segítségével, persze csak akkor, ha jártasak vagyunk a ***GIMP*** használatában. A képek mérete 2773x2773 pixel, és a formátumuk ***PNG***, használhatunk színeket is, de úgyis monokróm kép készül a lemez hátoldalára. Viszont érdemes a képet a ***GIMP*** segítségével szürkeárnyalatossá konvertálni, és megemelni a kontrasztot, különben nagyon halovány képet kaphatunk a lemez címkéjén.
Az előnézeti képek mérete 86x86 pixel, ezt könnyedén el tudjuk készíteni alapvető ***GIMP*** tudással, elég a kész címke egy részét kivágni és átméretezni.
A metaadatokat simán másolhatjuk, ezeknek a furcsasága, hogy ***UTF-16*** kódolással készültek, ***XML*** formátummal bírnak, egy ***XML***-szerkesztővel gond nélkül szerkeszthetők. Ha nincs ***XML***-szerkesztőnk, akkor át kell kódolnunk például ***UTF-8*** kódra és ezt már bármelyik konzolos editorral tudjuk szerkeszteni.
Célszerű egy új nevet (például ***md00009***) választani, és ezzel létrehozni a kettő ***PNG*** és a metaadat-hodozó ***XML*** állományt. A ***Simple Labeler*** program indulás után észre fogja venni az újabb mintát, és használni is tudjuk (*6. ábra*).
Ha írni akarunk a mintánkra, akkor megtehetjük a címkéző programmal is, de a ***GIMP*** sokkal tágabb lehetőségeket ad szövegek írására. Egyszerűen csak ki kell törölni a ***„Top text”*** és a ***„Bottom text”*** szöveget, a program ekkor nem rondítja össze a szépen koreografált címkénket.
Az előnézetben megtekinthetjük (*7. ábra*), hogy valóban szépen néz-e ki amit elterveztünk, a program ugyanis rárajzolja a lemez írási határait is.
A kész művünk (*8. ábra*) minden „igényt” kielégít, egy alkoholos filctoll biztos nem ilyen eredményt ad, bár a késztermék még messze van a szitázott lemezcímkétől, és sajnos csak monokróm lehet, de a gyártók már gőzerővel dolgoznak a színes lemezcímke technológiáján. Talán az újraírható ***LightScribe*** is meg fog érkezni egykor...

Auth Gábor
(auth.gabor@javaforum.hu)

KAPCSOLÓDÓ CÍMEK

LightScribe technológia –
➲ http://www.lightscribe.com

VNC – Tudom mit tettél tavaly az asztalodon

Egy egyszerű, de nagyszerű eszköz, mely segítségével távoli munkaasztalokon dolgozhatunk, mindezt bonyolult beállítások nélkül.

A *VNC (Virtual Network Computing)* egy rendkívül népszerű és elterjedt eszköz, melynek számos kereskedelmi és nyílt forrású változata érhető el az *Interneten*. Munkaasztal megosztó alkalmazás, mely egy szerverből (a távoli gépen fut, melynek a munkaasztalát megosztjuk) és kliens(ek)ből áll. Szinte minden mai operációs rendszeren megtalálható, sőt *Java* megvalósítása is van. *RFB (Remote FrameBuffer)* protokollt használja arra, hogy a megosztott asztalt billentyűzettel illetve egérrel irányítsuk. Ezt eredetileg az *Olivetti Research Ltd* és az *AT&T Labs* fejlesztette ki. Platformfüggetlen kapcsolatokat is létrehozhatunk vele, például linuxos *VNC* kliens képes akár *Windows* rendszeren futó szerverhez csatlakozni.

Használjunk VNC-t, de melyiket?

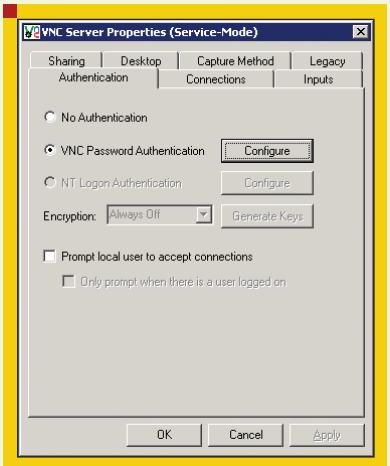
RealVNC, *TightVNC*, *UltraVNC*, *PalmVNC*, *KDE Desktop Sharing*, és így tovább, mind *VNC* változatok. Ezek közül a *RealVNC*-t, illetve a *KDE* alatti fejlesztéseket vizsgáljuk meg közelebbről. A *RealVNC* egy nagyon jól skálázható munkaasztal megosztó alkalmazás, a *KDE*-s *krfb* (*K Remote Frame Buffer*) illetve *krdc* (*K Remote Desktop Client*) pedig pont az ellentéte, egyszerű, könnyen kezelhető programok.

RealVNC (Free Edition)

Egyik legelterjedtebb változat, mely egyaránt megtalálható *Windows* és *Linux* alatt is. *Free, Personal és Enterprise edition* verziók érhetőek el belőle. Szabadon használható, és egyben a legkisebb tudású *Free Edition*-nal foglalkozunk, mely igazából csak a *VNC3* illetve *VNC4* protokollokat támogat semmi mást. A „nagy testvérek" már titkosított adatátvitelt, fájlmásolást, *Windows* asztal átméretezést, *Windows* és *NIS/NIS+* felhasználói azonosítást is támogat. Legfrissebb verzióját a ➲ *http://www.realvnc.com/download.html* oldalról tölthetjük le.

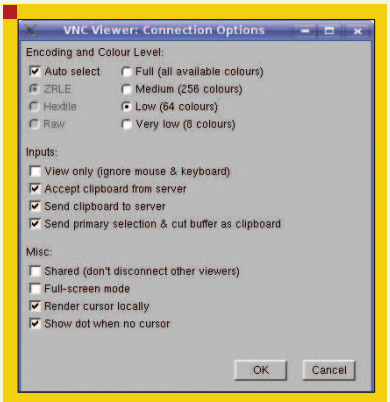
Windows alatt

A *Vncserver* alkalmazást általában *Windows* alá telepítjük (legtöbbször a *Windows* felhasználók szorulnak segítségre :), kezdjünk tehát ezzel. *Windows*-os változat roppant egyszerű, a szokásos „tovább" gombokra kattintás után a „beállítások" ablak jelenik meg.
A teljesség igénye nélkül nézzünk néhány fontosabb beállítást.
Az *„Authentication"* fülre kattintva néhány, a kapcsolat létesítését befolyásoló beállítási lehetőséget találunk. Először is azt dönthetjük el, hogy akarunk-e jelszavas azonosítást (*No Authentication/VNC Password Authentication*). Ha igen, akkor a *VNC Password Authentication* rádiódomb mellett elhelyezett *„Configure"* gomb segítségével állíthatjuk be a szupertitkos jelszavunkat. Ne felejtsük el, hogy bejelentkezéskor a jelszavunk mindenfajta titkosítás nélkül „utazik" kiszolgálónkhoz. Sajnos a *„NT logon Authentication"* (*Windows* bejelentkezés kezelés) és a *„Encryption"* (titkosított adatátvitel) lehetőségek a *VNC Free Edition* verzióban nem elérhetők. A *„Prompt local user to accept connections"* beállítás engedélyezésével, csak akkor tudunk csatlakozni a kiszemelt *VNC* kiszolgálóhoz, ha a gép előtt ülő felhasználó a felbukkanó dialógus ablak segítségével engedélyezi azt.



1. ábra RealVNC szerver beállítása

A *„Connections"* lapon a kliensek kapcsolódásait szabályozhatjuk. Megadhatjuk, hogy *VNC* szerverünk mely *TCP* kapun hallgatózzon (alapértelmezett az 5900), mennyi idő után szakítsa az inaktív kapcsolatot (alapértelmezett a 3600 másodperc), illetve az *„Access Control"* komponenesen elhelyezett listában adhatunk meg *IP* címeket, alhálózatokat, ahonnan a kapcsolatot kiszolgálónk elfogadja. A *„Only accept connections from the local machine"*-t engedélyezve, a távoli bejelentkezést tudjuk tiltani.
„Inputs" lapon érdekes és egyben hasznos beállításokat találunk. *„Accept pointer events from clients"*-t engedélyezve képesek vagyunk az egérmutatót mozgatni, klikkelni. *„Accept keyboard events from clients"* engedélyezése esetén aktív *VNC* kapcsolat alatt a billentyűzetünkkel tudunk adatot bevinni a *VNC* szervert futtató gépen. Előfordul, hogy a *VNC* kliens vágólap tartalmára van szükségünk.



2. ábra Linux alatt futó vncviewer opciói

Ekkor jól jöhet a *„Accept clipboard updates from cliens”* lehetőség engedélyezése, így a kliensen vágólapra másolt adatokat saját *VNC viewer*-t futtató gépünkön is elérhetővé válik. Természetesen a kliens vágólapjára is küldhetünk adatokat, ha a *„Send clipboard updates to clients”*-t engedélyezzük. Engedélyezett *„Allow input events to affect the screen-saver”* kapcsoló esetén képesek vagyunk az esetlegesen aktiválódott képernyő pihentetőt megszakítani, és munkánkat tovább folytatni (Mondanom sem kell, hogy milyen kellemes dolog, ha ez az opció nem engedélyezett, és a „tanult” felhasználónk gépén egy perc után már a buborékokkal teli akvárium élménye tárul elénk). Sok esetben előfordul hogy a brutális *„Disable local inputs while server in use”* lehetőséget is engedélyeznünk kell (a „magától” működő egér igen irritálja a „lúzert”). Ekkor a felhasználónk csak a *POWER/RESET* gombok felett uralkodik.

A *VNC* szerverhez egy időben akár több *VNC* kliens is képes csatlakozni. Mint neve is mutatja a *„sharing”* fülön a *VNC* kapcsolatok megosztásait befolyásolhatjuk („mindig/soha/kliens oldali beállítás” szerint fogad el osztott kapcsolatokat), illetve a *„Non-shared connections replace existing ones”* hatására, egy új kapcsolat megszakítja az épp folyamatban lévőt.

Alacsony sávszélességnél jól jöhet a *„desktop”* fül beállításai, ahol a háttérkép és háttérminták átvitelét, sőt a grafikus effektusok egy részét is letilthatjuk. Ugyanitt állíthatjuk be, hogy mi történjen az utolsó kliens lekapcsolódása esetén (*Do nothing*, ne történjen semmi / *Lock workstation*, fiók zárolása / *Logoff user*, aktív felhasználó kijelentkeztetése). *„Capture method”* fülön a megjelenített kép átvitelének módját befolyásolhatjuk. *„Poll for changes to the desktop”* kiválasztása esetén csökken a *VNC* szervert futtató gép terhelése, viszont nő a válaszidő az egérrel illetve billentyűzettel kiváltott eseményekre. *„Use VNC hooks to track changes”* a klasszikus *VNC* képfrissítési eljárás előnye, hogy gyorsabb, szebb képfrissítést eredményez, bár bizonyos esetekben „elveszett” (nem frissült) területek is megjelennek a *VNC* kliensünk ablakában. A *„Poll console windows for updates”* engedélyezése nélkül az esetlegesen futtatott *„cmd”* parancs ablaka rendkívül lassan frissül, ezért érdemes ezt a beállítást engedélyezni. *„Capture alpha-blended windows”* opcióval két képmegjelenítési eljárás közt választhatunk. Ha nem engedélyezett, akkor gyorsabb kép-megjelenítést kapunk, viszont néhány grafikus effektus (például buborékok) nem jelennek meg. Ellenkező esetben minden grafikus effektus átvitelre kerül a klienshez, de ez a rendszer terhelését természetesen növeli, sőt a helyi egérmutató sok esetben vibrál a képernyőn.

Nagyjából ennyi a *Windows* alatt futó *RealVNC* szerver beállításairól. *Viewer* (kliens) részében nagyon hasonló beállításokat találunk, használata szinte gyerekjáték, ezért ezt most nem részletezem.

Linux alatt

A ➲ *http://www.realvnc.com/products/free/4.1/download.html* címről a lefordított *Linux* binárist illetve a forráskódot is letölthetjük. Maradjunk az előre fordított bináris csomagnál. Kicsomagolás után a `./vncinstall` szkripttel tudjuk helyére pakoltatni a fájlokat. A parancs paraméterként várja a bináris fájlok helyének elérési útját (`/usr/local/bin` javasolt), a *man* oldalak telepítési útvonalát (`/usr/local/man`), illetve a *VNC* modul leendő elérési útját (`/usr/local/lib`). Ha minden jól ment, akkor már használható is a frissen telepített *VNC* csomagunk.

Nézzünk meg néhány *VNC* programocskát és azok fontosabb beállításait.

A `vncviewer` egy egyszerű és nagyon gyors kliens program, paraméterként a *VNC* szervert kiszolgáló gép nevét is a port számot adhatjuk meg (például `localhost:5902`). Aki idegenkedik a parancssori kapcsolóktól, az indítsa paraméter nélkül, mivel szinte minden beállítása elérhető grafikusan is.

Szinte minden beállítás megegyezik a *Windowsos* verziónál tárgyaltakkal. A képátvitel kódolása mellett kiválaszthatjuk a megjelenítendő színmélységet. Említést érdemel a *„View only”* opció, mely hatására csak betekintést kapunk a távoli asztalra, egér és a billentyűzet eseményeit nem küldi el kliensünk a kiszolgálónak (így majdnem észrevétlenek lehetünk). A *„Render cursor locally”* engedélyezésével kliens gépünk fogja az egérmutatót megjeleníteni, ami akkor lehet hasznos, ha alacsony sávszélesség mellett érjük el kiszolgálónkat, így az egérmutató nem fog „szaggatni”.

Az `Xvnc` tulajdonképpen az *X VNC* szerver program. Ez indít el egy *VNC* szervert és egy új *X* szervert virtuális *framebuffer*-ként. Normális esetben a lentebb bemutatott `vncserver` szkript indítja el, megfelelő paramétereket átadva, de természetesen önállóan is futtatható. Parancssori kapcsolói közül érdekes lehet a `-localhost` kapcsoló, melynek hatására csak a gazdagépről fogad el kapcsolatokat. Gyakorlati haszna leginkább az *SSH* port átirányítással kombinálva van, hiszen a *VNC* felhasználó-azonosító mechanizmusa nem igazán mondható biztonságosnak, mivel minden egyes *TCP* csomag titkosítás nélkül kerül továbbításra. A `-QueryConnect` kapcsolóval indított *VNC* kiszolgáló új kapcsolat létrehozása előtt a gép előtt ülő felhasználó jóváhagyását kéri.

Az `Xvnc` program képes egy *mini-http* kiszolgálót is futtatni, a `-httpd <elérési út>` helyen található *Java viewer*-t futtatva. Így akár weben keresztül, egyszerű böngésző segítségével is elérhetjük távoli asztalunkat. A `-httpPort <portszám>` segítségével állíthatjuk be, hogy a mini web kiszolgálónk melyik *TCP* kapun hallgatózzon kapcsolat után. A `-inetd` kapcsolóval kiszolgálónk tudomásul veszi, hogy igény esetén az

1. táblázat *A vncserver néhány hasznos kapcsolója*

`-name <asztalnév>`	A kapcsolóval a kliensnél megjelenő asztal nevét állíthatjuk be.
`-geometry <szélesség x magasság>`	A megjelenítendő asztal méretét állíthatjuk be (alapértelmezett méret a 1024x768).
`-depth <színmélység>`	Megjelenítés színmélységet adhatjuk meg 8,15,16,24 értékek közül (alapértelmezett a 16 bpp).
`-kill <:képernyő>`	Az előzőleg indított VNC képernyő futását megszakítja (`vncserver -kill localhost:1`).
`vncpasswd`	Segítségével beállíthatjuk illetve menet közben cserélhetjük a VNC kiszolgáló eléréséhez szükséges jelszavat. A `vncszerver` szkript első indításkor, futtatja a `vncpasswd`-t. Jelszavunknak minimum hat karakter hosszúnak kell lennie, mely kódoltan a saját mappánk `/.vnc/passwd` fáljába íródik.
`vncconfig`	E program segítségével lehet a futó `Xvnc` beállításait módosítani. A `vncconfig` program *VNC4* előtti verzióknál nem használható.

inetd/xinetd fogja elindítani. *Xinetd* esetén a következővel kell bővíteni a `xinetd.conf` állományt:

```
service vnc01
{
  protocol     = tcp
  socket_type = stream
  user         = programmer
  wait         = no
  server       = /usr/bin/Xvnc
  server_args = -inetd
  ➥-geometry 1024x768 -depth
  ➥16 -rfbauth
/home/programmer/.vnc/passwd
}
```

Utána adjuk ki a

```
# echo "vnc01 50001/tcp" >>
➥/etc/services
```

parancsot, ettől az *xinetd* tudni fogja, hogy a *vnc01* szerviz az 50001-es *TCP* kaput jelenti. Az *xinetd* újraindítása után már kapcsolódhatunk is az új *VNC* asztalunkhoz, mely 1024x768 méretű, 16 bites színmélységgel. Az `-rfbauth` kapcsolóval a felhasználó jelszavát tartalmazó fájlt adjuk meg. A `vncserver` egy *Perl* szkript, mely elindítja a *VNC* szerver szolgáltatást, illetve létrehozza a szükséges konfigurációs fájlokat. Paraméterként megadhatjuk, hogy hányadik *X* képernyőként induljon el (például `vncserver :13`). Ha nem adunk meg semmit, akkor az első szabad *X* képernyőt veszi alapértelmezettnek (Ha fut *X* kiszolgáló gépünkön, ami általában a :0 képernyő, akkor a *VNC* automatikusan a :1 képernyőn indul el). Néhány hasznos kapcsolóját az *1. Táblázatban* foglaltuk össze.

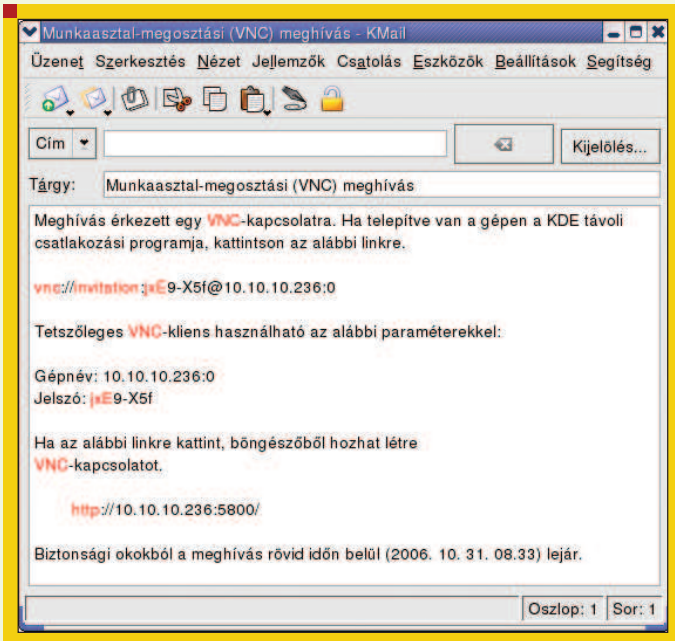
Nem csak ennyi

A *Windowsos* illetve *Linuxos RealVNC* verziók is ettől több és összetettebb opciókkal rendelkeznek, melyeknek ismertetésével talán még egy könyvet is meg lehetne tölteni. Aki többre kíváncsi, annak ajánlom a *VNC* eléggé részletes súgóoldalait. Ettől függetlenül elmondható, hogy a *VNC* programok döntő többsége, könnyen és egyszerűen használható, minimális ismeretével már sikeresen oszthatunk meg munka asztalokat.

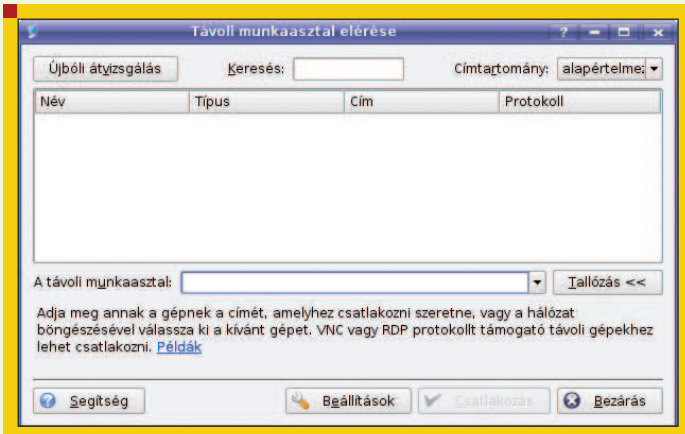
KDE alatt

Krfb, krdc a legtöbb disztribúció *KDE* telepítése tartalmazza, általában a *kdenetwork* csomag részei. Használatuk nagyon egyszerű, segítségükkel, néhány „klikkeléssel” képesek vagyunk kedvenc asztalunkat megosztani, illetve távoliakat elérni. Egyik program sem rendelkezik hasznos parancssori kapcsolókkal, mindent grafikusan tudunk beállítani.

3. ábra A krfb egyszerű felülete



4. ábra Krfb meghívó levélként, már csak a címzettet kell megadnunk



5. ábra A krdc letisztult felülete

A *krfb* indításakor lehetőségünk van meghívók kezelésére. „Egyéni meghívó készítése” menüpontot általában a segítségre szoruló felhasználók használhatják, akik probléma esetén a távoli rendszergazda segítségét kérik. Megjelenik egy ablakban gépük *IP* címe, egy, a program által generált jelszó, és a lejárási idő. Ameddig érvényes a meghívó (lejárási idő), addig lehet kapcsolódni a megosztásunkhoz. Megosztás kész, most már rajtunk múlik, hogy a távoli felhasználónak hogyan juttatjuk el ezeket az adatokat. *„Meghívó küldése emailen”* menüpont nagyon hasonló az előzőekhez, csak itt a *krfb* elindítja a rendszer alapértelmezett levelező kliensét, és egy új levelet hoz létre, mely tartalmazza a jelszavat, *IP* címet, lejárási időt. *„Meghívók kezelése”* menüpont kiválasztásakor a már létrehozott meghívóink listáját láthatjuk, melyet bővíthetünk újakkal, de törölhetünk is belőle. A *krfb* főablakán találunk még egy *„Beállítás”* gombot, melyre kattintva egy *KDE* beállítómodul indul el, mellyel minimálisan befolyásolhatjuk kiszolgálónk működését. A *„Meghívás nélküli kapcsolatok engedélyezése”* beállítást engedélyezve a felhasználó meghívása nélkül is tudunk kapcsolódni a *krfb-t* futtató géphez, csak a jelszavat kell ismernünk. „A szolgáltatás bejelentése hálózaton” opciónak meghívás nélküli kapcsolatoknál van értelme, mikor a kiszolgáló üzenetet küld a hálózaton lévő gépeknek a szolgáltatás létezéséről. Beállíthatjuk, hogy a felhasználó jóváhagyása nélkül is átvehessük a teljes kontrollt az asztal fölött.

„Munkafolyamat fülön” a háttérkép átvitelét lehet letiltani (alacsony sávszélességnél hasznos), míg a hálózat fülön a *TCP* kapu automatikus kiválasztását tudjuk befolyásolni.

A *krdc* nagyon egyszerű, felhasználóbarát *VNC/rdesktop* kliens. „A távoli munkaasztal” mezőbe írhatjuk be a cél *URL*-jét. Egyszerű *VNC* kapcsolat esetén elég a *„hostnév:port”* *URL*-t megadni. Terminál szerver elérése esetén a hosztnév elé egy *rdp:/* tagot kell csak szúrnunk (*rdp:/rdpszerver*). A *„Beállítások”* gombra kattintva jelenik meg a *krdc* beállító panelja. *„Profilok”* fül tulajdonképpen egy *„history”* azokról a kapcsolatokról, melyeket már használtunk. A *„VNC alapértelmezései”* illetve „Az *RDP* alapértelmezései” füleken a két protokollra vonatkozó alapbeállításokat találjuk, mint a *VNC* csatlakozás típusa átviteli sebesség szerint, digitális notesz (*kwallet*) jelszókezelő használatának engedélyezése, míg az *RDP* fülön a munkaasztal méretét, színmélységét, illetve a kívánt billentyűkiosztást tehetjük alapértelmezetté.

KDE „feeling”

A *krfb* és a *krdc* is olyan, mint maga a *KDE*. Nagyon szép, nagyon könnyen kezelhető, minimális beállítási lehetőséget kell átnéznünk, viszont cserébe egy lassú elindulási és működési sebességet kapunk (egy *RealVNC* klienssel összehasonlítva).

Gráma Tibor
(tibor.grama@hoya.lmh.hu)

1997 óta „Linuxozik”, UHU hívő. Szabadidejében gyermekeivel és vizsla kutyáival játszik, ha éppen nem kertészkedik vagy horgászik.

KAPCSOLÓDÓ CÍMEK

- http://www.realvnc.com
- http://www.kde.org

Távoli hőmérséklet-figyelés a Linuxszal

Készítsünk távoli hőmérséklet-figyelő berendezést kis igényű Linux és olcsó vas felhasználásával.

Egy hűtés, fűtés és légkezelés területén tevékenykedő barátommal folytatott beszélgetésem után vágtam bele ebbe a projektbe a hőmérsékleti adatok távolról történő eléréséhez és rögzítéséhez. Az ő feladata biztosítani, hogy a kinti időjárástól függően kellemes legyen a benti légkör – nem túl meleg, de nem is túl hideg. Tapasztalata szerint sok új berendezésben adódnak kezdeti hibák, amelyeket meg kell oldani, hiszen nincs két pontosan egyforma összeállítás. A munka utolsó szakasza a legnyomasztóbb. A befejezés után a vásárló felhívja, és közli, hogy valami nincs rendben. A megrendelő elégedetlen, de a barátom nem tudja, hol kezdje keresni a problémát, mert nincs semmi használható objektív információ a berendezés hibás működésének mibenlétéről.

Egyetértettünk abban, hogy hasznos lenne rögzíteni a külső hőmérsékletet, és elektronikusan naplózni a méréseket. Ez lenne a hibakeresési folyamat javításának egyik módja. Ezután elkezdtem olcsó, könnyen telepíthető és egyszerűen használható készre gyártott rögzítőeszközt keresni. A keresgélés során rengeteg kereskedelmi terméket és készletet találtam. Voltak önállóak, és voltak olyanok, amelyek számítógép segítségével jelenítették meg és rögzítették a hőmérsékleti adatokat. A három követelmény egyformán fontos volt, és a termékek többségét túl drágának találtam a költségvetéshez képest. A telepítés egyszerűsége volt a másik jellemző probléma. Bizonyos eszközök huzalozása bonyolult volt, vagy követelmény volt, hogy a hőmérséklet mérésének helyén kell elhelyezni. Sokan úgy gondolják, hogy a hőfokszabályzó csak elcsúfítja a falat, ezért nem lehetett volna könnyen meggyőzni őket, hogy még egy dobozt szereljenek fel a hőmérséklet rögzítésére.

A linuxos megoldás

A költségek és a telepítés problémáinak orvoslására végül megvizsgáltam a rendszer alkotóelemenként történő felépítésének lehetőségét. A ***Linux*** elvét követve, amely az eszközök összegyűjtésével és egyesítésével oldja meg a feladatokat, arra gondoltam, hogy egy digitális multimétert és egy számítógépet használok, valamint egy programot, amely segítségével együtt tudnak működni. A terv szerint az ***RS-232*** soros kapuval ellátott digitális multiméter egy érzékelővel méri a hőmérsékletet. A számítógép összegyűjti a multiméter adatait, és feldolgozza azokat a megjelenítéshez.

Ismertem ***Linuxot*** használó nyílt forráskódú alkalmazásokat a multiméterhez, és korábban vásároltam egy multimétert általános hibakereséshez. Rendelkezésre állt egy leselejtezett ***PC***, így minden szükséges összetevő a kezemben volt a hőmérséklet-rögzítő rendszer mintapéldányának megépítéséhez.

A digitális multiméterhez tartozott egy soros kapukábel, valamint egy ***DOS*** program. A mellékelt ***DOS*** programot nem használtam, mert nem lehetett úgy módosítani, hogy lehetővé váljon az érzékelővel történő hőmérséklet-mérés. Ehelyett teljesen új programot írtam a ***QuickBasic*** segítségével. Rendelkezésre álltak a multiméter soros kapujára vonatkozó szükséges információk, a ***QuickBasicben*** pedig minden olyan szolgáltatás adott volt, amelyekre szükségem volt.

Sikerült működésbe hoznom egy kezdeti kommunikációs programot, de a program méretének növekedésével a ***DOS*** és a ***QuickBasic*** memóriakezelési problémáiba ütköztem, különösen amikor hozzáláttam megoldani az adatok megjelenítésének és rögzítésének igényét.

Ezen a ponton nagy visszalépésnek tűnt a memóriakezeléssel küzdeni. Tudtam, hogy a ***Linux*** olyan környezetet biztosítana, ahol nem kell a memóriakezeléssel foglalkoznom, ezért egy olyan terjesztés után néztem, amellyel helyettesíthetem a ***DOS***-t. Kiderült, hogy a ***Linux*** legnépszerűbb

terjesztései nem megfelelőek ehhez az alkalmazáshoz. Még a terjesztések minimális telepítése is meghaladná a leselejtezett gép teljesítményét. Találtam egy terjesztést, amely felülemelkedett a korlátokon: a ***University Linuxot***, ***Paul Mullertől***, amely kis memória- és lemezigénnyel rendelkezik. Kevesebb, mint ***20 MB DOS*** által formatált lemezterület és ***24 MB RAM*** felhasználásával tudtam futtatni.
A legnagyobb előny az, hogy ez a terjesztés jól tűri az áramszüneteket. Ha kimarad az energiaellátás, a számítógép anélkül indul újra, hogy kézi beavatkozást igénylő fájlkárosodást okozna. Ez pénzt takarít meg, és csökkenti a rendszer összetettségét, mert nem kell szünetmentes áramforrást (***UPS***) alkalmazni ahhoz, hogy a rendszer az áramkimaradások idején tovább működjön.
Miután mindent beállítottam a gépen, mér nem volt szükség billentyűzetre vagy monitorra. Egy ***Windows PC*** és ***Telnet***, valamint ***Ethernet*** kapcsolat segítségével tudtam kommunikálni a rendszerszámítógéppel a fejlesztéshez és teszteléshez. A folyamatos írás és tesztelés híve vagyok, ezért a ***Perl*** nyelvet választottam a projekthez.
A ***University Linuxban*** a ***Perl 5.003*** változata található meg. Nem tudtam ***Perl*** modulokat használni, mert az alkalmazás mérete túl kicsi volt, és ez kisebb kényelmetlenséget jelentett.
A ***University Linux*** az ***Acme Labs*** ***thttpd*** kiszolgálóját is tartalmazza. Ennek segítségével úgy lehet beállítani a rendszert, hogy egy webböngésző segítségével jelenítse meg a hőmérséklet-méréseket.

A hardver tesztelése

Tandy 22-805 típusszámú digitális multimétert használtam, amelyhez egy használati utasítás, egy ***DOS*** program, mérővezetékek és egy 9 tűcsatlakozóval ellátott kábel tartozik. A kézikönyv szerint az adatátviteli beállítások 600 baud, 7 adatbit, 2 stop bit és nincs paritás. Egy fontos adatot kihagytak a leírásból, de a szükséges információt megtaláltam az interneten.
A ***DTR*** és az ***RTS*** vezetékre külön figyelmet kell fordítani. A ***DTR*** vezetéket alacsonyra, az ***RTS*** vezetéket pedig magasra kell állítani ahhoz, hogy a multiméter adatátvitelt tudjon folytatni a soros kapun keresztül. Ha a két vezeték ettől eltérően van beállítva, lehetetlen adatot kapni a multimétertől.
Ennél a terjesztésnél csak az ***stty***-t tudtam soros kommunikációra használni, és a parancsfájlban nem lehetett közvetlenül vezérelni a ***DTR*** és az ***RTS*** vezetéket.
Azt tapasztaltam, hogy amikor a parancsfájlban meghívom a ***stty***-t, a ***DTR*** és az ***RTS*** alacsonyból magasba vált. Ez a ***DTR*** esetében jó, de az ***RTS***-nek alacsonyan kell maradnia. Észrevettem, hogy a számítógép második soros kapuján alacsony az ***RTS***, és nincs használatban. Ha a multiméter ***RTS*** soros felület kábelét a második soros kapu ***RTS*** tűjéhez csatlakoztatnám, elhitetném a multiméterrel, hogy a helyes vezetékbeállításokat látja. Egyszerűen eltávolítottam az ***RTS*** vezetéket a multiméterből, és a második soros portra kötöttem.
Miután megoldottam ezt a problémát, bekapcsoltam a multimétert és az alábbi rövid próba-parancsfájlt (***serialtest.pl***) hoztam össze:

```
# !/usr/bin/perl
#
# serialtest.pl
#
# Parancsfájl 22-805 típusú
# Tandy multiméter
# soros kapun keresztül történő
# olvasásához
$port = "/dev/ttyS1"; # set to
➥COM1
system ("stty 600 cs7 cstopb
➥clocal -ixon -echo < $port");
open (SERIALPORT, "+>$port") or
➥die "can't open $port. ";
print SERIALPORT ("\n"); # take
# a reading
$R = <SERIALPORT>; # read
# returned string
print "$R" ;
close (SERIALPORT); # close
# port
exit 0;
```

A parancsfájl sikeres futása esetén olyan karakterláncot kell kapnom, amely azonos a multiméter kijelzőjén megjelenő értékkel. Az eredmény:

```
OH    O.L MOhm
```

Kezdetnek jó! Ezután a multiméterrel le kellett olvasnom a hőmérséklet érzékelőt.
Egy ***NTC (negative temperature coefficient – negatív hőmérsékleti együtthatójú)*** termisztort választottam a hőmérséklet-érzékelőbe. Az ijesztően hangzó név ellenére ez csupán egy kétvezetékes elektronikus alkatrész, amely a hőmérséklettől függően változtatja az elektromos ellenállását.
A multiméter esetében az ellenállás érték szolgáltatja a hőmérsékletet meghatározó információt. A termisztort nem lehet fordítva bekötni, mert nem érzékeny a feszültségpolaritásra (+ vagy -). Ez azt jelenti, hogy a helyszínen a telepítést végző szakembernek eggyel kevesebb dolog miatt kell aggódnia.
A termisztor nem törékeny, de a testéhez csatlakozó vezetékek erős rángatás vagy hajlítás hatására megszakadhatnak. Egy kétállású elosztóblokkot használtam a probléma megoldására és a vezetékekhez történő csatlakozás egyszerűsítésére. Az egyik termisztor kábelt és egy vezetéket egy csavaros lezáró alá helyeztem, majd meghúztam a csavart, hogy biztos mechanikus és elektronikus érintkezést kapjak.
A termisztort a mérővezetékek végeihez csatlakoztatva, a mérővezetékeket pedig a multiméterbe dugva ismét bekapcsoltam a készüléket, majd elindítottam a próba-parancsfájlt. A következő ellenállás értéket kaptam:

```
OH   34.23kOhm
```

Az állás számérték része `34,23`, utána pedig egy k betű áll. A `k` a kilo, azaz 1000 rövidítése. Mivel a multiméter kijelzőjén nincs elég hely a nagy számok megjelenítéséhez, szorzót alkalmaz. Ebben az esetben a `34.23k` 34 230 Ohm-ot jelent.
A gyártó által mellékelt ellenállás-hőmérséklet táblázatból kiderítettem, hogy ez az érték nagyon közel van a 0 °C-hoz. Ez megegyezik egy másik hőmérő mérésével, amelynek az érzékelője a közelben volt elhelyezve, ezért biztos voltam abban, hogy az összeállított rendszer működik majd, és pontos értéket szolgáltat.
Most létre kellett hoznom egy parancsfájlt, amely az adatokat felhasználva megjeleníti a hőmérséklet-értéket.

Szoftverválasztás

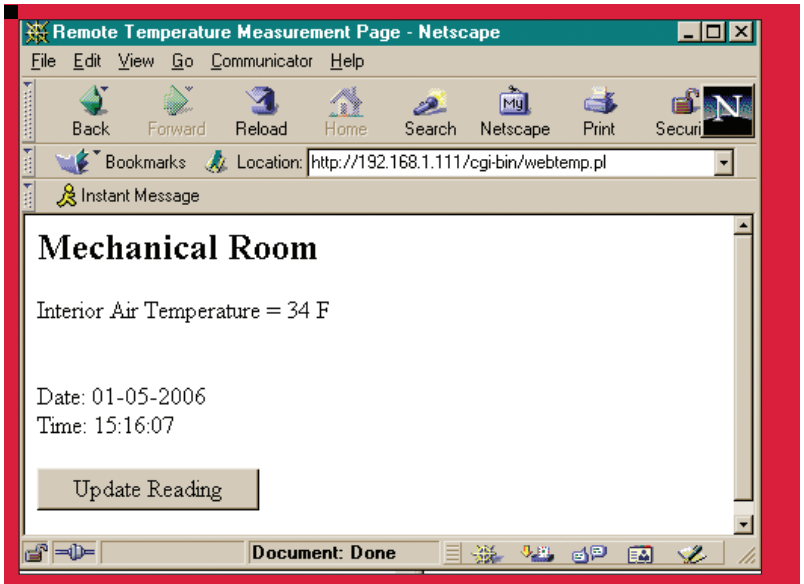
Kétféleképpen lehetett megoldani a ellenállás-hőmérséklet átváltást a parancsfájlban. Használhattam volna egy adattáblázatot, tömbben tárolva az ellenállás-hőmérséklet értékpárokat. Ennél a megközelítésnél a tömbben található elemek teljes száma jelent hátrányt. Egy -40°C-tól +40°C-ig terjedő tartományban 81 (ne feledjük a 0°C-ot) értékpárra van szükség. Nem létezik olyan egyszerű módszer, amellyel módosítani lehetne termisztor gyártója által rendelkezésre bocsátott szövegfájlt, ha pedig magunk gépeljük be az értékeket, az sok időbe telik, és hibákat eredményezhet.
Ehelyett az úgynevezett ***Steinhart-Hart*** egyenletet alkalmaztam (lásd a keretes írást). Ezt az egyenletet arra fejlesztették ki a '60-as évek végén, hogy segítse a termisztorokkal gyűjtött óceán-hőmérséklet adatok feldolgozását, és az alkalmazásával közvetlenül hőmérsékletre lehet váltani az ellenállást. Egy interneten talált táblázatkezelő segítségemre volt az egyes termisztorcsaládokra jellemző együtthatók kiszámításában, és ezt használtam az egyenletben.

Adatok megjelenítése

Miután a parancsfájl a multiméter méréséből kiszámítja a hőmérsékletet, meg kell jeleníteni, valamint tárolni kell azt. Ezt szem előtt tartva bővítettem a parancsfájlt, hogy átváltsa és megjelenítse a hőmérsékletet, továbbá kiírja az idő és az ellenállás értéket.
A ***University Linux*** 2.0 rendszermagot használ, és engedélyezi a rendszergazda felhasználó bejelentkezését a *Telneten* keresztül. Amikor hagyományos felhasználók próbálják futtatni a ***grabtemp.pl*** parancsfájlt, hibajelzés jelenik meg a ***/dev/ttyS1*** soros kapunál használt jogosultságok miatt. Ezt a jogosultságok megváltoztatásával küszöböltem ki:

```
chmod a+x /dev/ttyS1
```

Ezután a hagyományos felhasználók is be tudtak jelentkezni, és a parancsfájlt futtatva ellenőrizni tudták a hőmérsékletet. Nem volt szükség rendszergazda hozzáférésre.
Íme az eredményül kapott ***showtemp.pl*** parancsfájl kimenete:



1. ábra A hőmérséklet-figyelő már webes felülettel rendelkezik

```
/perlserial: perl -w
➥showtemp.pl
01-05-2006 14:43 34 F 1.3 C
➥30.52 k Ohms
```

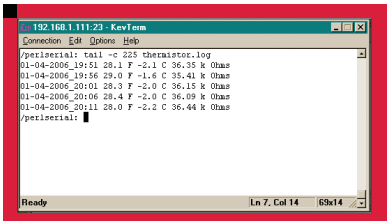
Itt láthatjuk a dátumot, a °F-ben és °C-ban megadott hőmérsékleteket, valamint a pillanatnyi ellenállásértéket. Ellenőriztem a hőmérsékletet ott, ahol az érzékelőt elhelyeztük, és azt láttam, hogy az érték pontos, tehát a parancsfájl átváltási képlete működött.
Nem sok számítógép-felhasználó szereti a parancssoros programfelületeket. Az egérrel irányított kezelőfelület sokkal barátságosabb. Tehát ismét kibővítettem a parancsfájlt, hogy lehetővé tegyem a felhasználók számára rendszer webböngészőn keresztül történő működtetését.
Az üzembe állított és működő ***thttpd*** kiszolgáló mellett nem volt más dolgom, mint átirányítani a parancsfájl kimenetét úgy, hogy egy weboldalt hozzon létre a megjelenítéshez.
Ahogy az alábbi kód mutatja, ez meglehetősen egyszerű:

```
print "content-type: text/html
➥\n\n";
print "<HTML><BODY><P>";
print "<HEAD><title>Remote
➥Temperature Measurement
Page</title></HEAD>";
print "<H2>Mechanical
➥Room</H2> ";
print '<form action=
➥"webtemp.pl" method=post>
➥<P> <P>';
print "Interior Air
➥Temperature = $out_tempF<BR>";
print "<BR>";
print "<BR>";
print "Date: $out_date  <BR>";
print "Time: $out_time  <BR>";
print "<BR>";
print '<input type=submit
➥value="Update Reading">';
print "</form>";
print "</BODY></HTML>";
```

A ***webtemp.pl /cgi-bin***-ből történő futtatásakor az *1. ábra* példaképernyőjéhez hasonló jelenik meg a felhasználók számára.
Ez a példa a szoba hőmérsékletét, illetve a mérés időpontját és dátumát is megjeleníti. Az ***Update Reading*** (Leolvasás frissítése) gombra kattintva ismét futtathatjuk a parancsfájlt, és újabb hőmérséklet értéket jeleníthetünk meg.
Nem nehéz olyan kiterjesztést írni a parancsfájlhoz, amely folyamatosan naplózza a hőmérsékletet. Beillesztettem egy sort az *rc* (indító) parancsfájlba, amely elindít egy adatrögzítő parancsfájlt, amely ezután folyamatosan fut a háttérben.
Úgy ítéltem meg, hogy 5-10 perces időközönkénti méréseket érdemes használni, mivel zárt, légkondicionált térben lassan változik a levegő hőmérséklete.

A termisztorok és a Steinhart-Hart egyenlet

Egy termisztor ellenállás-hőmérséklet függvénye grafikonon ábrázolva nagyon hasonlít egy sí rámpa ívéhez, és minden termisztorcsalád saját, egyedi görbével rendelkezik. Egy egyszerű $y = mx + b$ képlet tehát nem segít az ellenállás hőmérsékletre váltásában. A görbék egyenlete egy többtagú kifejezéssel írható le. A ***Steinhart-Hart*** egyenlet egy trinomiális, vagyis egy három kifejezésből álló egyenlet. Ahhoz, hogy az egyes ellenállás-méréseknél megoldjuk az egyenletet, három együtthatóra van szükség: az a-ra, a b-re és a c-re. Vannak gyártók, amelyek megadják a termisztorok együtthatóit. Mások csak az ellenállás-hőmérséklet átváltási táblázatot mellékelik.
Ha az együtthatók nem állnak rendelkezésre, egy létező táblázatkezelő alkalmazás segíthet kiszámítani azokat a gyártók táblázatai alapján. Nincs szükség számolásra, csak egyszerűen begépeljük a három hőmérséklet és ellenállás értéket a táblázatba, ami automatikusan kiszámítja az együtthatókat. Ezek általában nagyon kis számok, amelyeket tudományos jelölésrendszerrel ábrázolnak. A kiszámított együtthatókat a táblázatból a *Perl* parancsfájlba pásztázva azonban csökkenthetjük a gépelési hibák esélyét.



2. ábra A hőmérséklet-figyelő naplóban tárolt példaértékek így mutatnak a Telneten keresztül

A ***Telnet*** segítségével a parancssoron keresztül elérhetjük a hőmérséklet naplót. Mivel a formátum helykorlátos, a ***Microsoft Excel*** segítségével grafikonokat és megjelenítési irányvonalakat készítettem a dátumfájlból. A 2. *ábrán* egy lehetséges kimenetet láthatunk.

Biztonsági kérdések

A végső cél a hőmérséklet-adatok megjelenítésére és rögzítésére szolgáló, megbízható, könnyen alkalmazható elektronikus eszköz létrehozása volt. A rendszer tényleges kialakításakor az elhelyezkedés és a hálózati kapcsolat meglehetősen változó lehet. Minden összeállításnál a körülményeket figyelembe véve kell mérlegelnünk ma biztonsági kérdéseket. Előfordulhat, hogy kerülőutakat kell alkalmaznunk a biztonsági problémák kezelésére. Megtehetjük például, hogy nem a ***cgi*** könyvtárban található parancsfájl, hanem egy háttérben futó parancsfájl alkalmazásával rögzítjük a hőmérséklet-értékeket szöveg vagy *HTML* formátumban, ezáltal elkülönítjük a naplózás folyamatát a webhozzáféréstől. Másik lehetőségként az egyik kiszolgálóról *FTP*-n vagy *HTTP*-n keresztül gyűjthetjük az adatokat, egy másik biztonságos kiszolgáló segítségével. Ez újabb védőréteget alkot, amely megakadályozza a közvetlen hozzáférést a külvilág felől, mégis elérhetővé teszi az adatokat.

További bővítések

A digitális multiméterek általános célú mérőeszközök. Ugyan ennél az összeállításnál termisztort használtam a hőmérséklet mérésre, de bármilyen érzékelőt alkalmazhatunk, amelynek kimenete ellenállás, feszültség vagy elektromos áram. Áramlást, nyomást, súlyt, fényerőt és páratartalmat is mérhetünk. Nincs szükség több multiméterre, ha több hőmérsékletet akarunk mérni. Megtehetjük, hogy egyetlen multimétert kapcsolóberendezésre kötünk. Ezután egy parancsfájlt készítünk a kapcsolóberendezés működtetéséhez, ami lehetővé teszi, hogy egyenként válasszuk ki a hőmérséklet-érzékelőket.

Összegzés

Ez a példa szemlélteti, hogy a ***Linux*** eszközelvűsége hogyan működik az olyan alkalmazások megoldásánál, amelyeknél fontos a költség és a rugalmasság. A rendelkezésre álló terjesztések más operációs rendszerekhez képest bő választéka célszerűvé tette a szükséges szolgáltatásokat megvalósító rendszer fejlesztését. Ezen felül a ***Perl*** és a ***University Linux*** terjesztés által biztosított fejlesztőfelület segítségével további szolgáltatásokat adhatunk hozzá.
A rendszer kevesebb, mint 100 dollárból elkészíthető. A multiméter, a termisztor és a huzalozás tartozékai számos elektronikai cikkeket forgalmazó kereskedésben elérhetők. Sok forgalmazó rendelkezik webhellyel, így megrendelés előtt könnyű összehasonlítani a szolgáltatásokat, a tulajdonságokat és az árakat. Használt multiméter vásárlásánál körültekintőnek kell lenni, mert nehéz megállapítani, hogy a korábbi használat befolyásolta-e a műszer pontosságát.

Linux Journal 2006., 144. szám

Steven M. Lapinskas a szoftverminőség-biztosítás, a mechanikus tervezés és a projektvezetés terén rendelkezik szakmai háttérrel. Szabadideje egy részében a Linux és a számítógépen kívüli való világ kapcsolatát kutatja.

KAPCSOLÓDÓ CÍMEK

➲ www.linuxjournal.com/article/8833

D-Link®
Building Networks for People

Hálózati eszközök (1. rész)
A vezeték nélküli hálózatok rejtelmei

Ugye milyen kényelmes dolog számítógéppel az ölünkben üldögélve bámulni a tévét, úgy, hogy nem fuldoklunk a vezetékben? Vagy éppen helyi hálózatot építeni anélkül, hogy szét kellene vésni az egész lakást? Na, hát ez a Wi-Fi, annak minden előnyével és hátrányával. A technológia mára a filléres kategóriába került, itt az ideje, hogy mi, földi halandók is megismerkedjünk vele.

Röviden a vezeték nélküli hálózatokról

Kezdetben vala a vezetékes hálózat. Aztán a mobil számítástechnikai eszközök (notebook, *PDA*) elterjedésével együtt a vezeték nélküli technológiák is fejlődésnek indultak. A kezdeti lassú, drága és kis hatótávolságú berendezéseket egyre gyorsabb, nagyobb hatósugarú és jóval olcsóbb eszközök váltották fel. Mára már számtalan, internettel ellátott lakásban megtalálható, mint az otthoni hálózat része, és akkor még nem beszéltünk a kávézókat, éttermeket, szállodákat, utcákat, tereket (!) lefedő úgynevezett ***hotspotokról***, amelyek lehetővé teszik az utca embere számára, hogy notebookjával, *PDA* -jával hozzácsatlakozva elérje a világhálót. Igen, tudom, erre már a mobiltelefon is alkalmas, de a jelenlegi technológia sem sebességben, sem árában, sem tudásában nem éri utol a *Wi-Fi* hálózatok nyújtotta lehetőségeket. Az új *3G* mobil szabvány már sokkal ígéretesebb, de a jelenlegi kis lefedettség mellet a méregdrága szolgáltatási díj is a népszerűsége rovására megy, és persze semmiképp sem fogja kiváltani az otthoni hálózatokat, vagy az irodákon belül elérhető vezeték nélküli kapcsolatokat. Mi is inkább ebbe az irányba vizsgálódunk cikksorozatunkban. Nem tudom megállni, hogy bele ne menjek egy-két unalmas technikai részletbe. Bizonyára többen is tisztában vannak vele, hogy az általunk használt internetes hálózatok többrétegűek. Legalul helyezkedik el a fizikai réteg, legfelül pedig az alkalmazási réteg. Mi felhasználók általában csak ezzel a legfelsővel találkozunk. Maga a vezeték nélküli *Wi-Fi* kapcsolat kizárólag a fizikai réteget érinti, azt cseréli le, vezeték helyett rádiójelekkel történő kommunikációra, természetesen szabványokban rögzített módon. Egy kivétellel 2.4 GHz körüli frekvenciatartományokban működnek, többnyire a törvényben szabályozott jelerősség alatt. Ez azért szükséges, hogy ne tudjunk túl messzire kommunikálni, az ugyanis állami monopólium.

A vezeték nélküli hálózatok felépítése

A vezeték nélküli hálózatok építőkövei az úgynevezett hozzáférési pontok (*AP*, *Access Point*). Ezek sugározzák az *SSID* (*Service Set Identifier*, beállított szolgáltatóazonosító) csomagokat, amelyek alapján a kliensek csatlakozni tudnak a hálózathoz. Lehetőség van persze két kliens kapcsolódásához, hasonlóan, mint fordított *UTP* kábellel, de ez (hasonlóan az *UTP*-s megoldáshoz) nem túlzottan elterjedt. Ilyen esetben az egyik kliens hozzáférési ponttá változik.

Mi is az a Wi-Fi?

Egyesek szerint a *Wi-Fi* rövidítés a *Wireless Fidelity* szóból származik – hasonlóan a *Hi-Fi* szócska mintájára. Mások – például a *Wi-Fi*, mint védjegy birtokosa – szerint nem jelent semmit. Az egyes médiumok is ellentmondanak egymásnak, még a *Wikipedián* is találni olyan szócikket, ahol erre hivatkoznak, nem beszélve rólunk felhasználókról, akik sokan meg vagyunk győződve, sőt egyesek egyenesen tagadják azt, hogy a *Wi-Fi* szó nem jelent semmit. Tény, hogy a *Wi-Fi* egy bejegyzett védjegy, és a védjegytulajdonosok szerint ez csupán az *IEEE 802.11* szabványcsaládra épülő termékek és technológiák gyűjtőneve (kellett egy név, ami jobban hangzik). Az viszont tény, hogy a *Wi-Fi* jelmondatában (*The Standard for Wireless Fidelity*) szerepelt ez a szóösszetétel, már csak az a kérdés, hogy melyik volt előbb... De talán nem is érdekes.
Ha a vezeték nélküli (*wireless*) kifejezést látjuk, általában a *802.11* szabványcsaládra épülő technológiát értjük rajta. A *Wi-Fi* tehát csak egyike a számtalan vezeték nélküli technológiáknak, nem összekeverendő a *GPRS*, *EDGE*, *HSDPA* és egyéb mobiltelefonos szabványokkal, vagy épp a mikrohullámú pont-pont kapcsolatokkal.

A TCP/IP modell öt rétege
5. Alkalmazási réteg
DHCP • DNS • FTP • HTTP • IMAP4 • IRC • NNTP • XMPP • MIME • POP3 • SIP • SMTP • SNMP • SSH • TELNET • BGP • RPC • RTP • RTCP • TLS/SSL • SDP • SOAP • L2TP • PPTP • …
4. Átviteli réteg
TCP • UDP • DCCP • SCTP • GTP • …
3. Hálózati réteg
IP (IPv4 • IPv6) • ARP • RARP • ICMP • IGMP • RSVP • IPSec • …
2. Adatkapcsolati réteg
ATM • DTM • Ethernet • FDDI • Frame Relay • GPRS • PPP • …
1. Fizikai réteg
Ethernet physical layer • ISDN • Modems • PLC • RS232 • SONET/SDH • G.709 • Wi-Fi • …

Több hozzáférési pont összekapcsolásával hálós szerkezetet (***meshed network***) lehet kialakítani például egy nagyobb terület lefedéséhez. Tudni kell ugyanis, hogy az ilyen hozzáférési pontok hatósugara ritkán több száz méternél. Hálós szerkezet esetén a hozzáférési pontok különböző csatornákon sugároznak a rádiójelek interferenciájának elkerülése végett. A csatornák szintén szabványban vannak rögzítve. Ilyen esetekben azonban a legtávolabbi hálózati pontok közötti kommunikáció a közbeesők sávszélességét is használja, azaz erősen romlik az amúgy sem kiemelkedő átviteli sebesség. Ennek elkerülése érdekében mindegyik hozzáférési pontot direkt vezetékes kapcsolattal látják el, tehát nem egymáson keresztül, hanem kábelen kommunikálnak az internet irányába. Az így lefedett területeket nevezzük ***hotspotnak***, amit legjobban talán aktív területnek lehetne fordítani.
Ha jól belegondolunk, ez pont ugyanaz a felépítés, mint amit a vezetékes hálózatok esetében már megszoktunk. Az ***AP*** tölti be a hálózati kapcsoló (switch) szerepét, a vezetékek helyett pedig rádiójeleket használunk. Természetesen az átlagfelhasználó ilyen hálózatokat csak középületekben használ, legtöbben kisméretű otthoni hálózatokban gondolkodunk, például azért, hogy meg tudjuk osztani az internetkapcsolatot a ház számítógépei között, szóval maradjunk mi is ezen a vonalon. Az ilyen hálózatok általában egyetlen (kombinált) hálózati kapcsolóból állnak, ami egyben ***AP*** is. Ez kapcsolódik az internethez, és osztja meg azt „hátrafelé”. Mi „hátulról” az ügyfél számítógépeinkben található vezeték nélküli hálózati kártyánkkal kapcsolódhatunk a hálózat fejéhez.
Egy ilyen ***AP*** által létrehozott „aura” legfeljebb 200Mbit/másodperc (***802.11n*** szabvány) átviteli sebességre képes ideális körülmények között. A gyakorlatban ez azonban nem teljesen így van. Egyrészt a régebbi a legelső (még ma is használatban lévő) szabványok 10 és 55 Mbit -es tartományban dolgoztak, másrészt minél több gép használja az „aurát”, és minél távolabb vagyunk a hozzáférési ponttól, minél több tereptárgy (például fal, mennyezet) kerül a számítógép és az ***AP*** útjába, annál lassúbb az átviteli sebesség. Napi használat során reális az 5-50 Mbit/másodperc vagy annál nagyobb sebesség, ami nem csak az internetezéshez, de egy másik gépen tárolt film on-line megtekintéséhez is elegendő, és tereptárgy ide-oda, azért a kertből is lehet még ellazultan számítógépezni. Az biztos, hogy nem olyan stabil, mint a vezetékes megoldás, de lényegesen kényelmesebb és mobilabb.

A vezeték nélküli hálózati kártyák használata Linux alatt

Ahhoz, hogy igénybe vegyük a kapcsolatot, szükségünk van vezeték nélküli hálózati eszközökre. Ezek egyes notebook-okban alapértelmezetten benne vannak, az alacsonyabb árkategóriás illetve a régebbi darabok azonban nem tartalmazzák. Ezekbe bővítményként 5-10 ezer forint körüli összegért tehetünk ilyen ***PCMCIA***-s vagy ***USB*** csatolófelülettel ellátott úgynevezett ***Wi-Fi*** kártyákat. (Vagy ha asztali gépünk van, akkor az ***USB***-s eszköz mellé használhatjuk a ***PCI***-os, hagyományos módon beépíthető bővítőkártyákat). Eredményül egy antenna lóg ki a gépből, most már csak be kell üzemelni a kártyánkat.
Az esetek egy részében még könnyebb dolgunk van, mintha ***Windows*** rendszeren telepítenénk, itt ugyanis semmi dolgunk, a rendszermag automatikusan betölti a szükséges modulokat (már amennyiben natív módon támogatott a ***Wi-Fi*** kártyánk, amire elég nagy az esély), nekünk csak a hálózat paramétereit kell beállítani. Ne szaladjunk azonban ennyire előre, emlékezzünk meg a kevésbé szerencsés társainkról, akiknek a dolog már majdnem olyan bonyolult, mintha ***Windowst*** használnának, de előtte tegyünk egy kis kitérőt.

Az eszközök támogatottságáról

Kétféleképpen bírhatjuk működésre a ***Wi-Fi*** kártyánkat ***Linux*** alatt. Az egyikről már beszéltem, ez nem is igazi működésre bírás, működik az magától. A ***Linux*** rendszermag minden frissítés során egyre több ilyen kártyát támogat, ún. natív módban. Ez a gyakorlatban azt jelenti, hogy valaki (sokszor a gyártó maga) megírta Linux alá a vezérlőprogramot, ami bekerült a rendszermag moduljaiba, így ha betoljuk a kártyát, csak betöltődik a modul, a kisujjunkat sem kell megmozdítani.
Legegyszerűbben parancssorból ellenőrizhetjük ha kiadjuk az `ifconfig` parancsot. A parancs listázza a számítógépben található hálózati eszközöket. Alapesetben látni fogunk egy `eth0` eszközt, valamint egy `lo` eszközt, és ha ezen felül találunk `wlan0`, `wifi0`, `ath0`, vagy hasonló nevű eszközt, akkor a kártya működik.
Ha mégsem működik, a másik módszer egy olyan szabványosítási kísérlet nyújtotta előny kihasználása, ami egyáltalán nem jellemző a ***Windows*** operációs rendszerre. Történt egyszer, hogy a ***Microsoft*** és a ***3com*** összefogott, és megalkotta az ***NDIS*** (***Network Driver Interface Specification*** – hálózati meghajtófelület kialakítás) felületet, amely a hálózati kártyák egységes vezérlését teszi lehetővé. A ***Windows***

A cikksorozatról

Jelen cikk egy 12 részes, egész éven át futó sorozat első része, amelyben megpróbáljuk az olvasóközönséget az „új” technológia irányába elmozdítani. A *Wi-Fi* eszközök nyújtotta előnyöket legjobban persze az egyszerű otthoni hálózatok esetében tudjuk kihasználni, ezért részletes útmutatást adunk majd az ilyen rendszerek beállításához, üzemeltetéséhez: hogyan kell csatlakozni az internetre, hogyan kell megosztani, milyen eszközökre van ehhez szükség, azok hogyan működnek: egyszóval hogyan kell frappáns otthoni hálózatot építeni kis ráfordítással. Kevesen vannak tisztában a vezetéknélküliség biztonsági kockázatával, ezért nem kerülhetjük ki ezt a témát sem. Ha pedig ezen túl vagyunk, a második szakaszban picit elmozdulunk majd a vállalati megoldások irányába, megnézzük ott miként válthatunk ki egy az internet elérésért felelős linuxos kiszolgálót egy jóval egyszerűbb, jóval olcsóbban üzemeltethető eszközre, hogy hogyan építhetünk virtuálisan szétválasztott helyi hálózatokat, és hogyan könnyíthetjük meg az életünket a nyomtatókiszolgálók használatával. Egy ilyen méretű terv megvalósításához természetesen jó pár hardvereszközre is szükségünk van, amelyet a *D-Link Magyarország Kft.* bocsátott a rendelkezésünkre. Ezeken keresztül mutatjuk majd be a beharangozott folyamatok minden egyes lépését, a lehető legegyszerűbb módon.

jelenleg is ezt használja a legtöbb hálózati kártya esetében. Ha tehát van egy olyan felületünk, amely képes értelmezni az *NDIS* kompatibilis kártyák utasításait, akkor nyugodtan használhatnánk az *NDIS* vezérlőprogramokkal a hálózati eszközöket. Ennek a felületnek a neve: *NdisWrapper*, ami valójában egy egész linuxos projektet takar. Lényege, hogy a hálózati kártyánkat annak windowsos vezérlőjével hajthatjuk meg. Így annyi a dolgunk, hogy letöltjük a kártya windowsos meghajtóprogramját, amit összepárosítunk a kártyával, majd betöltjük az *ndiswrapper* modult. Nem minden kártya használ ám *NDIS*-t, vagy előfordulhat, hogy nem működik rendesen. A kompatibilis kártyák folyamatosan frissülő listája megtalálható az ➲ *http://ndiswrapper.sourceforge.net/mediawiki/index.php/List* oldalon. Ha a kártyánk szerepel a listán, meg vagyunk mentve, de ha nem, az sem jelenti azt, hogy nem működik, csupán még nem próbálta senki, vagy ha próbálta, nem tette közzé az eredményt. Nyissuk meg az eszközkezelőt (szinte minden terjesztésben van ilyen, *Gnome* alatt *HAL Device Managernek* hívják) és keressük meg a hálózati eszközöket. Itt akkor is látszódni fog, ha a rendszer egyébként nem ismerte fel a kártyát. Az itt található részletes adatokból világosan kiderül, milyen hálózati eszközzel rendelkezünk, összevethetjük a listával.

A telepítés NdisWrapper környezet esetében

Töltsük le a windowsos meghajtóprogramot. Ez általában egy *.inf* illetve egy *.sys* fájlból áll – összecsomagolva. Csomagoljuk is ki őket valahová egymás mellé. (például */usr/lib/windriver/*)
Telepítsük az *NdisWrappert*, ami *Ubuntu*/*Debian* alapú rendszereken az

```
apt-get install ndiswrapper-
➥utils
```

paranccsal tehető meg.
Végül állítsuk be az *NdisWrappert*:

```
ndiswrapper -i /usr/lib/
➥windriver/<vezérlőprogram inf
➥fájlja>
```

Esetleg hozzá kell rendelni az eszközt a vezérlőprogramhoz, de ez nem minden esetben szükséges. Az utolsó lépés a modul betöltése:

```
modprobe ndiswrapper
```

Ezen kívül létezik még egy „egyéb” kategória, amit önjelölt felhasználók fejlesztettek, a telepítésük egyedi és nagy szakértelmet kíván, így ha nem akarunk a fentiekkel vacakolni, akkor válasszunk olyan hardvert, amit támogat a *Linux* rendszermag, vagy az *NdisWrapper*, szerencsére bőven válogathatunk. Sajnos nehéz biztosan megmondani, hogy mely kártya támogatott, ugyanis a natívan támogatott *Wi-Fi* kártyák legfrissebb karbantartott listája 2004-ben frissült utoljára, ha pedig az *NdisWrapper* listán nincs fenn a kártya, attól még működhet. Inkább fordított irányból érdemes a problémát megközelíteni: kiszemelünk egy kártyát, és arra keresünk rá a *Google*-on, fórumokon, hogy támogatott-e. Ha ehhez nincs kedve a tisztelt olvasónak, akkor hadd ajánljak én három eszközt, ami biztosan működik, hiszen kipróbáltam.
Az első a *D-Link DWA-142* típusú eszköze, amely *USB* csatolófelülettel rendelkezik. Előnye, hogy nem kell szétszedni hozzá a gépet, elég ha összekötjük az *USB* kapun keresztül. Ha valakinek nem tetszik, hogy egy újabb kütyü lóg ki a számítógépből, akkor válasszon *PCI* csatolófelülettel rendelkező eszközt, amit beszerelhet a számítógépbe. A *D-Link DWA-547* tökéletesen megfelel a célra. Ha viszont notebookba akarunk vezeték nélküli eszközt, akkor választhatjuk *PCMCIA*-s csatolófelületű *DWA-645* típusú kártyát.
A fenti eszközök mind *NDIS* megfelelők, a windowsos vezérlőprogramokkal boldogulni fogunk.
Közös tulajdonságuk továbbá, hogy a legújabb, *IEEE 802.11n* szabványtervezet szerint működnek. Ez a jelenlegi leggyorsabb és legmegbízhatóbb *Wi-Fi* technológia. Az előnyeit persze csak akkor tudjuk kihasználni, ha a hozzáférési pontunk is ilyen, de erről majd később ejtünk szót.
A sorozat következő részében a *Wi-Fi* hálózatok ügyféloldali beállításait tekintjük át, tisztázzuk az alapfogalmakat, bemutatjuk, telepítjük a cikkben javasolt hálózati eszközöket, megismerkedünk egy hasznos segédprogrammal, és beüzemeljük a vezeték nélküli hálózati kapcsolatunkat, amit aztán bármelyik *hotspotnál* vagy hozzáférési pont közelében használni tudunk majd.

KAPCSOLÓDÓ CÍMEK

- ➲ http://en.wikipedia.org/wiki/Wifi
- ➲ http://ndiswrapper.sourceforge.net/wiki
- ➲ http://www.dlink.hu

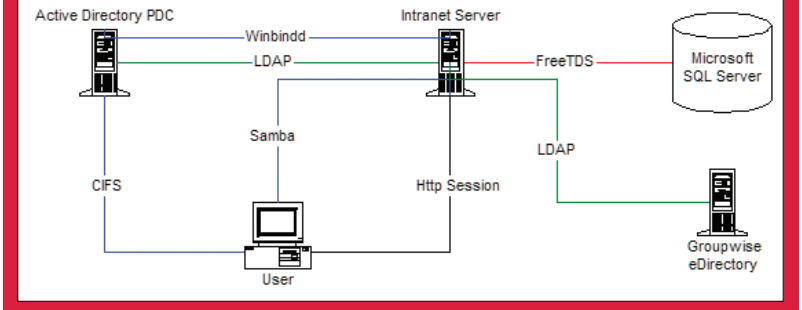
Irodai intranet – kicsiben

Ebben a cikkben értékes tanácsokkal kívánok szolgálni vállalati szolgáltatások felhasználóbarát intranetes integrálásához.

■ Az ***intranet*** már elég régóta meghatározza az életünket. Egyike volt a ***World Wide Web*** alternatív felhasználásának a 90-es évek elején. Noha a web házon belüli alkalmazása vonzó volt, nehézséget jelentett a már meglévő szoftvereszközök integrálása. Emiatt sok ***intranet*** csupán egyszerű hirdetőtáblaként funkcionált. A helyzet mostanra eléggé megváltozott, hiszen a nyílt forrásnak köszönhetően egyszerűbbé és költséghatékonyabbá vált az ***intranet***ek beállítása. Az úgynevezett ***LAMP*** kiváló felületet nyújt számos alkalmazás közös felhasználói felület alá történő integrálásához. Ezt próbáltuk mi is megvalósítani. 1999-ben egy webalapú hirdetőtáblával és egy céges naptárral indult ***intranet***ünk, mely ***Apache*** alól, egy ***Red Hat 6.0*** szerveren kapott helyet. Statikus ***HTML*** oldal volt, amelyet a marketing igazgatónk tervezett és tartott karban. Miután 2002-ben kilépett a cégtől, olyan ***intranet***et szerettünk volna, amely nem egy személytől függ. Ahogy az lenni szokott, újabb és újabb funkciókkal ruháztuk fel a rendszert az évek során. Jelenlegi állapotában nagyon hasznos és felhasználóbarát ***intranet***ünk van mindenféle felesleges vagy statikus tartalom nélkül, amelyek megnehezítenék a karbantartást. Ebben a cikkben saját példákon keresztül fogom bemutatni, hogyan oldottuk meg a négy leggyakoribb integrációs problémát ***LAMP*** környezetben.

Technikai áttekintés

Jelenlegi – 70 alkalmazottat kiszolgáló – ***intranet***ünk egy ***IBM x335***-ös szerveren fut ***Fedora Core 4*** operációs rendszer alatt. A hagyományos ***LAMP*** eszközöket (***Linux, Apache 1.3x, MySQL, Perl***)használjuk ***mod_perl*** modullal kiegészítve a megfelelő teljesítmény érdekében. A szerveren az ***Apache*** mellett e-mail ellenőrző, belső ***DNS szerver***, ***Jabber***, ***Samba*** és pár egyéb szolgáltatás is fut. A közös szervernek köszönhetően egyszerűbb a hálózati meghajtók kezelése és a hálózati forgalom is kisebb. Néhány cégnek ez bizonyára szűk keresztmetszetet nyújtana, de az elveket figyelembe véve hasonlóképp kivitelezhető többszerveres környezetben is. Minden felhasználó ***Windows XP***-t használ és az ***Active Directory*** azonosítja őket. ***GroupWise***-t használunk email fogadásra egy ***NetWare 6***-ot és ***Novell eDirectory***-t futtató szerveren. Van továbbá egy ***Windows NT 4.0*** szerver ***Microsoft SQL*** adatbázissal, amely a munkaidő nyilvántartásért és a számlázásért felel. Az *1. ábrán* látható, hogy függnek össze a szolgáltatások.



■ ***1. ábra*** Céges szolgáltatások kapcsolata

Szerveroldali hitelesítés

A felhasználóknak ne kelljen mindig azonosítaniuk magukat az ***intranet*** felé, ezt már az elején kikötöttük. A szerver automatikusan „tudja” az IP cím és a munkaállomás bejelentkezési adatai alapján, hogy ki használja éppen arról a gépről az ***intranet***et. Mi ezt szerveroldali hitelesítésnek (***SSC – Server-Side Credentialing***) hívjuk. Eredetileg egy saját készítésű kliensoldali alkalmazással oldottuk meg, amely a szerveroldali ***CGI*** szkript kéréseire válaszolt, valahányszor szükség volt azonosításra. Noha működik, túl sok bizalmat vet a kliensoldalba. Egy lehallgatóprogram és egy ***Perl*** szkript segítségével például bármely kliensről be lehet csapni a szervert. Most ***Samba***-t és ***winbindd***-t használunk ugyanerre.
Minthogy az ***intranet*** szerverünk a megbízható belső hálózatra csatlakozik, így minden hálózati ténykedés – ki honnan lépett be – titokban marad. Minden irodai számítógép bejelentkezéskor hozzárendel egy meghajtó-betűjelet a ***Samba*** szerverhez. Ezután a kapcsolódások listája alapján egyértelműen azonosítható, hogy ki honnan lépett be. A meghajtó betűjel csupán a szerveroldali hitelesítés miatt szükséges. Úgy vélem, ez egy

■ *2. ábra* Egy példa a felhasználói profilra az Intraneten

lényeges tulajdonság, hiszen egyszerűbbé teszi a regisztráció és belépés nélküli böngészést. A felhasználóknak már így is elég sok felhasználónevet és jelszót kell észben tartaniuk.
A szerveroldali hitelesítés beállítása nagyban függ a felhasználók azonosítási módjától. Mi az ***Active Directory***-t használjuk, így azzal fogom bemutatni. Az ***Active Directory*** bosszantó egy kicsit(nem túl meglepő), hiszen nem tárol információkat a kapcsolatokra vonatkozóan. Hagyományos ***RPC hívásokat*** kell használnunk a ***Samba*** net parancsával, hogy megbízható eredményeket kapjunk. A szerveroldali hitelesítő szkriptünket ***smbconn.sh***-nak hívjuk, és valahogy így néz ki:

```
#!/bin/sh
net status sessions parseable  \
| grep -i "\\\\$1\\\\"         \
| sed 's/^.*\\\(.*\)\\.*\\.
➥*\\.*$/\1/g'   \
| sed 's/DOMAIN+//g' | tr -d ' '
```

Elég egyszerű, nem? Csupán a `DOMAIN`-t kell kicserélni az általunk használt ***Active Directory*** domainjére. A szkript az általunk megadott IP-ről belépett felhasználói objektumot adja válaszként. Ez az objektum az ***Active Directory sAMAccountName*** mezőjét takarja. Ezzel az információval felvértezve már indíthatunk ***LDAP*** keresést, amely visszaadja a felhasználó teljes nevét, de akár bármi mást, amire szükségünk van. Ezt a szkriptet az *1. Lista* tartalmazza.
Első paraméterként a felhasználó ***sAMAccountName*** értékét várja, második paraméterként pedig egyéb opcionális attribútumot adhatunk meg. Ha nem adunk meg opcionális attribútumot, akkor a felhasználó teljes nevével tér vissza. Ehhez használhatunk egyéni ***mod_perl*** kiegészítést, így az információ mindig elérhető lenne, de úgy tűnik, hogy ez a legtöbb esetben feleslegesen bonyolítana a dolgokon. A mi esetünkben csupán pár részre korlátozódik a dolog, így minden ***CGI*** programot igény szerint futtatunk. A szerveroldali azonosítás kérése így néz ki az egyik ***CGI*** programunkban:

```
##: Get this connection's user
➥credentials
my $remoteip=$ENV
➥{'REMOTE_ADDR'};
open(SMBCONN,"smbconn.sh
➥$remoteip |");
my $cn=<SMBCONN>;
$cn=~s/\s+//g;   ##: Strip
➥whitespace
close(SMBCONN);
open(GETEMPINFO,"getempinfo.pl
➥$cn |");
my $username=<GETEMPINFO>;
close(GETEMPINFO);
if($username eq "") {
  $username="Guest";
}
```

Ez a kódrészlet a felhasználó `sAMAccountName` értékét adja vissza a `$cn` változóban és a teljes nevét a `$username` változóban. Ha a `$username` tartalma `Guest`, akkor sikertelen volt a keresés vagy a kliens számítógépen nincs belépett felhasználó. Most már minden létfontosságú információnk megvan, hogy eldöntsük, jogosult-e a felhasználó megtekinteni az oldalt vagy sem. Ezzel az is lehetővé válik, hogy a felhasználó számára egyedi oldalt jelenítsünk meg. Ezt az ***intranet*** oldalunk ***index.cgi*** fájljának egy részével demonstrálnám:

```
##: My Intranet section
my $mint="";
if(($username eq "Guest") ||
➥($username eq "")) {
  open(EMPSNAP,"./random-
  ➥employee.pl 2>&1 |");
  my @snap=<EMPSNAP>;
  close(EMPSNAP);
  $mint.=join("\n",@snap);
  chop($mint);
} else {
  $mint.=&get_emp_card($cn);
  $mint.="<b>E-Mail
➥Controls:</b><br>\n";
  $mint.="<a href=
  ➥'selfserv.cgi'>My Mail
  ➥</a>\n";
  ...
}
...
print STDOUT $mint;
```

Látható, hogy ellenőrizzük, vajon az oldalt megtekintő személy azonosított felhasználó-e. Ha nem, akkor véletlenszerűen megjelenítjük véletlenszerűen egy munkatársunk fotóját és adatlapját az oldal adott részén. Ha azonosított, akkor az *LDAP*-ból lekérjük a megfelelő információkat és összeállítjuk a *My Intranet* (az *Intranet*em) oldalt. Ezen az oldalon a felhasználó módosíthatja a profilját, megnézheti a levelezését, stb. A `get_emp_card($cn)` eljárás csupán kikeresi a felhasználó adatait az *Active Directory*-ból és egy *HTML* formátumú részt jelenít meg (2. *ábra*).

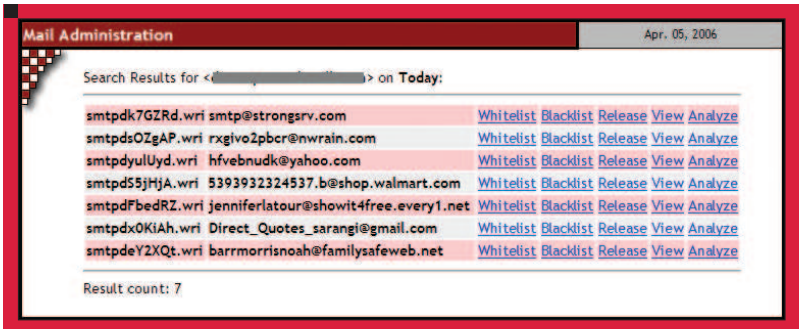
Az Active Directory integrálása

Az *intranet*ünk másik hasznos bővítése: *Active Directory* felhasználói adatbázis elérése *LDAP*-on keresztül. Ezt használjuk a cégnél dolgozó összes munkatárs listázásakor. Az adatbázist valós időben építjük fel, valahányszor szükség van rá, ezáltal tehermentesítve a rendszergazdákat. Valahányszor új felhasználó kerül az *Active Directory*-ba, azonnal látszik az *intranet*es listában is. Felhasználóink maguk szerkeszthetik adataikat, melyek aztán egy *CGI szkript* segítségével kerülnek be az *Active Directory*-ba. Az eljárás magától értetődő, azonban néhány dolgot figyelembe kell venni. Bemutatom lépésről-lépésre, mi hogyan csináltuk.
Először is létrehoztuk az *Active Directory*-ban a *proxyuser* felhasználót. Ezzel a felhasználónévvel azonosítja magát szkriptünk az *LDAP* felé. A *proxyuser* írhatja és olvashatja azokat a felhasználói objektumokat, melyek ou=Domain Users konténerében vannak. Ezt az *Active Directory*-n belül kell végre hajtanunk. A *CGI* programjainkhoz *Perl*-t használunk, így az *LDAP* elérése a *Net::LDAP* könyvtáron keresztük történik. *CGI* szkriptünkkel így kapcsolódhatunk az *Active Directory*-hoz:

```
##: Active Directory connection
use Net::LDAP;
my $ldap=Net::LDAP->new
➥('adserver.domain.com');
my $mesg=$ldap->bind
➥('proxyuser@domain.com',
            password=>
             ➥'proxyuser'  );
```

1. Lista getempinfo.pl

```
#!/usr/bin/perl -w
use Net::LDAP;
use strict;
my $cn=$ARGV[0] || "none";
my $attr=$ARGV[1] || "none";
##: If nothing was given on command line then return
if($cn eq "none") {
  print STDERR "ERROR: No LDAP cn given\n";
  exit(1);
}
##: Bind anonymously to the ldap database
my $ldap=Net::LDAP->new('directory.domain.com',timeout=>5)
  or die "Couldn't connect to directory server.\n";
my $mesg=$ldap-
>bind('proxyuser@domain.com',password=>'proxyuser')
  or die "Couldn't connect to directory server.\n";
##: Query LDAP to get a list of employees
if($attr ne "none") {
  $mesg=$ldap->search( base=> "ou=Domain
Users,dc=domain,dc=com",
                       filter=> "(sAMAccountName=$cn)",
                       attrs=> ['givenName','sn',"$attr"] );
} else {
  $mesg=$ldap->search( base=> "ou=Domain
  ➥Users,dc=domain,dc=com",
                       filter=> "(sAMAccountName=$cn)",
                       attrs=> ['givenName','sn'] );
}
my $count=$mesg->count();
($count==1) or die "Error: LDAP enumeration error.";
my $entry=$mesg->entry();
my $value;
my @values;
if($attr ne "none") {
  $value="";
  @values=$entry->get_value("$attr");
  my $i=1;
  for(@values) {
    if($i>1) {
      $value.="/$_";
    } else {
      $value.=$_;
    }
    $i++;
  }
} else {
  $value=($entry->get_value('givenName')." ";
  $value.=$entry->get_value('sn'));
}
##: See if that attribute was defined for the given cn
if(!(defined($value))) {
  print STDERR "ERROR: That attribute was not defined.\n";
  exit(1);
}
$mesg=$ldap->unbind;
print("$value\n");
```



■ *3. ábra* Műveletek az elfogott spamekkel

Az ***Active Directory*** ilyen formátumban várja a felhasználónevet. Ez egyike azoknak, amelyet az ***Active Directory LDAP*** felülete igényel. Kapcsolódás után már lekérhetjük a `ou=Domain Users` konténerben található felhasználók listáját:

```
##: Query LDAP to get a list of
➥employees
my $basedn="ou=Domain
➥Users,dc=domain,dc=com";
my $filter="(objectClass=user)";
$mesg=$ldap->search(
  base=> $basedn,
  filter=> $filter,
  attrs=> ['givenName','sn',
   ➥'mail',
        'telephoneNumber',
         ➥'streetAddress',
        'l','st','department',
         ➥'postalCode',
        'employeeNumber',
         ➥'homePhone',
        'title',
         ➥'sAMAccountName' ]
  );
```

Megkapjuk a listát az összes felhasználóról az összes kapcsolódó attribútummal egyetemben. Tovább finomíthatjuk a keresést azzal, hogy csupán azokat a dolgozókat listázzuk, akiknek a vezetékneve a ***CGI szkript*** által kapott betűvel kezdődik. Így címjegyzékszerű végeredményt kapunk és nem kell mind a 70 felhasználót egyszerre megjeleníteni. Ha a ***CGI szkript*** nem kapott kezdőbetűt, akkor az *„a”* betűs neveket jeleníti meg:

```
##: Get letter requested in the
➥URL
my $letter;
$letter=param('letter') || "a";
...
my $filter="(&(objectClass=user)
➥(sn=$letter*))";
```

Ha az Olvasó kevéssé járatos az ***LDAP*** keresésekben, akkor mindenképp érdemes átolvasni az ***RFC-2254***-es dokumentumot. Ez az a pont, ahol végignézzük a kapott listát és igény szerint csinosítjuk. Lekérdezhetjük a munkatársakhoz tartozó ***sAMAccountName*** értéket is. Ha a találat megegyezik az azonosított személlyel, akkor kitehetünk egy linket, és arra kattintva módosítani tudja az adatait. Ez valahogy így néz ki:

```
##: Display the directory
foreach my $entry ($mesg->
➥sorted('sn')) {
  my $san=$entry->
   ➥get_value('sAMAccountName');
  $empdir.="<div class=
   ➥'empcard'>";
  if(lc($cn) eq lc($san)) {
    ##: This is our man.
➥ Add a button.
    $empdir.="<a href=
     ➥'empedit.cgi'>Edit</a>";
  }
  $empdir.="<span id='name'>";
  $empdir.=$entry->get_value
   ➥('givenName')." ";
  $empdir.=$entry->get_value
   ➥('sn');
  $empdir.="</span><br>";
  $empdir.="<span id='title'>";
  $empdir.=$entry->get_value
   ➥('title').";
  $empdir.="</span><br>";
  ...
  $empdir.="</div>";
}
print STDOUT $empdir;
$mesg=$ldap->unbind();
```

SpamAssassin és E-mail integrációja

2001-ben állítottuk üzembe a cég email átjáróját, és azóta is azt használjuk. A ***Linux Journal 2001*** decemberi számában írtam róla egy cikket. Rengeteget változott azóta, de az alapok ugyanazok maradtak. Egyszerűen tárol, ellenőriz és továbbít. Minthogy mindez a ***Linux***os szerveren belül bonyolódik, így a ***Windows*** felhasználók nem férnek hozzá a személyes ***SpamAssassin*** konfigurációjukhoz. Ezt néhány ***CGI szkript***tel oldottuk meg, melyekkel a felhasználók testre szabhatják ***SpamAssassin*** beállításaikat. A felhasználók beállításaikat a ***My Intranet*** megfelelő részén érik el (*2. ábra*). A lenyíló menüből kiválaszthatják, melyik napra kíváncsiak. A gombra kattintva elindul a ***selfserv.cgi*** szkript. A szkript nem vár semmiféle azonosítást, hiszen azt már korábban a szerveroldali hitelesítéssel megoldottuk. A kezdeti lekérdezés után újra meghívjuk a ***getempinfo.pl***-t az alábbi módon, hogy kiderítsük az email címét:

```
##: Get this user's email address
open(GETEMPINFO,"-|",
➥"getempinfo.pl",$cn,"mail");
my $searchstring=<GETEMPINFO>;
close(GETEMPINFO);
```

Ezután a `$searchstring` változót – mint reguláris kifejezésünk alapját – fogjuk a */spam* könyvtárban történő kereséskor felhasználni. Minthogy az ***Active Directory*** email mezője szabadon módosítható, így az esetleges elgépelések ellen védekeznünk kell:

```
##: Make sure this email address
is valid
unless($searchstring=~/^[a-z]*\
➥@domain\.com$/) {
  print STDOUT "Content-Type:
➥text/plain\n\n";
  print STDOUT "Access Denied:
➥Your identity on \
    the network can't be
➥verified.\n";
  return(0);
}
```

Ha sikeresek voltak az ellenőrzések, akkor a szkript megjeleníti a kért nap spamforgalmát. Minden elem mellett számos link található. Ezek lehetővé

teszik a spam átengedését, a feladó engedélyezését vagy tiltását, *SpamAssassin* jelentés elkészítését vagy csupán az email tartalmának megjelenítését. Hogy a felhasználó mit tehet és mit nem tehet, azt jelen esetben is a szerveroldali hitelesítés segít eldönteni. Részletesebben nem tárgyalom a dolgot, hiszen a szkriptek csupán a fájlok megfelelő áthelyezését végzik. Szeretnék viszont a (engedélyező és tiltó) listákról bővebben beszélni.
A *SpamAssassin* felhasználói beállításai a felhasználó könyvtárában, a *.spamassassin/user_prefs.cf* fájlban találhatóak. Hétköznapi esetben, ha linuxos a levélszerver, akkor ez megfelelő. A mi esetünkben azonban nem. A *Linux* szerverünk csupán a be és kimenő levélforgalmat vizsgálja, nem rendelkezik információval a felhasználókról és email címeikről. A megoldás érdekében cselhez folyamodtunk. A *SpamAssassin* rendszerszintű konfigurációja a */etc/mail/spamassassin/local.cf* állományban található, amelyet minden induláskor beolvas. Valójában az összes *.cf* végződésű állományt végignézi a */etc/mail/spamassassin* könyvtárban. Ezt a javunkra fordíthatjuk és *CGI szkript*ünkkel itt hozzuk létre a felhasználói listákat `$cn_prefs.cf` formátumban. A spamd-t cron-nal óránként újraindítjuk memória-felszabadítás végett, így ezzel nem lesz gond. Amennyiben ezt a megoldást használjuk, figyeljünk oda, nehogy a felhasználók `*@hotmail.com` vagy ehhez hasonló *SpamAssassin* feltételeket hozzanak létre. Ezek ugyan személyes állományok, mégis kihatással vannak a *SpamAssassin* globális működésére, hiszen a fő konfigurációs könyvtárban találhatóak.

Microsoft SQL Server integrálása

Cégünk a *CPAS* munkaidő nyilvántartó és számlázó rendszert használja. Ez a szoftvercsomag tartalmazza az összes partnerünk és számlánk információit, valamint a marketing igazgatónk által a reklámkampányoknál felhasznált adatokat is. Szerettük volna biztosítani munkatársaink számára a hozzáférést néhány alapvető adathoz. Mindezt az ügyfélszolgálat megkerülésével, hogy időt takarítsunk meg. A *CPAS Microsoft SQL*-t használ, így az integráláshoz segítségül kellett hívnunk a *FreeTDS*-t és a *DBD::Sybase Perl* modult.
A beállítás pár egyszerű lépés. Először le kellett töltenünk a legfrissebb *FreeTDS* csomagot az *Internet*ről, és ki kellett csomagolnunk.
A könyvtárba belépve kiadtuk az alábbi parancsokat:

```
> ./configure -prefix=/usr/
➥local/freetds
> make
> su -c 'make install'
```

Ez a megfelelő helyre telepítette a *FreeTDS*, így egyszerűbb dolgunk lesz a *Sybase* modulnál. Következő lépésként letöltöttük a *CPAN*-ról a *DBD:Sybase* csomagot. Rendszergazdaként az alábbi parancsokat adtuk ki:

```
> perl -MCPAN -e shell
> install DBD::Sybase
```

Ha néhány teszt hibát jelez, nyugodtan folytassuk a telepítést, a csomag szerzője szerint ez elég gyakori.
A program ezután már a gépen van, de meg kell ejtenünk még a *FreeTDS* konfigurálását is. A konfigurációs fájlban adjuk meg az adatbázisok adatait, amelyekhez kapcsolódni szeretnénk. Az állomány jól dokumentált és a logikáját is elég könnyű átlátni. Így néz ki egy bejegyzés:

```
[JACKSON5]
   host = jackson5.domain.com
   port = 1433
   tds version = 4.2
```

Ha a *FreeTDS*-t beállítottuk, akkor már elérjük *Perl*-ből az adatbázist a klasszikus *DBI* interfészen keresztül. Íme egy példa, amelyben a *JACKSON5* nevű *Windows* szerverünkön lévő *concerts* adatbázishoz kapcsolódunk:

```
#!/usr/bin/perl -w
use DBI; $ENV{'SYBASE'} = '/
➥usr/local/freetds';
$dbh = DBI->connect
➥('dbi:Sybase:server=JACKSON5',
➥ 'username', 'password')
   or die 'connect';
$dbh->do("use concerts");
```

Vegyük észre: környezeti változóként meg kell adnunk a *FreeTDS* könyvtárát is mielőtt csatlakoznánk. A változó közli a *DBD::Sybase* modullal, hol találja a *FreeTDS* modulokat. Ezután már a megszokott módon használhatjuk a *DBI* felületet.
Ha eddig *MySQL*-lel dolgozott a kedves Olvasó, mindenképp nézze át alaposabban a *Microsoft SQL Server* használatára vonatkozó információkat. Néhány dolog igencsak más a megszokottakhoz képest.

Zárszó

Reményeim szerint sikerült pár hasznos ötletet adnom, hogyan lehet hatékonyan integrálni *intranet*té a legalapvetőbb szolgáltatásokat, amelyek egy irodában előfordulhatnak. Az *intranet* sokkal több egy elektronikus hirdetőtáblánál. Megfelelő *intranet* időt takarít meg a rendszergazdák számára, de munkatársak is hatékonyabban végezhetik munkájukat. A felhasználók számára ugyanis sokkal egyszerűbb webböngészővel dolgozni, mint parancssorból. Kovácsoljunk ebből előnyt és az *intranet* cégünk értékes befektetésévé válik.

Linux Journal 2006., 151. szám

Dave Jones a Pearce, Bevill, Leesburg & Moore cég informatikai igazgatója Birmingham-ban, Alabama államban. Nyolcéves hálózatadminisztrátori múlttal rendelkezik. Szabadidejében webes naplót ír és programokat fejleszt
➲ www.sector62.com

KAPCSOLÓDÓ CÍMEK

FreeTDS
➲ www.freetds.org

RFC-2254
➲ ftp.rfc-editor.org/in-notes/rfc2254.txt

E-mailes vírusok szűréséről szóló cikkem
➲ www.linuxjournal.com/article/4882

CPAS
➲ www.cpasoftware.com

Pearce, Bevill, Leesburg & Moore P.C.
➲ www.pearcebevill.com

A Linux NTFS fájlrendszerrel kapcsolatos eszközei

Elképzelhető, hogy a legjobb Windows segédprogramok közül néhány már ott lapul a Linux partíciónkon...

Linux alól több tucat különféle fájlrendszerhez férhetünk hozzá, legyenek azok akár a helyi gépünkön, akár egy hálózati kiszolgálón. Hogy rögtön a lényegre térjek, *Linux* alatt számos olyan eszköz létezik, amelyekkel *Windows* fájlrendszereken végezhetünk műveleteket, vagy akár meg is javíthatjuk azokat, kisegítve ezzel a bajból a *Windows*-t.

Ezeknek a segédeszközöknek egy jelentős része a *Linux-NTFS* projekt terméke. Vannak köztük olyanok, amelyek képességei már-már csodálatosnak nevezhetők. Van például olyan, amelyikkel átméretezhetünk egy *NTFS* partíciót. Másokkal egyedi fájlokat kezelhetünk, míg megint másokkal klónozhatunk egy *NTFS*-re formázott lemezrészt. *Linux* alól megoldható például teljes *Windows* telepítések biztonsági mentése, de készíthetünk olyan lemezképeket is, amelyekről hálózaton keresztül klónozhatók a windowsos munkaállomások. És mivel valamennyi ilyen eszköz *Linux* alatt fut, örökli mindazt a hatékonyságot, amivel a *UNIX* rendszer csak rendelkezhet. Használhatjuk ezt az eszközkészletet akár egyetlen olyan gépen is, amelyre két operációs rendszer van telepítve, de igazán hasznosnak azok érezhetik, akik egy nagyobb hálózatot tartanak karban. Mivel *Linux* alatt van átirányítás, csövek, és persze írhatunk héjprogramokat is, könnyedén automatizálhatunk számos olyan adminisztratív feladatot, amelyek végrehajtása *Windows* alatt igen fáradságos és persze unalmas lenne.

Telepítés

Az említett segédprogramok számos helyen megtalálhatók, és a támogatásuk is egészen kiváló. Gyakorlatilag valamennyi olyan *Linux* terjesztésre létezik belőlük megfelelő csomag, amely tartalmaz valamilyen csomagkezelőt, sőt, még a *Knoppix CD*-n is megtalálhatók. Ugyanakkor a legtöbb terjesztés telepítője alapértelmezésként úgy állítja be a jogosultságokat, hogy ezeket az eszközöket csak a rendszergazda legyen képes futtatni. Aki nem tudja, hogy a kérdéses csomag fönt van-e már a gépén, próbaképpen adja ki a `man ntfsprogs` parancsot. Ebből egyértelműen kiderül, hogy legalább a dokumentációja föltelepült-e.

Még ha a próba nem is járt sikerrel, a fentiek miatt minden bizonnyal egészen egyszerű lesz a megfelelő csomag telepítése. *SUSE*, *Debian*, *Ubuntu* és *Gentoo* alatt a csomag neve a korábban említett *ntfsprogs*. Erre kell rákeresnünk a telepítési adathordozón, és föltelepíteni, ha megtaláltuk. Egyes rendszereken a csomagban benne van az összes, az *NTFS* fájlrendszerek kezelésére használható eszköz, míg másokon bizonyos dolgok hiányoznak. A *Debian Etch* például tartalmazza az `ntfsmount` eszközt, míg a *Sarge*-ben található csomag nem. A *Red Hat*/*Fedora* terjesztés licenszelési problémák miatt egyáltalán nem tartalmazza magát a csomagot sem, viszont az interneten számos helyen megtalálhatjuk annak kifejezettem ezekhez a rendszerekhez idomított változatát, olyannyira, hogy az még a *Linux-NTFS* projekt hivatalos webhelyéről is elérhető. Aki a legfrissebb dokumentációt vagy forráskódot szeretné letölteni, annak természetesen szintén a hivatalos weblapot (➲ *http://www.linux-ntfs.org*) tudjuk ajánlani. A kapcsolód dokumentáció minden olyan követelményt és megoldást is tartalmaz, ami a programok forrásból való lefordításához szükséges lehet. Utóbbi módszer természetesen a Linux bármely változatán működik, sőt, ha valaki ki szeretné próbálni a legújabb szolgáltatásokat is, annak feltehetőleg nem is lesz más választása, mint hogy forráskódból kezdjen el dolgozni. Egyes rendszereken persze gondot okozhat, hogy ilyenkor óhatatlanul kikerüljük a csomagkezelő adminisztrációs rendszerét.

Egy fontos megjegyzés mindazoknak, akik a fenti figyelmeztetés ellenére mégis a forráskód mellett döntöttek: a csomag lefordításához szükség van a *FUSE* könyvtárra, aminek a forrását a ➲ *fuse.sourceforge.net* webhelyről szerezhetjük be. A *Linuxnak* van ugyan egy beépített *NTFS* meghajtója, az *NTFS* segédeszközök azonban egy másikra támaszkodnak, ami szintén benne van a csomagban. A „nem natív” meghajtó az, ami a *FUSE* könyvtárra támaszkodik, és amely számos extra szolgáltatást nyújt a „kincstári” `ntfsmount`-hoz képest. Igaz ugyan, hogy valamivel lassabban működik, mint a kernel részét képező *NTFS* modul, de ez talán nem olyan nagy probléma. Ugyanakkor fontos megjegyezni, hogy szükségünk lesz a *FUSE* kernelmodul betöltésére is. Aki tehát telepíteni szeretné a *FUSE* könyvtárat, az először is töltse le

a legfrissebb forráskódot egy megfelelő könyvtárba, amely akár azzal is azonos lehet, amely az ***ntfsprogs*** csomagot tartalmazza. Ami a fordítás folyamatát illeti, abban nincs semmi különleges. Egyszerűen csak ki kell adnunk a `configure`, `make`, `make install` parancsokat, utóbbit persze root-ként bejelentkezve (a módszer amúgy mára gyakorlatilag szabványosnak is nevezhető, a verziószám pedig természetesen változhat, hiszen a rendszert folyamatosan fejlesztik):

```
tar -xzvf fuse-2.5.3.tar.gz
cd fuse-2.5.3.tar.gz
./configure
make
make install
```

A ***FUSE*** könyvtár és a kernelmodul telepítése amúgy nem föltétlen szükséges abban az esetben, ha csupán olvasási jogot szeretnénk az ***NTFS*** partíciókhoz. Ennek az az oka, hogy – amint az korábban is említettem – a ***Linux*** kernel eleve tartalmaz egy az ***NTFS*** fájlrendszerek olvasására való modult, ami együttműködik a közönséges `mount` paranccsal, ráadásul még gyorsabb is. Ugyanakkor nem rendelkezik mindazokkal a kiegészítő szolgáltatásokkal, amelyek az `ntfsmount` igen.
A következő lépés az ***ntfsprogs*** forráskódjának letöltése. Ezt is helyezzük el egy megfelelő könyvtárban, majd – rootként bejelentkezve – fordítsuk ugyanazzal a módszerrel, amivel az imént a ***FUSE*** csomagot (a verziószám természetesen ismét eltérhet az itt megadottól):

```
tar -xzvf ntfsprogs-
➥1.13.1.tar.gz
cd ntfsprogs-1.13.1
./configure
make
make install
```

Ha rendszerünk nem tartalmazza a ***FUSE*** könyvtárat (még ha a ***FUSE*** modul fut is) a `configure` parancs futtatása közben a következő figyelmeztetést fogjuk kapni:

```
checking for FUSE_MODULE...
➥configure: WARNING: \
ntfsmount requires FUSE
➥version >= 2.3.0
```

Ez persze nem végzetes, az ***NTFS***-t kezelő eszközöket ettől még ugyanúgy lefordíthatjuk, az `ntfsmount` programot azonban nem.
Aki ***Red Hat/Fedora*** rendszert használ, annak a gépén még a megfelelő kernelmodul sem lesz fönn. Ilyenkor két választásunk van: vagy átállunk egy olyan speciális kernelre, amely tartalmazza az ***NTFS*** modult, vagy mi magunk lefordítjuk a ***FUSE*** könyvtárat a bemutatott módon.

Lássuk a programokat

Most tehát ott tartunk, hogy az Olvasó vagy sikeresen föltelepített gépére az ***ntfsprogs*** csomagot a fent bemutatott módszerek valamelyikével, vagy egyszerűen csak megtalálta azt a az lapértelmezésként telepített csomagok listájában.
Az ***ntfsprogs*** súgóoldala tartalmazza a rendelkezésünkre álló segédprogramok teljes listáját. Lássuk közülük most csak a legfontosabbakat:

- `mkntfs(8)` – Egy partíció megformázása ***NTFS*** fájlrendszerrel.
- `ntfscat(8)` – Egy ***NTFS*** fájlrendszeren található fájl tartalmának kiküldése a szabványos kimenetre.
- `ntfsclone(8)` – Egy ***NTFS*** partíció tartalmának hatékony klónozása, létrehozása, visszaállítása vagy mentése lemezképbe/ből.
- `ntfscluster(8)` – Felderíti az ***NTFS*** partíció egy adott szektorának vagy klaszterének tulajdonosát.
- `ntfscp(8)` – Egy ***NTFS*** partíción található fájl felülírása.
- `ntfsfix(8)` – Egyes gyakori hibák ellenőrzése és javítása, A ***LogFile*** törlése, illetve a ***Windows*** utasítása arra, hogy a következő bootolásnál végezzen átfogó ellenőrzést az adott fájlrendszeren.
- `ntfsinfo(8)` – Információkat jelenít meg egy teljes ***NTFS*** partícióról, vagy egy azon található könyvtárról vagy fájlról.
- `ntfslabel(8)` – Az ***NTFS*** partíció címkéjének megjelenítése, vagy beállítása.
- `ntfsls(8)` – Egy ***NTFS*** lemezrészen található könyvtár tartalmának listázása.
- `ntfsmount(8)` – ***FUSE NTFS*** modul.
- `ntfsresize(8)` – ***NTFS*** partíció átméretezése adatvesztés nélkül.
- `ntfsundelete(8)` – Törölt fájlok visszaállítása egy ***NTFS*** fájlrendszeren.

A listában szereplő segédprogramok közül némelyiknek kétségtelenül olyanok a funkciói, hogy azok inkább a fejlesztőknek, semmint a hálózati rendszergazdáknak, vagy a saját, két operációs rendszert tartalmazó gépüket karbantartó otthoni felhasználóknak lehetnek hasznosak. Ugyanakkor sok olyan is van, ami valóságos életmentő. Ez utóbbi kategória talán legfontosabb képviselője az `ntfsclone`.

Az NTFS eszközök használata

Ahhoz, hogy kipróbálhassuk az `ntfsclone` működését, először is szükségünk lesz egy ***NTFS*** fájlrendszerre, meg persze egy másikra, ami elég nagy ahhoz, hogy elférjen rajta az előbbi lenyomata. Utóbbira első közelítésben egy ***ext2***, ***ext3***, ***xfs*** vagy ***ReiserFS*** fájlrendszert javasolnék.
Az `ntfsclone` dokumentációja ugyan fölhívja a figyelmet arra, hogy a ***ReiserFS*** kifejezetten lassú tud lenni, ha szétszórtan elhelyezkedő fájlokat kell kezelni, én azonban ezt nem igazán tapasztaltam az újabb változatokkal kapcsolatban. Használhatunk természetesen külső meghajtót is, ha van rajta elég hely. Arra azonban ügyeljünk, hogy bizonyos műveletek végrehajtásához akkora szabad hely szükséges, mint amekkora az egész ***Windows*** partiónk. Amennyiben a külső meghajtó ***FAT32*** fájlrendszert tartalmaz, szintén kellemetlen meglepetésben lehet részünk, hiszen itt van egy meglehetősen szűk felső korlátja a legnagyobb tárolható fájlnak.
A ***FAT32*** általánosságban tehát nem igazán alkalmas erre a célra. Végezetül – bár ez meglehetősen triviális – a gépünkön ***Linuxnak*** kell futnia, tehát na nincs ilyen telepített rendszerünk, akkor használjunk valamilyen Live terjesztést, például egy ***Knoppix CD***-t.
Az `ntfsclone` fenti leírásában azt állítják, hogy a program hatékonyan teszi a dolgát, és ez nem csak valami hangzatos reklámszöveg. Olyannyira nem, hogy az újabb gépeken egy ***Windows XP***-t futtató munkaállomás teljes

klónozása nem tart tovább pár percnél. Ha a klónozni kívánt ***NTFS*** fájlrendszer az első ***IDE*** merevlemez sorrendben első partícióján található, a mentéshez használt ***Linux*** pedig ugyanazon a gépen fut, akkor a következő parancs egyetlen fájlba „klónozza le" a teljes ***Windows XP***-t:

```
ntfsclone /dev/hda1 -O /usr/
➥local/backup/ntfs.img
```

A parancsban szereplő `-O` kapcsoló arra utasítja a programot, hogy írja felül a képfájlt, de ha az eredetileg nem létezik, természetesen automatikusan lére is jön. Ha ilyen módon futtatjuk a programot, akkor az a képfájlt nem tömöríti, olyannyira nem, hogy azt később akár be is csatolhatjuk a visszacsatoló (***loopback***) eszközön keresztül. Ez utóbbihoz először is hozzuk létre magát a csatolási pontot:

```
mkdir /usr/local/backup/mtpt
```

Ha ez megvan, akkor az `ntfsmount` segítségével pontosan ugyanazt a szintaxist használva fűzhetjük be a virtuális ***NTFS*** fájlrendszerünket, mint ahogy azt bármely más képfájllal tennénk:

```
ntfsmount -o loop /usr/local/
➥backup/ntfs.img \
 /usr/local/backup/mtpt/
```

Az `ntfsmount` alapértelmezésként írható és olvasható állapotban fűzi be a megadott fájlrendszert, ami azt jelenti, hogy könnyedén másolhatjuk vagy törölhetjük is a rajta levő fájlokat. Persze ezen a ponton meg kell majd küzdenünk mindazokkal az „apró" problémákkal, amelyek mindig előjönnek, valahányszor különböző operációs rendszerek között próbálunk mozogni. Ha például konfigurációs fájlokat kezdünk szerkeszteni, mindenképpen ügyeljünk rá, hogy az adott rendszeren pontosan milyen karakterkombináció is jelzi a sor végét. Mellesleg pedig meggyűlhet a bajunk az eltérő karakterkódolásokkal is, de ez manapság szinte természetes. Ha a becsatoláshoz a natív ***NTFS*** meghajtót akarjuk használni, természetesen a hagyományos `mount` parancsot kell futtatnunk, de a szintaxis gyakorlatilag azonos:

```
mount -o rw,loop,nls=utf8 -t
➥ntfs \
 /usr/local/backup/ntfs.img \
 /usr/local/backup/mtpt/
```

Figyeljük meg, hogy itt már a becsatoláskor megadtuk a ***Windows*** számára természetes karakterkódolást is. Persze ez még nem véd meg bennünket mindentől. Itt is kell némi szakértelem, bölcs tapasztalat, meg némi éberség, ha teszem azt elkezdjük ***Emacs***-ból átszerkeszteni a ***boot.ini*** fájlt. Az az igazság, hogy éberség jobbra, gyakorlat balra, az ilyesmit még mindig biztonságosabb ***Windows*** alól elvégezni, esetleg a ***Notepad*** segítségével ***Wine*** alatt.
Ha írási és olvasási jogra is szükségünk van az ***NTFS*** fájlrendszeren, akkor a becsatolásra másodikként bemutatott módszerrel vagy szerencsénk lesz, vagy nem, attól függően, hogy pontosan melyik kernelverziót használjuk a gépünkön. Ezzel kapcsolatban csak annyit tudok mondani, hogy a natív ***NTFS*** meghajtó kezelése és egész működése némiképp trükkös. Ha valami nem tetszik neki, általában küld egy figyelmeztetést, és ezzel párhuzamosan visszakapcsol csak olvasható módba. Ami pedig a régebbi változatait illeti, nos azok némelyike egyenesen életveszélyes, ha engedélyezzük neki az írást is.
A fájlrendszer lecsatolását mindkét esetben ugyanúgy végezhetjük:

```
umount mtpt/
```

A most létrehozott `ntfs.img` nevű fájlt ugyanúgy mozgathatjuk és másolhatjuk, mint bármely más – bár kissé talán túlméretes – fájlt. Ha szükséges tömöríthetjük, tárolhatjuk biztonsági másolatként, vagy feltölthetjük egy távoli gépre. A másolatot a bemutatott módszerre akár át is szerkeszthetjük, és szükség esetén felülírhatjuk vele az eredeti fájlrendszert. A lementett képfájl visszaírása – feltéve, hogy abban a könyvtárban vagyunk, amelyben található – a következő paranccsal történhet:

```
ntfsclone  ntfs.img -O
➥/dev/hda1
```

Néha a kisebb a jobb, így aztán az `ntfsclone` parancsnak is vannak olyan kapcsolói, amelyekkel hatékonyan tömöríthetjük a képfájlt. Talán meglepő, de ezekkel a jelzőkkel a klónozás még gyorsabb is lesz, akár a helyi gépen végezzük, akár hálózatról. Íme egy példa, amikor a képfájlt majdnem pontosan ugyanúgy mentjük, mint az imént:

```
ntfsclone --save-image /dev/
➥hda1 -O \
 /usr/local/backup/ntfs.img
```

Persze azért semmi sem lehet tökéletes: ezt a képfájlt nem csatolhatjuk be a bemutatott módon, hacsak vissza nem írjuk egy merevlemezre, vagy egy másik, tömörítetlen fájlba. Az eredeti partícióra való visszaállítás a következőképpen fest:

```
ntfsclone --restore-image
➥--overwrite /dev/hda1 \
/usr/local/backup/ntfs.img
```

Figyeljük meg, hogy ebben a példában a nem túl beszédes `-O` kapcsolót a szkriptekben inkább szokásos `--overwrite` formára cseréltük. Persze a hatás pontosan ugyanaz lesz, csak a forma más. Általában is igaz, hogy a legtöbb jelző vagy kapcsoló értékét megadhatjuk hosszú de beszédes, vagy egybetűs formában is.
És akkor most jön a dolog legjobb része. Az `ntfsclone` program az adatokat alapértelmezésként a szabványos kimenetére küldi. Ez egyben azt is jelenti, hogy az így kinyert adatokat könnyedén átadhatjuk a legkülönbözőbb tömörítőprogramoknak, vagy átküldhetjük őket közvetlenül a hálózaton is. Alapvetően bármely olyan program fogadhatja ezt az adatfolyamot, amely képes a szabványos bemenetről olvasni. Nézzünk néhány efféle alkalmazási példát is.
Ha már a mentés során tömöríteni szeretnénk a keletkező képfájlt, használjuk a következő feldolgozási sort:

```
ntfsclone --save-image --output
➥- /dev/hda1 | gzip \
 -c >ntfs.img.gz
```

Az `ntfsclone` a képfájl adatait a szabványos kimenetére küldi, amit úgy állítottunk be, hogy a --output jelző után csupán egy mínuszjelet adtunk meg. A következő lépésen a `gzip`

azonnal tömöríti az átáramló adatfolyamot, amely végül az ennek a kimeneteként megadott `ntfs.img.gz` nevű fájlba kerül.
Ha azt szeretnénk, hogy a képfájl egy távoli gépen keletkezzen, a következő formát használhatjuk:

```
ntfsclone --save-image -o -
➥/dev/hda1 | ssh \
backups@storage.mydomain.org \
"dd of/home/backups/windows/
➥images/ntfs.img"
```

Itt a `--output` jelzőt felváltotta az egybetűs megfelelő (`-o`), a kimeneti csatorna azonban továbbra is a szabványos kimenetre irányul. Az adatfolyam ugyanakkor most az `ssh` bemenetére kerül, amely átküldi azt a hálózaton a megadott nevű gépre. Itt végül egy `dd` parancs fogja majd elhelyezni a megfelelő könyvtárban a képfájlt. Az egész folyamat, valamint az `ssh`-val való bejelentkezés is a `backups` nevű felhasználó nevében, az ő jogosultságaival zajlik.
Íme egy újabb példa:

```
wget ftp://storage.mydomain.org/
➥home/backups/
windows/images/ntfs.img.gz \
-O - |  gunzip | tee /usr/
➥local/backup/ntfs.img | \
ntfsclone --restore-image --
➥overwrite /dev/hda1 -
```

Ezt a részletet akár közvetlenül egy klónozószkriptből is vehettük volna, hiszen láthatólag se jelszót nem kér, se bármilyen más felhasználói bemenetre nincs szüksége. A tömörített képfájlt a `wget` segítségével tölti le, majd a `gunzip` paranccsal kibontja. Ezután egy `tee` parancs következik, amivel elágaztatjuk az adatfolyamot. Az egyik ág a Linux partícióra kerül, amely a klón képfájl egy biztonsági másolatát továbbra is hordozni fogja. A másik ág a `/dev/hda1` partícióra íródik, ami természetesen nem más, mint a visszaállítani kívánt ***NTFS*** partíció. A dolog működéséhez természetesen elengedhetetlen, hogy a `storage.mydomain.org` nevű kiszolgálón működjön egy anonim ***FTP*** szolgáltatás. A „jelszómentes” letöltés persze máshogy is megoldható. Használhatjuk például a ***wget*** és ***Apache*** megfelelő kombinációját, vagy készíthetünk titkosító kulcspárt is a két gépen, amelyek segítségével az ***SSH*** kapcsolatot automatikusan hitelesíteni tudjuk. Erre a problémára is igaz, hogy a megoldások számát szinte csak a fantázia korlátozza, hiszen a megfelelő eszközök rendelkezésre állnak.
Az ***ntfsprogs*** csomag egy másik igen hasznos tagja az `ntfsresize`. Ez pontosan azt teszi, amit a neve ígér: úgy lehet vele átméretezni ***NTFS*** fájlrendszereket, hogy közben az adataink is megmaradnak. A program működik valódi partíciókon, de használhatjuk az `ntfsclone` segítségével készített klónfájlokon is.
Érdemes megjegyezni, hogy az `ntfsresize` nem módosítja a partíciós táblát, csupán a partíción belül található ***NTFS*** fájlrendszert nyomja össze. A partíciós táblát az új helyzetnek megfelelően az `fdisk` vagy az `sfdisk` segítségével módosíthatjuk.
Ebben a cikkben nem tudom részletesen leírni, hogyan is kell egy lemezt particionálni. Ami azt illeti, annak a részletes és körültekintő leírása, miként kell átméretezni egy olyan partíciót, amit eredetileg teljes egészében egy ***NTFS*** fájlrendszer foglal el körülbelül olyan hosszú lenne, mint az egész cikk. Maga a művelet persze nem hosszadalmas, csak kissé veszélyes. Aki figyelmetlen, vagy egyszerűen csak nincs szerencséje, könnyen olyan helyzetbe kerülhet, hogy többé nem tudja bebootolni a gépét. Ezt figyelembe véve, na meg azt, hogy egy merevlemez manapság körülbelül annyiba kerül, mint egy tank benzin, a továbbiakban egyszerűen feltételezem, hogy a particionálás már kész. Tegyük fel ugyanakkor, hogy az ***NTFS*** partíció egyszerűen túl kicsi ahhoz, hogy befogadja a ráírni kívánt ***NTFS*** fájlrendszert. Ilyen helyzetbe akkor szaladhatunk bele, ha nem vettük figyelembe, hogy a legtöbb particionálásra használt segédprogram lefelé kerekíti a szektorszámot, vagy ha egy hibás merevlemezt cseréltünk ki egy elvileg ugyanakkorára (méretként ugyanaz a szám van ráírva), csak – az ismert okok miatt – nem jött be az elv, vagy más a geometria.
Az `ntfsclone` kiválóan működik a „túl nagy” partíciókon, de semmit nem tud kezdeni a kicsikkel. Ha akár csak egyetlen olyan bit van a képfájlban, ami nem írható ki a rendelkezésre álló szabad helyre, a program egyszerűen el se indul.
Ilyenkor az `ntfsresize` lehet a mentőangyalunk. Ahhoz, hogy kiderítsük, vajon mennyire lehet összenyomni a képfájl tartalmát, futtassuk a következő parancsot (abból a könyvtárból, ahol az `ntfs.img` található):

```
ntfsresize --info ntfs.img
```

A program egy ilyesféle jelentést fog készíteni:

```
ntfsresize v1.11.2
Device name        : ntfs.img
NTFS volume version: 3.1
Cluster size       : 4096
➥bytes
Current volume size:
➥90009203200 bytes (90010 MB)
Current device size:
➥90009203200 bytes (90010 MB)
Checking filesystem consistency
➥...
100.00 percent completed
Accounting clusters ...
Space in use       : 6508 MB
➥(7.2%)
Collecting resizing constraints
➥...
You might resize at 6507421696
➥bytes or 6508 MB
(freeing 83502 MB). Please make
➥a test run using both
the -n and -s options before
➥real resizing!
```

A fenti diagnózis szerint pánikra semmi ok, a fájlrendszer tartalmát akár 6508 MB-ra is összenyomhatjuk. Persze annak sincs értelme, hogy a lehető legkisebbre összenyomjunk egy fájlrendszert, olyannyira, hogy ha ezt a rendszerpartíció tartalmával tesszük meg, akkor a ***Windows*** valószínűleg el se tud indulni a kialakult helyhiány miatt. Okos ember hagy némi helyet a jövőbeni „növekedésre”. Maga a program ugyan azt javasolja, hogy ha nem vagyunk biztosak a dolgunkban, akkor futtassuk először teszt üzemmódban a `-n` és `-s` kapcsolók egyidejű megadásával, ha azonban tényleg biztosra akarunk menni, akkor legyen egy biztonsági másolatunk az eredeti képfájlról. Persze mindkét módszert is használhatjuk egyszerre.

Ha a fájlrendszert 10000 MB méretűre akarjuk összenyomni, a következő parancsot kell kiadnunk:

```
ntfsresize --size=10000M
➥ntfs.img
```

Működés közben az `ntfsresize` elég sok diagnosztikai információt ír ki, amely sok egyéb mellett a következőket is tartalmazza:

```
100.00 percent completed
Updating $BadClust file ...
Updating $Bitmap file ...
Updating Boot record ...
Syncing device ...
Successfully resized NTFS on
➥device 'ntfs.img'.
```

Ezzel elvileg elő is állítottunk egy olyan *NTFS* fájlrendszert, ami már ráfér a véletlenül alulméretezett partícióra.

Összefoglalás

Azt ugyan nem lehet állítani, hogy ha valaki egy biztonságos *Windows* munkaállomást szeretne magáénak tudni, akkor föltétlen szüksége van az itt bemutatott eszközökre, azt viszont nyugodtan, hogy ezek használatával jóval egyszerűbb lesz az élete.

Ha valakinek egyetlen számítógépe, és azon két operációs rendszere van, az *NTFS*-t kezelő linuxos segédprogramokkal teljes mentést készíthet az *NTFS* partícióról egy másik fájlrendszerre, vagy akár egy olyan kivehető merevlemezre, amelyen megfelelő nagyságú szabad hely áll rendelkezésre. Könnyen elképzelhető persze, hogy ez a lehetőség nem mindenkinek éri meg a fáradságot. Ugyanakkor ha már eleve adott a gépen a két különböző operációs rendszer, telepíthetjük a csomagból azokat az egyéb eszközöket is, amelyekkel fokozott biztonság érhető el, s ez már talán meggyőző érv lehet mindenki számára.

Ami a hálózati rendszergazdákat illeti, ha számos *Windowst* futtató gépet kell karbantartaniuk, az itt bemutatott eszközök igen nagyot lendíthetnek munkájuk hatékonyságán. A két operációs rendszert futtató gépeken létre lehet hozni megosztott partíciókat (a témáról bővebben lásd *Kevin Farham* cikkét a *Linux Journal 2006 júliusi számában*). Ha mellesleg a *GRUB* és néhány egyéb menüfájl is telepítve van egy ilyen megosztott partícióra, akkor írhatunk olyan szkripteket, amelyek egy olyan speciális futási szinten indítják újra a gépet, amelyen működve az automatikusan visszaállítja a megadott *Windows* lemezképet, frissíti azt, és így tovább.

A *Windows* és a *Linux* számos területen versenytársaknak tekinthetők. Ugyanakkor a *Linux* egyik legnagyobb erőssége a nyitottsága, valamint parancssori eszközeinek hajlékonysága. A *Linux-NTFS* eszközök egyrészt lehetővé teszik a kommunikációt az *NTFS* fájlrendszerekkel, másrészt mivel ez a kommunikáció alapvetően egyirányú, egyben természeténél fogva ideálisan biztonságos is.

Linux Journal 2007., 154. szám

Steven Mathes először valamikor 1995-ben telepített Linuxot a számítógépére. Ekkor a Windows partíciók tartalmáról még csak a tar paranccsal lehetett biztonsági másolatot készíteni. Ha valakinek kérdése van, Stevent a smathes@tiac.net címen érheti el.

A Citadel csoportmunka-kiszolgáló

Microsoft Exchange, had mutassam be Önnek az utódját...

Elérkezett az ideje, hogy fontolóra vegyük az átállást a drága, magas költséggel fenntartható ***Windows*** rendszerről egy ügyesebb, robusztusabb ***Linux*** megoldásra. A rendszerek különbözőségének felmérése megtörtént, az értekezletet megtartva, a prezentációk elkészültek, ideje elkezdeni. Bár egy linuxos háttérszerver telepítése igazi kihívás lehet számunkra, mindannyian tudjuk, hogy a felhasználókat ez a téma nem igazán hozza lázba. Számukra a megszakítás nélküli működés a fontos, hogy végezhessék tovább a munkájukat. A felhasználók egyik alkalmazásról a másikra történő költöztetése (például ***Microsoft Office***-ról ***Openoffice.org***-ra) szinte rutinfeladatnak számít, azonban az igazi mumus, ami nem hagy minket aludni éjjelente az e-mail és a csoportmunka kérdése. Hogyan nyújthatunk a ***Microsoft Outlookához*** hasonló szolgáltatásokat széles tömegek számára?
A válasz egy szóban: ***Citadel***.
A ***Citadel*** csoportmunka-kiszolgáló (***Citadel GroupWare Server*** ➲ www.citadel.org) a szabad és nyílt forrású szoftverek egyik méltatlanul mellőzött csodája. A projekt 1987-ben indult a már létező ***Citadel-CP/M*** alkalmazás ***UNIX***-os változataként, és kezdetben egyetlen fejlesztő irányította.
Húsz évvel később a modern idők ***Citadelje*** a kiforrott csoportmunka-kiszolgálók összes szolgáltatásával büszkélkedhet. A különlegességek egyike, hogy A ***Citadel*** gyakorlatilag a ***Microsoft Exchange*** összes gyakran használt szolgáltatását kínálja, közben kevésbé akadékos, és sokkal kevesebbe kerül. Számtalan mai cég, szervezet jut arra a felismerésre, hogy az informatikai költségvetésük legnagyobb része a ***Microsoft*** licencdíjaira és hardverkövetelményeinek kielégítésére megy el. A cégek persze fel tudnak készülni előre bizonyos költségek finanszírozására, a legtöbbjük mégis elborzadva tekint a ***Vistára***. Igaz, hogy a ***Vista*** hardverkövetelményei nem szemtelenül nagyok, a legtöbb cégnek mégis szüksége van a hardvereszközök frissítésére a ***Vista*** futtatásához, és előbb vagy utóbb kénytelenek lesznek bevállalni. A számítógépek állandó frissítése a cégek számára a fájdalmak soha ki nem merülő forrása, mivel nem csak kiszolgálók de az az azokon futó szoftverek frissítése szükséges. S bár ez önmagában jól körülhatárolható volna, emellett a tömérdek munkaállomás is figyelmet kíván.
Cégünk méretétől függően valószínűleg a ***Microsoft Excange*** kiszolgáló a legizmosabb darab a szerverszobánkban, és a kiváltására alkalmas gép keresése az esetek túlnyomó többségében kivágja az embernél biztosítékot.
A ***Citadel*** egy olyan csoportmunka megoldás, amely lehetővé teszi a cég számára, hogy elkerülje a szoftverfrissítéssel járó kellemetlenségeket, és a hardverigénye is lényegesen alacsonyabb, így pár évvel kitolható az újabb gépek beszerzése.

Hogyan juthatunk hozzá?

Helyes dolog a bevett gyakorlat, miszerint a telepítést és a tesztelést egy tesztgépen végezzünk, mielőtt az éles környezetbe tennénk. A levelezőszerver cseréje esetében sincs ez másként, azaz végezzük a ***Citadel*** tesztelését olyan messze az éles rendszertől, amennyire csak lehetséges. A ***Citadel*** telepítése és beállítása nagyságrendekkel könnyebb egy internetre kötött géppel, mivel így kihasználhatjuk az ***Easy Installation*** (egyszerű telepítés) folyamat nyújtotta előnyöket.
A cikk írásának pillanatában a ***Citadel*** legfrissebb változata 6.84-es. Ajánlott ellátogatni a ***Citadel*** honlapjára hogy hozzájussunk a kiszolgáló és a telepítési útmutató legfrissebb változatához.
A tesztkörnyezetünk egy ***VMWare Player 1.0.2***-ben futó ***Debian Sarge***, 2.6-os kernellel. Minden különösebb jelentőség nélkül a cikk írása során a Webről történő telepítési lehetőséget részesítettem előnyben, de a többi telepítési eljárás is működni fog, mivel a ***Citadel*** mindent felrak, amire szüksége van. Korábban már telepítettem, majd futtattam ***Citadelt Debian Sarge*** kiszolgálón minden gond nélkül, és mindkét telepítési mód tökéletesen működött.

Telepítés

Az ***Easy Install*** folyamatot fogom bemutatni. Nemcsak azért mert könnyű, de erőforrások szempontjából is meglehetősen igénytelen, ennélfogva senki nem tudna jobbat mutatni. Az egyetlen követelmény, hogy működésre bírjuk az ***Easy Installt***, az egy – lehetőleg gyors – internetkapcsolat.
Az ***Easy Install*** folyamathoz a telepítendő rendszeren szükségünk van a fordítókörnyezetre (***build environment***). Ezen felül kell még egy ***curl*** (vagy ***wget***). Ha ***SSL*** elérést is szeretnénk a szerverünkhöz szükség lesz még a ***libssl-dev*** csomagra is. ***Debian*** rendszeren a következő parancs kiadásával telepíthető illetve ellenőrizhető a fordítókörnyezet megléte:

```
apt-get install build-essential
➥curl libssl-dev
```

Nem sok előnye van a *Citadel* fekete doboz (zárt rendszer) szemlélet szerinti tervezésének. A szemlélet egyrészt úgy jelenik meg a gyakorlatban, hogy a *Citadel* a saját hitelesítő felhasználói-adatbázisát használja és nem a rendszer adatbázisát (leggyakrabban: */etc/passwd*). Ha azt szeretnénk, hogy a *Citadel* a rendszer hitelesítő adatbázisát használja, mielőtt hozzáfognánk a telepítéshez, állítsuk be az `IS_AUTOLOGIN` környezeti változót 'Igen' értékre az alábbiak szerint:

```
export IS_AUTOLOGIN=yes
```

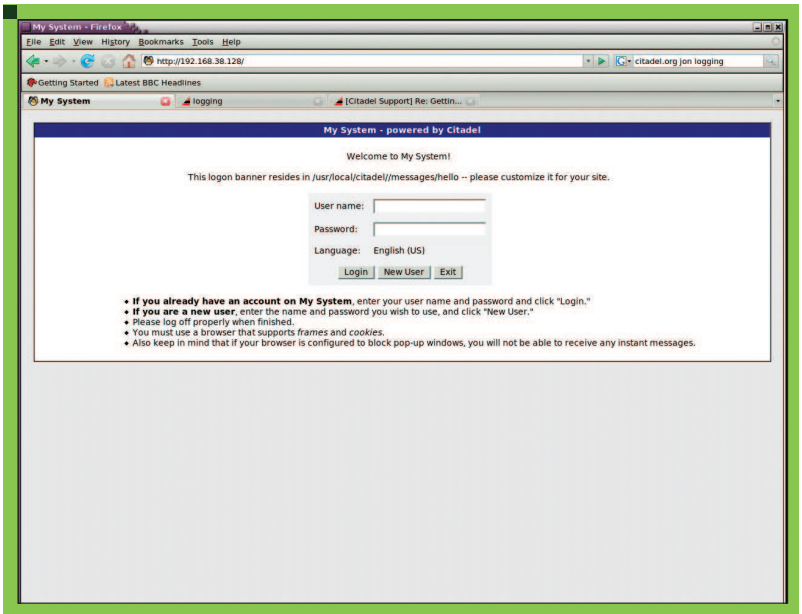
Most, hogy a környezet kész, itt az ideje, hogy berúgjuk az *Easy Installt* a következő paranccsal:

```
curl
➥http://easyinstall.citadel.org/
➥install | sh
```

vagy ha a *wget* szimpatikusabb, akkor:

```
wget -q -O -
➥http://easyinstall.citadel.org/
➥install | sh
```

A *Citadel* letölti a telepítőt, kicsomagolja, majd elindítja a telepítési folyamatot. A telepítési folyamat során a *Citadel* feltesz néhány kérdést, amihez ott az alapértelmezett válasz is. Így ha nem válaszolunk sem marad nyoma semmilyen misztikus beállítási naplóban.
A *Citadel* igen szerény, és bár rengeteg hasznos eszközzel gazdagítja a társaságot, nem feltételezi, hogy ezek közül bármire is szükségünk lesz. A *Citadel* megkérdezi, hogy szeretnénk-e a beépített *POP*, *SMTP* vagy *IMAP* kiszolgálót használni, vagy hagyja a rendszerünkön már meglévőeket úgy ahogy vannak. Továbbá ott van még a *WebCit* névre hallgató webes felület, amit a felhasználók ugyanúgy használhatnak a leveleik, naptárbejegyzéseik és névjegyzékük eléréséhez ha úton vannak, vagy valahol távol a helyi e-mail és személyi adataikat kezelő ügyfélprogramjuktól. Ha kiválasztjuk a *WebCit* telepítését, a *Citadel* nem akarja egyből a 80-as kaput lefoglalni. Így lehetővé válik a *WebCit* futtatása egy nem szabványos kapun, változatlanul hagyva ezzel a kiszolgálón már meglévő egyéb webszolgáltatásokat.



1. ábra A bejelentkezési képernyő

A kívánciak számára: A *Citadel* a */usr/local/citadel* mappába kerül, a *WebCit* – ha kértük a telepítését – pedig a */usr/local/webcit* mappába. A kapcsolódó rendszerkönyvtárak a */usr/local/ctdlsupport* mappában találhatók.

Eltávolítás

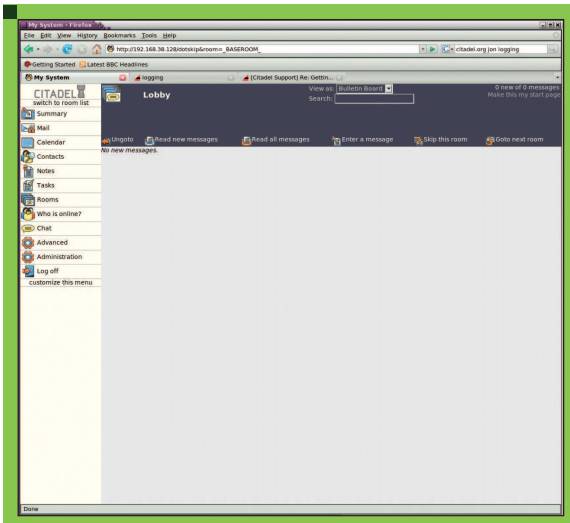
A telepített *Citadel* példány eltávolítása nagyon egyszerű:

1. Töröljük a három fentebb említett mappát (*/usr/local/webcit*, */usr/local/citadel*, */usr/local/ctdlsupport*).
2. Távolítsuk el a *Citadel* és *WebCit* bejegyzéseket az *inittab* fájlból (általában: */etc/inittab*).
3. Adjuk ki az `init q` parancsot az *init* újraindításához.

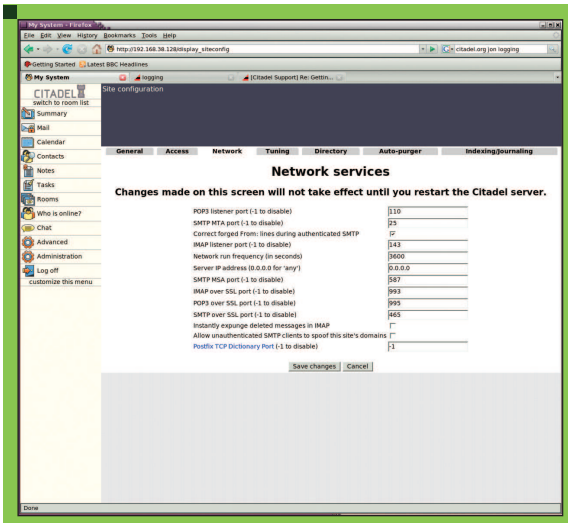
Kész.

Kezdeti beállítások

A beállításhoz és a használathoz is a *WebCit* webes felületét választottuk, ám ezalatt a szép felhasználói felület alatt egy szöveges alapú *BBS* szív dobog. Látszólag a *Citadel* minden beállítási és a legtöbb mindennapos használatú lehetősége elérhető szöveges módban a szebb napokat is látott *BBS* segítségével. Sajnos ez a fajta kapcsolattartás szinte teljesen ismeretlen a mai felhasználók számára, így a *WebCit* felületre összpontosítunk a munkánk elvégzéséhez.
Az a helyzet, hogy ettől még néha szükséges lehet konzolról vagy héjból is bejelentkezni, ezért meg kell változtatnunk a *Citadel* naplózási módját. Alapértelmezetten a *Citadel* a konzolra naplóz, és ezt át kell irányítani valahová máshová, hogy el tudjuk végezni a dolgunkat a héjban is. Számtalan különböző módszer létezik ennek eléréséhez, de mióta a *Linux* konfigurálható *syslog* démonnal rendelkezik, logikus választásnak tűnik a */etc/-syslog.conf* szerkesztése (*Debian* rendszeren), és a *local4* képesség átirányítása egy naplófájlba, vagy valahová máshová, ahol nincs útban.
Az első ember, aki bejelentkezik a *Citadel* webfelületén rendszergazda szintű felhasználóvá válik. A rendszergazda fiók elkészítéséhez írjuk be a böngészőnkbe az elérési utat és a kaput, amit a *WebCit* telepítése során megadtunk, írjuk be a felhasználónevet és a jelszót, és nyomjuk meg az Új felhasználó gombot (*1. ábra*). Onnét tudjuk, hogy rendszergazdává váltunk a rendszerben, hogy látjuk a Rendszergazda feliratú gombot a menü bal alsó sarkában a bejelentkezés után (*2. ábra*).
A rendszerszintű beállítások eléréséhez kattintsuk a *Rendszergazda* feliratú gombra, és egy jól szervezett,



2. ábra A nyitólap, ha rendszergazdaként jelentkeztünk be



3. ábra Hálózati szolgáltatások beállításai

mindenre alkalmas beállítómenübe jutunk. A fő kategóriák az oldal teteje mentén találhatók, rájuk kattintva az adott terület beállításai tűnnek fel a képernyőn. Már említettük, hogy a *Citadel* a webfelület alatt szöveges módú, és a beállítási lehetőségek egy része elég nyilvánvalóan arra vonatkozik.

Az összes beállítási lehetőség alapos ismerete túl mutat a cikk hasábjain, a legfontosabb lehetőségek azonban megtalálhatók *Hálózat és Könyvtár* (*Network and Directory*) menüpont alatt (amennyiben *LDAP*-ot használunk). A Hálózat fül alatt az egyes kapuk megváltoztatására és az azokon futó szolgáltatások engedélyezésére és tiltására van lehetőség. A *Könyvtár* menüpont alatt az *LDAP* beállításokat adhatjuk meg. Ha mi nem használunk *LDAP*-ot, ezeket a képernyőket hagyjuk változatlanul, mivel a *Hálózat* menüpont alapbeállításai is egész korrektek, illetve a telepítés során megadott értékeket tükrözik.

Megnézhetjük a *Hozzáférés (Access)* fület is, hogy megbizonyosodjunk arról, hogy a megfelelő értékek vannak beállítva az új felhasználók létrehozására vonatkozóan. Valószínű, hogy egy céges kiszolgáló esetében a rendszergazda létrehozza előre az összes felhasználói fiókot, ezután a felhasználók általi fiókkészítést lehetőségét ki lehet kapcsolni.

Általában, mielőtt a felhasználókat rászabadítanánk a *WebCit* felületre, valószínűleg be szeretnénk állítani az oldal kinézetét egy kicsit. Ahogy végigmegyünk a *WebCit* telepítési folyamatán, láthatjuk az alapértelmezett szöveges feliratokat az oldalon, amely tartalmazza azok elérési útját is. Ennek egyik legjobb példája az üdvözlőfelirat a fő bejelentkező képernyőn (*1. ábra*). Változatos tartalmú szövegfájlok találhatók a */usr/local/citadel/messages* mappában, amelyek az igények szerint testre szabhatók.

Az e-mail-ek beállítása

Kezdjük mindjárt az elején, és mielőtt a levelezési bejegyzéseket átirányítanánk a vadiúj *Citadel* kiszolgálóra, rögzítsük, hogy mely tartományokra érkező leveleket fogadja a rendszer. Ajánlom, hogy bízzuk ezt a dolgot a *Citadel* kínálta lehetőségre, és ne próbálkozzunk a beállítási fájlok hegesztésével. Ahhoz, hogy beállítsuk az általunk kezelni kívánt levelezési tartományokat, kattintsunk a *Haladó (Advanced)* menüpontra, majd ott a *Tartománynevek és Internet e-mail* (*Domain names and Internet e-mail*) beállítási hivatkozásra.

A megjelenő képernyőn írjuk be az első tartományt, amire fogadni akarjuk a leveleket a *Helyi álnevek* (*Local host aliases*) mezőbe. Kattintsunk a *Hozzáadás* (*Add*) gombra, és folytassuk a többi tartománynév beírását, ahogy épp az adott helyzet megkívánja.

A helyi álnevek mező az egyetlen, amelyet kötelező megfelelően kitölteni, de ezen felül számos egyéb haladóbb működés beállítására is lehetőségünk van ezen a képernyőn belül. Megadhatjuk a tartományokat, amit a *Globális Címlistában* (*Global Address List, GAL*) a címek végéhez fűzzön, kijelezhetjük az intelligens gépazonosítókat (smart host adresses), ha a kiszolgálónk nem közvetlenül küldené a leveleket, vagy egy spamszűrőhöz, illetve *valósidejű feketelistához* (*realtime blackhole list, RBL*) irányíthatjuk, hogy kitakarítsa a kéretlen leveleket, mielőtt a helyi fiókba kézbesítené.

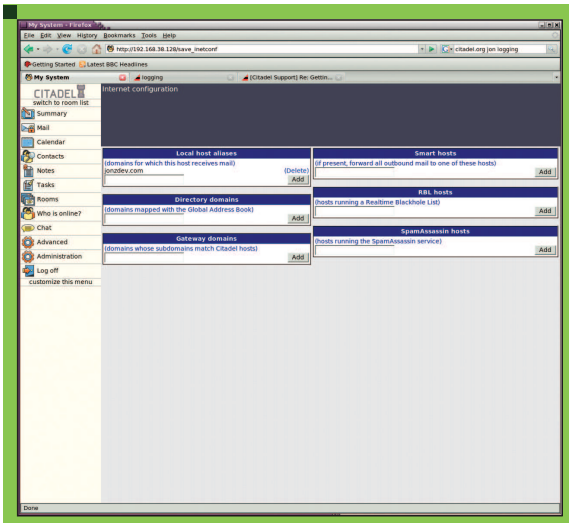
És ennyi. Működik a levelek küldése, fogadása a telepített *Citadel* kiszolgálónkon.

Ügyfelek beállítása

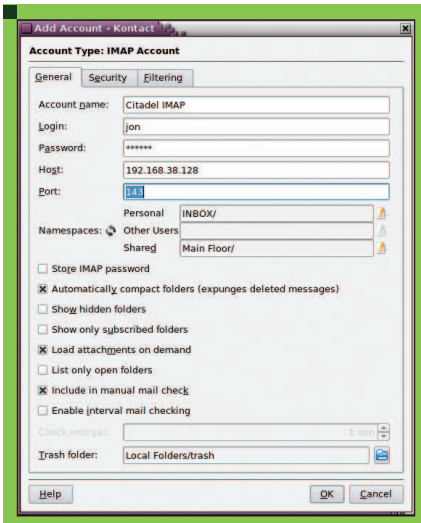
Szakmailag egyáltalán nem szükséges, hogy külön ügyfélprogramot állítsunk be. A *WebCit* az összes gyakran használt csoportmunka lehetőséget webfelületen keresztül kínálja, így a felhasználók azonnal hozzákezdhetnek az életük szervezéséhez a *WebCiten* keresztül. Ugyanakkor a helyi gépen futó ügyfélprogramoknak van néhány előnyös tulajdonsága, és a legtöbb felhasználó sem lesz elégedett a webes felülettel.

Szóval tennünk kell valamit.

A felhasználók szükségleteitől függően igen sokféle linuxos ügyfélprogramot használhatunk a *Microsoft Outlook* kiváltására. Számos próbálkozás után a *KDE Kontact* személyi adatkezelőt találta a legegyszerűbben használható programnak, amit



4. ábra Internetbeállítások



5. ábra KMail beállítása a Citadel kiszolgálóhoz

a *Citadel* kiszolgálóhoz csatlakoztathatunk, tehát ezt fogjuk használni. A *Kontact* a *KDE* projekt „mindent egyben" személyi adatkezelője. Bizonyos értelemben a *Kontact* egyszerűen csak a személyi adatok kiolvasásának egységes felületét nyújtja a *KMail*, *KOrganizer*, *KAddressbook* és néhány más jegyzetelő és hírolvasó összetevő számára.

KMail

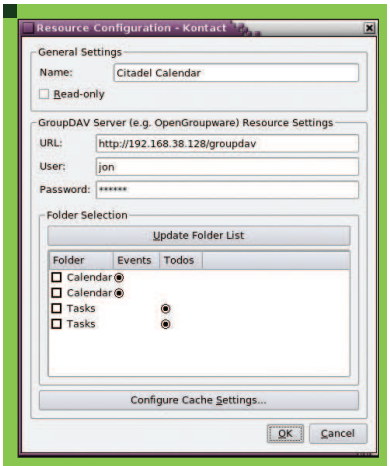
A *KMail* beállítása inkább nevezhető rutinfeladatnak. Ha valaha is találkoztunk már levelező ügyfél beállításával, akkor a *KMaillel* is boldogulni fogunk mindenféle tanács nélkül. Feltéve, hogy beállítottuk a *Citadel* kiszolgálónk *POP* vagy *IMAP* szolgáltatását, be fogjuk tudni állítani a *KMailt*, hogy kapcsolódjon hozzájuk. Egyszerűen csak adjuk meg a *Citadel* kiszolgáló *IP*-jét vagy *URL*-jét, a felhasználói fiók adatait, és ezzel készen is vagyunk (*5. ábra*).

KOrganizer

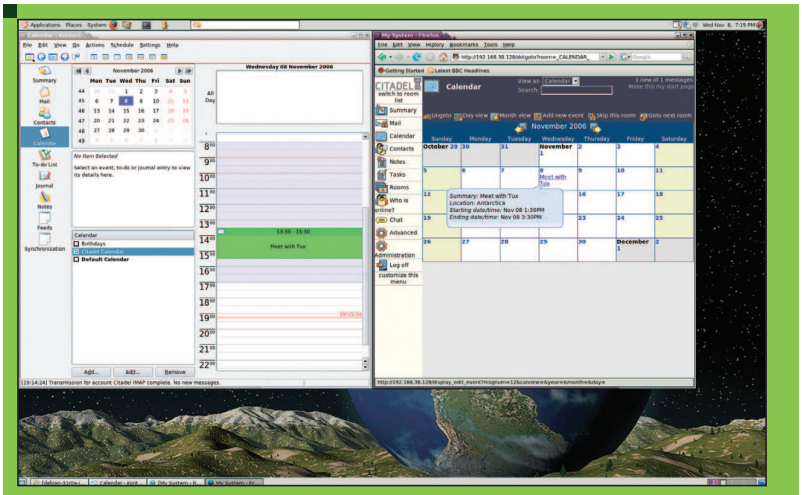
A *Kontact* naptárfunkcióinak beállítása egy kicsit nehezebb feladat. Úgy találtam, hogy a *GroupDAV* protokoll a legegyszerűbben beállítható és legprofibb, szóval ezt fogjuk használni. A néhány dolog egyike, amit tudnunk kell, hogy hogyan állítsuk össze a *GroupDAV URL*-ünket. Egész egyszerűen a *GroupDAV URL*-ünk a *Citadel* kiszolgáló *URL*-je (beleértve a nem szabványos *HTTP* kaput, ha úgy állítottuk be a *Citadelt*, hogy az ne a 80-as kapun figyeljen), a */groupdav* taggal a végén. Esetemben például a *GroupDAV URL*: http://192.168.38.128/groupdav.
A *KCalendar* csoportmunka lehetőségének engedélyezéséhez kattintsunk a *Naptár (Calendar)* ikonra a bal oldali sávon. A középső ablakrész alján megjelenik egy *Naptár* feliratú terület. Kattintsunk jobb gombbal a középső terület bármelyik részére, és válasszuk a *Hozzáadás (Add)* menüpontot.
A megjelenő ablakban válasszuk a *GroupDAV* kiszolgáló lehetőséget. Ha nem látnánk ezt az opciót, valószínűleg azért van, mert nincs telepítve a *kdepim-kresources* csomag. Telepítsük, majd indítsuk újra a *Kontact* programot, ezek után már jól kell működnie.
A *GroupDAV* kiszolgáló lehetőség kiválasztása után az Erőforrások beállítása (*Resource Configuration*) ablak fog megjelenni. A *Név (Name)* mezőben adjunk nevet a leendő naptárunknak (bármit, ami utalhat a naptár jellegére), az *URL* mezőbe pedig írjuk a már említett *GroupDAV URL*-ünket. A felhasználói név és a jelszó ugyanaz, amit beállítottunk, amikor legelőször beléptünk a *Citadel* kiszolgálóra. Kattintsunk a *Mappalista frissítése* (*Update Folder List*) gombra, és az alsó *Mappaterület* (*Folder Section*) meg fog telni a *Naptárak és Teendők* (*Calendar and Tasks*) választógombokkal (*6. ábra*). Elvileg a *Naptárak és Teendők* elemei előtti jelölőnégyzetekre kattintva engedélyezhetjük azokat, de még kicsit bugos a rendszer. Sokszor két példányt mutat ugyanabból., ahogy az a *6. ábrán* is látható. Úgy tűnik, hogy a *Naptárak és Teendők* engedélyezésének egyetlen biztosan működő módja, ha jobb gombbal kattintunk az egyes elemekre, és a felugró menüből az *Engedélyezés* menüpontot választjuk.

Contacts

A *Kontact Contacts* (mondjuk el ötször gyorsan egymás után) programjának beállítása pontosan ugyanúgy zajlik, mint a *KCalendar* esetében. Kattintsunk a *Kapcsolatok* (*Contats*) ikonra a bal oldali sávon. A középső ablakrész alján megjelenik egy *Címjegyzékek* (*Address Book*) feliratú terület.
A jobb gombos trükk itt nem működik, helyette kattintsunk Hozzáadás (Add) gombra. Válasszuk itt is a *GroupDAV* kiszolgáló lehetőséget, és töltsük ki ugyanazokkal az adatokkal, mint a *KCalendar* beállítása esetében. Kattintsunk a *Mappalista frissítése* (*Refresh Folder List*) gombra, majd alkalmazzuk a jobb gombos engedélyezéses trükköt, és már túl is vagyunk az egészen (*8. ábra*).
Hasonlóan, mint a *KCalendar* esetében, ha már beállítottuk a *GroupDAV* csatlakozást, a *KDE*-ből és a *WebCit* felületről egyaránt kezelhetjük az adatainkat (*9. ábra*).
A *Feladatok* (*Tasks*) és *Journal* területekkel a *KCalendar* beállítása után már csak annyi a dolgunk, hogy elvégezzük velük a munkánkat. Egyikük sem igényel külön beállítást.

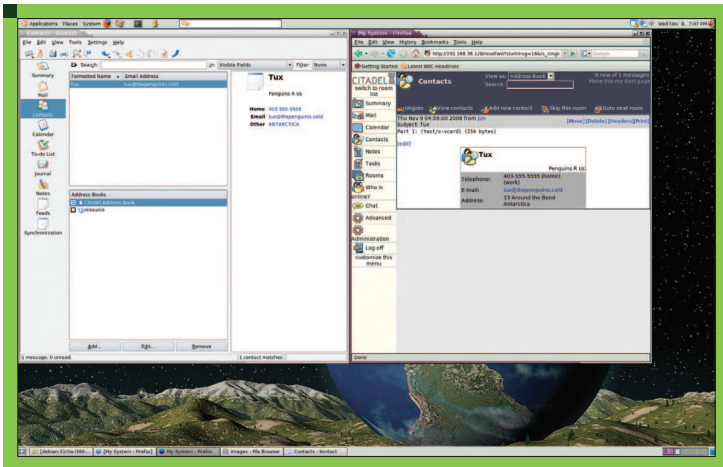


6. ábra A KOrganizer (Kontact) naptár beállítása Citadel-hez



7. ábra Kontact akcióban

8. ábra A GroupDAV beállítások a Contacts alkalmazásban



9. ábra Citadel és a Kontact egyszerre férnek hozzá ugyanahhoz az adathoz

Egy csomó más ügyfélprogram támogatja eltérő fejlettségi fokon a ***GroupDAV*** protokollt. Ezek bármelyike használható a ***Kontact*** helyett, jóllehet, valószínűleg kevesebb funkcióval. Az összes szóba jöhető ügyfél és azok ***GroupDAV*** támogatásának fejlettségi fokáról a ***GroupDAV*** weboldalán tájékozódhatunk (www.groupdav.org/implementations.html).

A ***GroupDAV*** nem az egyetlen technológia, amit a ***Citadel*** támogat. A ***WebDav*** és a ***Webcal*** protokollokat szintén használhatjuk, ha naptárak és határidőnaplók megosztásához ügyfélprogram gyanánt a ***Mozilla Sundbird***-öt, vagy az ***Evolutiont*** szeretnénk igénybe venni. Fejlesztés alatt áll még egy ***Microsoft Outlook csatlakozó*** (***connector***) is, de jelen pillanatban az ***Outlook*** csak a ***POP/IMAP*** leveleket és az ***IMAP*** mappákat tudja elérni. Ahogy az idő halad előre, egyre több ***GroupDAV*** és/vagy ***WebDAV*** protokollt támogató ügyfélprogram kerül a nyilvánosság középpontjába.

A ***Citadel GYIK*** tartalmaz egy folyamatosan frissülő listát az ügyfélprogramokról és azok beállításának módjáról.

Találhatunk még néhány a ***Microsoft Exchange***-t is megszorongató, csoportmunkát támogató projektet az út szélén, de a ***Citadel*** egész egyszerűen a legkönnyebben telepíthető és üzemeltethető rendszer. A ***Citadel*** telepítésének legnehezebb része kivárni, amíg letöltődik mindegyik összetevő. A ***Citadelt*** folyamatosan fejlesztik, és mire ez a cikk a nyomdába kerül, lehet, hogy új változatot adnak ki belőle. A vezető fejlesztőt, ***Art Cancrot*** elérhetjük az ***UNCENSORED! BBS*** fórum ***Citadel*** támogatás témakörén keresztül (***uncensored.citadel.org***), s vele együtt még más ***Citadel*** fejlesztőket és tapasztalt felhasználókat.

Linux Journal 2007., 154. szám

Jon Watson (www.johnwatson.ca) kanadai GNU/Linux rajongó aki rendszeresen publikál cikkeket a Linux közösség számára. Amikor nem ír, akkor a szabad és nyílt forráskódú programokról értekezik internetes naplójában, podcast-jaiban (előre elkészített hangfelvétel, amit bárki letölthet). Jon sűrűn fellelhető az irodájában Linux+ tanúsítványát fényesítve, amely rajta kívül senkit sem hoz lázba.

A Linux DVR már nem mese – Itt a MythTV!

James Turner áttekintést nyújt a MythTV-ről, a TiVo-t helyettesítő Linux alkalmazásról

A személyi digitális videó rögzítők (*DVR*-ek) megjelenése emberek milliói számára változtatta meg a tévénézés élményét. A *VCR* talán függetlenítette az embereket a csatornák által diktált programtól és időbeosztástól, de a *DVR* sokkal több szabadságot és irányítási lehetőséget nyújt számunkra.

A legtöbb fogyasztó ***set-top DVR***-t használ, akár külön megvásárolva, akár egy műholdas vagy kábeles csomag részeként. Természetüknél fogva ezek a készülékek nem tudják teljesen kihasználni a *DVR* előnyeit. Először is, a *DVR* gyártók üzletpolitikai okok miatt vonakodnak olyan technológiákat kifejleszteni, ami a nézők számára lehetővé tenné a reklámok automatikus átugrását. Emellett a kereskedelmi rendszerek havi előfizetési díjat kérnek a műsorajánlókért, ami egy év alatt már könnyen túllépheti a felvétel eredeti költségét. Végül a gyártók igyekeznek elvenni a tulajdonosok kedvét az olyan egyszerű változtatásoktól, mint például a lemezterület kibővítése vagy több *DVR* összekötése, ami lehetővé teszi, hogy a házban levő bármely tévén megnézhessük a felvett műsort.

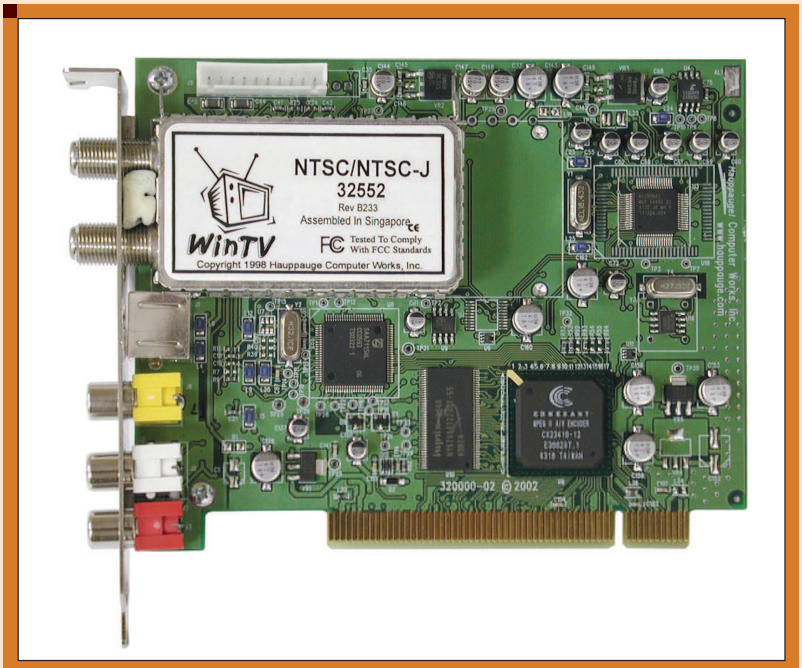
Ez elvezet a kettes számú megoldáshoz – mi magunk hozzuk létre a *DVR*-t. Számos olyan csomag elérhető ***Windows*** alatt – bár egyik sem ingyenes –, ami megfelelő *DVR* képességeket biztosít. Amennyiben ***Linux*** alatt futó minőségi *DVR*-t szeretnénk, a megoldás a *MythTV*. Ez a cikk végigvezet azokon a lépéseken, amire a *MythTV* telepítéséhez szükségünk lesz egy működő ***Linux*** rendszeren.

A *MythTV* felépítése egész egyszerű. Egy démon folyamat, a `mythbackend` felügyeli a tuner kártyákkal való párbeszédet, meghatározza a rögzítendő műsorokat, és ellátja a mindennapos *DVR*-feladatokat. Elméletileg ez minden, amit futtatnunk kell a szerverünkön. Például, ha van egy ***Hauppauge MediaMVP set-top*** átalakítónk, futtathatunk rajta egy speciális betöltőt, ami közvetlenül a szerveren futó ***MythTV back enddel*** fog kommunikálni, és lehetővé teszi, hogy a házon belül bármely tévén megnézzük a felvett műsort.

A legtöbb felhasználó azonban a `mythfrontend` programot is szeretné futtatni, ami egy ***GUI***-n keresztül az összes felhasználói szintű *PVR* funkciót biztosítja. Amellett, hogy a néző kiválaszthatja, mit vegyen fel, és milyen rögzített anyagokat illetve élő adásokat nézzen, a `mythfrontend` megjeleníthet időjárási adatokat, aktuális híreket, böngészhet weboldalakat, és még játékra is alkalmas megfelelő bővítmények (***plugin***) használatával. Egyidejűleg több kezelőfelületet is használhatunk (különböző gépeken futtatva), amelyek mind ugyanazzal a háttérszerverrel vannak kapcsolatban.

Először is beszéljünk egy kicsit a hardverről. Mint általában, a hardverszükségletünket itt is az határozza meg, hogy mire akarjuk használni. Minél több feladatot szeretnénk a *MythTV* szerverrel egyidejűleg kezelni, annál nagyobb számítási teljesítményre lesz szükség. Ha két műsor egyidejű felvétele alatt egy harmadikat nézünk, és eközben egy negyediket készítünk elő *DVD* írásra, ez már komoly erőforrásokat vesz igénybe, így nem árt rászánnunk a pénzt egy valamirevaló processzorra. Szerencsére nincs szükségünk folyékony nitrogénnel hűtött, háromszorosan tuningolt erőműre a feladat elvégzéséhez, egy 2.8 GHz-es chipnek elégnek kell lennie.

Emellett megfelelő lemezterületre is szükségünk lesz, ha kedvenc sorozatunk összes epizódját tárolni szeretnénk. Mivel a legkevésbé sem szeretnénk egy zajos lemez zúgását hallgatni a kedvenc műsorunk közben, használjunk *SATA*-t. Egy pár 250 GB méretű meghajtót már 250 dollárért is megkaphatunk, és ez már elég tárhelyet jelent ahhoz, hogy még a legkeményebb videófüggőket is kielégítse. Hamarosan a fájlrendszerekről is szót ejtünk.



1. ábra A Hauppauge WinTV PVR-250 a legkedveltebb MythTV kártya analóg TV-hez

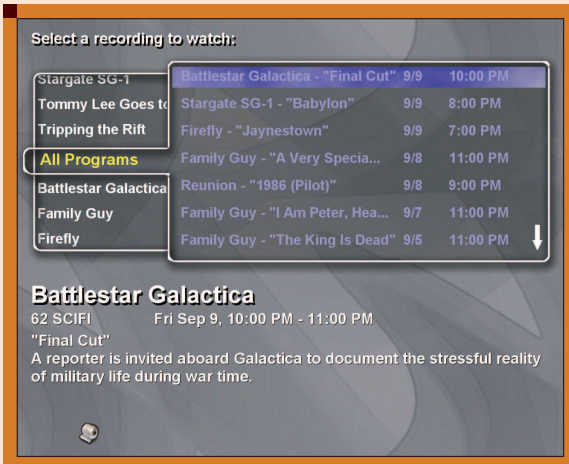
Meglepő módon a videóillesztő nem tartozik a kritikus összetevők közé. Ez nem jelenti azt, hogy elő kéne ásnunk az 1995 körül használt ***Hercules*** kártyánkat, de egy viszonylag új (pl. tavalyi) ***AGP*** kártyának megfelelőnek kell lennie, bár a fejlett ***OpenGL*** támogatás sokat segít. A tunerkártya már komolyabb kihívás. A megfelelő kártyák kiválasztása nagyban megkönnyítheti a ***MythTV*** rendszer telepítését és használatát. Egyértelműen figyelembe kell vennünk bizonyos tényezőket, például, hogy akarunk-e ***HDTV*** adást felvenni; ebben az esetben egy ***HD-re*** nem képes kártya eleve kiesik. A cikkben a legtöbb ***MythTV*** rendszerben használt kártyákkal, a ***Hauppauge WinTV-PVR-250***-el és a ***WinTV-PVR-350***-el foglalkozunk. Ami a ***Hauppauge*** kártyákat olyan vonzóvá teszi, az, hogy integrált ***MPEG*** kódolóval rendelkeznek, ami drasztikusan csökkenti a ***CPU*** terhelését. A 250-es és a 350-es változat közötti különbség az, hogy a 350 egy hardveres ***MPEG*** dekódert is magába foglal, emellett a videókimenetnek köszönhetően ***TV*** készülékhez csatlakoztatható. Ez azonban körülbelül 50 dollárral többe fog kerülni, mint a 250-es. Mivel elég valószínű, hogy nem a nappali közepén fogjuk elhelyezni a szerverünket, csak hogy csatlakoztathassuk a tévéhez, javaslom a 250-est, ami kiskereskedelmi áron körülbelül 130 dollárból már kijön, és vegyünk egy ***MediaMVP***-t (kb. 80 dollár) a tévécsatlakozáshoz.
A ***Hauppauge*** kínálatában megtalálható a ***WinTV-PVR-500 MCE*** két beépített tunerrel és egy videókimenettel, de nem tartozik hozzá távirányító, ami hasznos lehet, ha távolabbról szeretnénk vezérelni a ***MythTV***-t.
A ***WinTV-PVR-250*** viszonylag gazdaságos megoldást nyújt a műsorok rögzítésére, de tartsuk észben, hogy ***HDTV*** adás felvételére a ***Hauppauge*** kártyák nem alkalmasak.
Egy másik dolog, amin el kell gondolkodnunk, hogy a csatornaváltáshoz szükségünk van-e műholdvevő vagy kábeles egység vezérlésére. Ehhez valószínűleg egy ***IR Blastert*** kell alkalmaznunk: ez az eszköz a soros portra csatlakozik, és a megfelelő parancsokat küldi a ***set-top*** átalakítónak. Arra is figyeljünk, hogy egy ***set-top*** átalakítót csak egy csatornára hangolhatunk, így ha egyszerre két műsort akarunk felvenni, az lesz a vége, hogy két ***set-top box*** árát kell kifizetnünk. Ez az egy nagy előnye a kábeles és műholdas szolgáltatók által kínált ***DVR*** megoldásoknak; a ***set-top*** átalakítóba beépítenek mindent, így ezzel nincs probléma.

A szokatlan hardverkövetelmények után lássuk magát a rendszert. Annak ellenére, hogy a telepítését nehéznek mondják, úgy találtam, hogy a ***Gentoo*** kínálja a legkönnyebb megoldást a ***MythTV*** installálására. Használjuk bármely szabvány útmutatót az alap ***Gentoo*** rendszer felállítására. A legfontosabb dolog, amire figyelnünk kell, hogy megfelelő módon hozzuk létre a fájlrendszereket. Feltéve, hogy beszereztünk két 250 GB méretű ***SATA*** meghajtót, használjuk a ***Logical Volume Managert (LVM)***, hogy a tárhely legnagyobb részén egyetlen nagy partíciót hozzunk létre. Javaslom, hogy normál módon futtassuk a telepítést, de hagyjuk a 0. meghajtó nagy részét és az 1. meghajtó teljes egészét érintetlenül. Így felhasználunk körülbelül 10 GB-ot a 0. meghajtón a root, usr, swap és var céljaira, és fennmarad 240 GB. Amint készen van, és fut a rendszer, hozzunk létre egy 490 GB-os ***LVM*** partíciót a fennmaradó helyen.
Általánosan elfogadott jó tanács, hogy használjunk ***JSF (IBM Journaling File System)*** fájlrendszert azon a partíción, ahol a felvett műsorokat tároljuk. Ez nyújtja a legjobb teljesítményt nagy fájlok törlésekor – amire a ***MythTV*** esetében gyakran sor kerül. Ez azt jelenti, hogy biztosítanunk kell a ***JSF*** belefordítását a kernelbe (nem modulként használjuk). Ugyanez érvényes az ***LVM*** támogatásra is.
A ***Gentoo Wiki*** oldal (lásd az online források között) kiváló áttekintést nyújt a kernel helyes beállításáról a beépített ***Hauppauge*** távvezérlő kezeléséhez és a szükséges csomagok telepítéséről. Amint minden kész és működik, készen állunk magának a ***MythTV***-nek a konfigurálására. Szerencsére a telepítés egyszerűen abból áll, hogy elindítjuk a `mythsetup` programot, és egy sor varázsló segítségével megadunk néhány beállítást, mint például a kábeltévével, illetve a műholdas rendszerrel kapcsolatos információkat. A telepítés nem telik öt-tíz percnél hosszabb időbe.
A ***MythTV*** egyik érdekessége, hogy mindent egy ***MySQL*** adatbázisban tárol (kivéve magát a videót, természetesen). Ez megkönnyíti az információ importálását és kinyerését, valamint trükköket is lehetővé tesz. Például egy csatorna frekvenciájának finom-

2. ábra Programajánló

3. ábra MythWeb

hangolásához különböző értékeket írhatunk a megfelelő adatbázistáblába, majd a csatornákat távvezérlővel váltogatva megnézhetjük, hogy javult-e a helyzet. Természetesen az lenne igazán nagyszerű, ha a grafikus felületen trükközhetnénk a finomhangolással; ezt talán valaki majd implementálja egy jövőbeli kiadáshoz. Ez azt is jelenti, hogy lefuttathatunk egy egyszerű lekérdezést, és megnézhetjük, milyen műsorokat láttunk már korábban, vagy akár írhatunk olyan szoftvert, ami felhasználja a 2 hetes programajánló adatait – ezt a *MythTV* automatikusan letölti nekünk.



4. ábra Felvett műsorok

Ha mindez működik, képesnek kell lennünk műsorok felvételére, és lejátszására a monitoron illetve a hangfalakon (amit a hangkártyába vagy az alaplap hangszóró kimenetére csatlakoztattunk). A *Hauppauge MediaMVP* használatához a szerveren engedélyeznünk kell az *NFS*-t és exportálni a fájlrendszert a videótartalommal együtt. Futtatnunk kell egy *DHCP* szervert és *tftp* szervert is. Ehhez kiváló áttekintést találunk a *SourceForge* oldalon (lásd a Forrásokat).

A másik lehetőség a *MythWeb* telepítése, ami egy *Apache*-alapú webes kezelőfelületet biztosít, ahol megnézhetjük a programfüzetet, az ütemezett felvételeket és a már felvett műsorokat. A *Gentoo*-ban ez nagyon egyszerű, mindössze annyit kell begépelnünk, hogy `emerge mythweb`. A *MythTV* egyik kiemelkedő vonása a reklámok automatikus átugrásának képessége. Ezt egy jelölőnégyzettel engedélyezhetjük a telepítő varázslóban. Ha bekapcsoltuk, a program végeztével a felvett anyagokat sorban átvizsgálja, és bejelöli a reklámokat. Ez azt jelenti, hogy nem ugorhatjuk át a reklámokat egy műsor felvétele alatt, vagy közvetlenül utána, de általában a műsor végétől számított 30 percen belül elkészülnek a jelzések. Lejátszás közben ezután már bármikor átléphetünk egy reklámblokkot az előreugrás gombbal. A reklámok bejelölése nem tökéletes, de nagyon közel áll ahhoz. A *MythTV* arra is alkalmas, hogy a felvett anyagot automatikusan átkódolja a későbbi *DVD* íráshoz. Azt is beállíthatjuk, hogy a felvétel elkészülte után tetszőleges *Linux* program fusson le a videófájlra. Vajon a *MythTV* az igazi megoldás? Ha egyszerűen az a célunk, hogy a *PVR*-rel összekapcsolt tévé műsorát felvegyük, valószínűleg nem. A kábeles illetve műholdas szolgáltatóktól beszerezhető *DVR*-ek olcsóbbak, jobban illeszkednek a rendszerbe, és a telepítésük sem tart egy napig (vagy tovább). Ha mégis fontos, hogy kézben tartsuk a teljes irányítást, a műsort az egész házban, egyetlen forrásból elérhetővé tegyük, és nem akarunk évente 150 dollárt kiadni műsorajánlókért, rugalmasság, konfigurálhatóság és alakíthatóság tekintetében a *MythTV* jelenti a legjobb megoldást.

Linux Journal 2006., 140. szám

James Turner
a Linux Journal Product Review rovatának szerkesztője. Két könyvet írt a nyílt forráskódú Java fejlesztésről és az Axis Technology, LLC vezető szoftvermérnöke

FORRÁSOK

- www.linuxjournal.com/article/8584

Doom 2 modern köntösben

Ismét csak az id Software... Ezekbe a srácokba a kikapcsolódást kereső Linux felhasználó lépten-nyomon belebotlik. Még akkor is, ha két éve nem publikáltak semmit. Sőt, ekkor csak igazán! A külsős csapatok tucatjai ilyenkor teljes erőbedobással dolgoznak, korszakalkotó játékprogramok motorjára építve saját munkáikat.

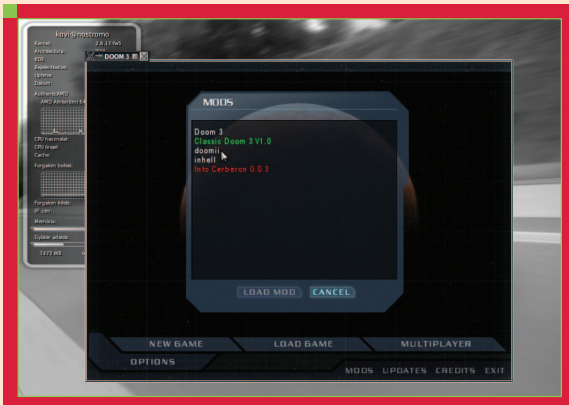
Aztán ezek a modifikációk több-kevesebb sikerrel szintén bevonulnak a köztudatba. Van idejük bőven, hiszen ***John Carmack*** gárdájára nem jellemző a kapkodás: általában éveket kell várni egy újabb ultramodern, ***id***-s lövöldözős játékra. Persze az életképes ***MOD*** projektek nem csak arra hivatottak, hogy kitöltsék ezt az űrt, hanem hogy elővarázsoljanak valami olyasmit, amivel tömegeket lehet a képernyő elé csábítani. Nem könnyű feladat, de van recept bőven: némely fejlesztő az érdekes újdonságokra hajt, némelyeknek szinte minden részlet lényegtelen (csak a végeredmény kellőképp divatos legyen).
De akadnak olyanok is, akik a ***remake*** próbákat értelmezik újra, és egy régi szoftver világának modellezését bízzák a naprakész alapkódra. Utóbbi csapatoknak hála, mára modern környezetben tapasztalhatjuk meg a ***Doom*** első részének frenetikus hangulatát, de akár a szakállas ***Descent*** sorozat ***Pyro*** űrhajóját is irányíthatjuk a ***Doom3*** szívére alapozva. Mivel azonban a második ***Doom*** is a halhatatlanok közé tartozik, így nem kerülhette el sorsát: a friss játékmotor manapság már az ***1994***-ben debütált elődjének újjászületéséhez is asszisztál...

Kiváló MOD

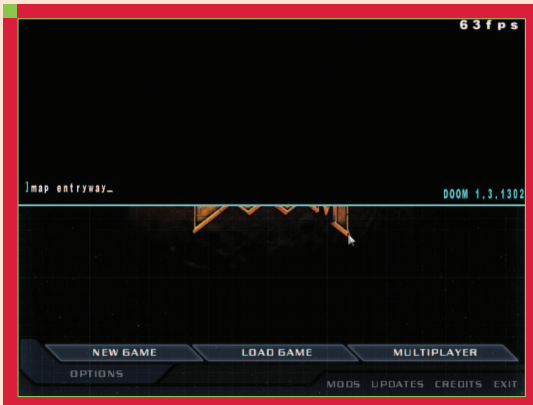
A ***Linuxvilág magazin 2006/02*** számában már ejtettem néhány szót a ***Classic Doom*** modifikációról. Akkor és ott röviden felvázoltam, milyen kiváló szoftver: nagyszerűségét (részben) a tizennégy éve született ős-epizód hangulati rehabilitálásának köszönheti. Ilyen megközelítésből kiindulva, a ***Doom 2*** fiatalító munkája is értékes – még akkor is, ha (népszerűségét tekintve) a valaha létezett leggyengébb ***id Software*** terméket hivatott modellezni. Magáról a játék menetéről nincs értelme írnom, hiszen egy igazi klasszikust formáz, kiforrott megoldásokkal: ami él és mozog leölendő, ezen felül a nyakatekert terepen kulcskártyákat keresve kell élve eljutni a kijáratig. A ***MOD*** teljes egészében használható ***Linux*** alatt is, kivitelezését és minőségét pedig nem érheti rossz szó. Amondó vagyok, becsüljük meg a befektetett energiát, és fejezzük ki sajátosan a hálánkat: „feszítsük rá" a ***Doom 2*** világát a modern alapkódra!

Íme, a hogyan...

Mivel modifikációról van szó, a használatba vételhez rendelkeznünk kell az anyaprogrammal. Ez nem jelent mást, mint a ***Doom3*** telepített, naprakész linuxos verzióját – valamint a felhasználóhoz (és az optikai médiákhoz) kötődő, érvényes kulcs-állományt (*/home/$/.doom3/base/doomkey*).
Ha ezek a feltételek adottak, el kell



1. ábra A linuxos Doom 3 „MODS" menüje



2. ábra Most ez a legfőbb konzolparancs

3. ábra Ugye, ismerős a felállás?

4. ábra Mint ahogy ez a helyzet sem idegen

5. ábra Úgy rémlik, 94-ben könnyebb dolgom volt...

6. ábra A tűzkoponyák a régiek

látogatnunk a fejlesztő csapat honlapjára. A ➲ ***http://s6.invisionfree.com/Disposable_Heroes*** címről indulva töltsük le a mindössze 10 ***Mbyte*** méretű ****.zip*** archívot. Tartalmát csomagoljuk a személyes ***/home/$/.doom3*** útra úgy, hogy a kibontott állományok egy erre a célra készült (például ***/doom2***) almappába kerüljenek, majd az adatokat hordozó ****.pk4*** fájl nevét alakítsuk csupa kisbetűsre.

Ilyen előzmények után a játék már készen is áll a használatra, a ***Doom 3*** „***MODS***” menüjéből tölthető be. Indítás után azonban egy fontos hiányossága rögtön szemet szúr: ***Linux*** alatt sajnos nem látszik a projektre szabott főmenü. Megoldásként hívjuk elő a vezérlőkonzolt, és gépeljük rá a `map entryway` parancsot – így léphetünk be az ismerős világba. A program hardverigénye egyezik a fő szoftverével, tehát csak bivalyerős központi egységgel, naprakész grafikus kártyával és rengeteg memóriával érdemes kipróbálni. Ellenkező esetben egy szaggatottan mozgó, nehézkes és csúnya játék benyomását fogja kelteni. Tapasztalatom szerint egymagos ***Athlon64 3500+ x86/64*** processzorral, 2 ***Gbyte DDR2*** központi tárral, ***nVidia 7300GT*** grafikus gyorsítóval már nagy felbontásban is szépen fut (még szerencse, hogy éppen most cseréltem le a masinámat). Valószínűleg sokaknak szemet szúr a letöltendő állomány szerény mérete. Ennek több oka is van. Először is, ebből a ***MOD***-ból hiányzik mindenféle zenei kíséret. Másodszor, szinte az összes felhasznált modell, fegyver és pályaelem az eredeti ***Doom3*** adataiból kerül meghívásra.

Ami elvárható

Az elvárható dolgokat úgy tudom legtömörebben érzékeltetni, ha a modifikációt pár szóval összemérem a többszörös méretű, 400 ***Mbyte*** helyigényű ***Classic Doom***-mal. A kisebb kód szerényebb megoldásokat foglal magában, mint a ***Flaming Sheep*** projektje: nincs olyan aprólékosan kidolgozott környezete, és egy idő múlva a zenei aláfestésének hiánya is elég feltűnő. Az sem mellékes, hogy maga a meg(re)formálandó ***Doom 2*** mindig is szerényebb játékélményt kínált, mint az egy évvel korábbi őse. Ennek ellenére ez a ***MOD*** minden ***id Software*** rajongónak kötelező darab. Ha valakinek dereng az ***1994***-es esztendőből valamilyen ***Doom***-os élmény, akkor roppant jól fog szórakozni. Zárásképpen még egy információ: ha az említett honlapon nem működne a letöltés linkje, akkor érdemes a ➲ ***http://www.doomwadstation.com*** oldalt meglátogatni (esetleg a hazai ➲ ***http://doom3.hardwired.hu*** portált), mivel a szükséges archív ott is elérhető.

Kovács Zsolt

Hamisítatlan nosztalgia

Igen gyakori vendégek felénk a modern játékok. Természetesen „nem csak a húsz éveseké a világ”, így ha valaki kevesellné a divatnak megfelelő nosztalgia-érzéseket, akkor igazat adnék neki. Ezennel pótolom a hiányosságot, a friss rovat retro jelleggel készült: Mario és Pong klónokról olvashattok.

Mario...

Istenem, mennyire megvénült ez az apró, köpcös vízvezeték-szerelő! Már húsz évvel ezelőtt is hallatott magáról, miközben akkor sem volt fiatal legény. Bizony: a bajsza már a nyolcvanas évek derekán is legalább olyan széles volt, mint az egész arca – márpedig egy ilyen fizimiska kialakulásához nem igazán elég egy-két esztendő. Valószínűleg sokunkat kellemes emlékek fűznek a figurához. Elég csak a korai ***Nintendo*** masinák közkedvelt ügyességi játékára gondolnom, de akár a mi „Tibi Bácsink” néhai ***Elektor Kalandor*** műsorát is említhetem... Igen, jómagam is ott ültem a fekete-fehér roncsunk előtt, és mereven bámultam a szerencsés játékosok bakijait. Micsoda technikát láthattam akkor! Igazi csoda volt, ahogyan a „kiválasztottak” ***Magyarország*** másik szegletéből irányították az ugráló figurát, egy vezetékes telefon gombjainak segítségével.

...és a követői

Mivel saját korának vezető ügyességi játékáról van szó, nem meglepő az idők során köré épült kultusz. Szerencsére a kellemes emlékek rengeteg emberben jelen vannak, így fejlesztő csapatok foglalkoznak a ***„Mario-hangulat”*** életben tartásával. Magáról a játék menetéről nem igazán érdemes írnom, hiszen aki nem tudja miről van szó, azt valószínűleg egyáltalán nem érdeklik a játékprogramok. ***GPL*** licenc alatt két igényes ***remake*** verziót érdemes megemlítenem: egyiküket ***Mega Mario***-nak hívják, a másik ***Secret Maryo Cronicles*** névre hallgat.

1. ábra A Mega Mario linuxos verziója

Mega Mario

Ez a változat a ➲ ***http://mmario.sourceforge.net*** címen érhető el, forrás és bináris formában egyaránt. Elsősorban ***Win32*** felületre készült szoftverről van szó, csak nemrég kapta meg a linuxos verzióját (***Hans de Goede*** munkájának nyomán). A játék kódja ***C***-ben íródott, futásának legfőbb feltétele a naprakész ***SDL*** könyvtárak jelenléte. Képi világa csupán ***640x480px*** felbontást enged, ott is csak „jó közepes” látvánnyal, cserébe viszont gyenge hardveren is szépen fut: egyéb megvalósításokkal szemben a ***Mega Mario*** nem kíván ***3D*** grafikus hardvert.
A forráskódot a szokásoktól kissé eltérően kell használni: a kicsomagolt archívban, ***root*** jogokkal kiadott `make PREFIX=/usr/local`, `make PREFIX=/urs/local install` parancsokra épül fel a bináris (ahol az ***/usr/local*** paraméter a szabadon választható, kívánt elérési út). Kevés programnál szoktam ajánlani az előre fordított verziók használatát, ellenben itt jó tapasztalataim voltak a „gyári” megoldásokkal is. Tehát, ha valaki nem szeretne vesződni a forrásállománnyal, válassza bátran a könnyebben járható utat: a kibontott, megfelelő ***tarballban*** található ***megamario*** állomány indítja a kész játékot.
Egy apró észrevétel: nem értem mi indokolja, de az általam tesztelt verzióban valami fröcsögő, piros

2. ábra Íme, a Secret Maryo Cronicles

folyadék (nevezzük vérnek) nagy szerepet játszik bármely modell elhullásakor. Javaslatom szerint a főmenüben használja mindenki a *„Blood off”* kapcsolót: egy ilyen programban a vér megjelenítésének nincsen helye.

Secret Maryo Cronicles

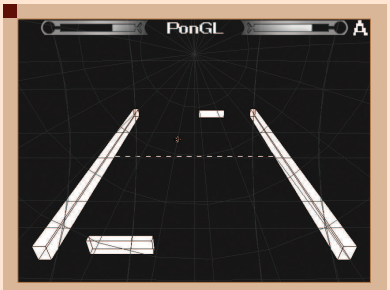
Talán ez *Mario* klón mondható legigényesebbnek a saját osztályában! A *Megamario*-nál szebb, sőt valamivel többet is tud nála. A zenei kísérete egyszerűen mesteri, pályái pedig kellően nehezek. Ellenben problémásabb a felélesztése, ráadásul a friss verzió már *3D* gyorsítókártyát kíván futásához. Azért nem kell csúcshardverre gondolni: egy régi, *Riva TNT* teljesítményű, *GLX* vagy *DRI* kapcson keresztül használható darab pont megteszi. A szoftver elérhető a ➲ *http://www.secretmaryo.org* honlapon, forrás és bináris formában egyaránt. Az előre fordított bináris használatát nem javaslom senkinek: nálam percenként összeomlik (feltéve, hogy egyáltalán hiba nélkül elindul). Foglalkozzunk inkább a forráskóddal: töltsük le, bontsuk ki, majd építsük fel!

Függőségként szükség lesz a friss *SDL* könyvtárak és a *CEGUI* csomag jelenlétére: ha ezekkel rendelkezünk, a forrásmappa gyökerében adjuk ki *rootként* az `sh autogen.sh` parancsot, ezután a szokásos `./configure`, `make`, `make install` utasításokat. (Ha valaki olyan rendszert használ, mely nem rendelkezik az *automake*, *autoconf* párossal, akkor a *scons* fordító használatával is megpróbálkozhat.) Válaszul rövid időn belül felépül a kívánt *smc* bináris, melynek segítségével felhasználóként indítható a móka. A játék menüjében a felbontás több lépcsőben változtatható: ha valakinél *Mario* szaggatottan mozogna, akkor éljen ezzel a lehetőséggel. A nehézségre való tekintettel itt egyébként akármikor elmenthetjük az aktuális állást, de érdemes finoman bánni a segítséggel: háromezer megszerzett pontunk bánja „körönként”, ráadásul amíg nincs ennyink, addig nem is elérhető ez a művelet.

Pong

Egyszer volt, hol nem volt... a számítógépes játékok világa. A kezdeti szárnypróbálgatások jelentős hányada az *Atari* gépeit, illetve a korai 8 bites masinákat érintette. Az ügyességi vonalat ekkor még elsősorban a *Pong* játékok képviselték. A következőről volt szó: a képernyő szélén, egy síkban mozgatható ütő segítségével kellett visszaütni a labdát a számítógép játékosának (vagy más játszótársnak), lehetőleg úgy, hogy ő ne tudjon hasonlóképpen válaszolni. Igen, tudom... az ötlet roppant primitív, de nagy karriert futott be, mivel egyúttal rém lényegre törő is. A legegyszerűbb klónkat emiatt nincs értelme megmutatnom, viszont két megoldás itt is kilóg a sorból.

3. ábra Arcana már vesztésre áll



4. ábra A perspektívába fordult asztal

TuxPuck

A *TuxPuck* elsősorban a grafikai munkái miatt látszik ki az átlagból: szépen megrajzolt, döntött elölnézeti képpel rendelkezik – miközben használatához nem kíván erős gépet, sem pedig *3D* grafikai eszközt. Számítógépes ellenfelünket két egység közül választhatjuk: játszhatunk *Arcana* ellenében, esetleg kihívhatjuk *Tux*-ot is egy meccsre. Előbbit viszonylag egyszerű legyőzni, ám ha rosszul áll a szénája, nem restell varázsolni egy keveset. Ilyenkor nem érinti meg ütőjével a korongot, hanem körkörösen mozgatja, majd kiszámíthatatlanul és villámgyorsan lepasszolja valamelyik hozzánk tartozó sarokba. De sokkal inkább a hupikék pingvin a keményebbik dió.
Vele kapcsolatban az első élmény elég kellemetlenre szokott sikerülni: az avatatlan kezeket gátlástalanul, minden-semmi arányban alázza le. Egyetlen fegyver vele szemben a lapos szögben lőtt, ide-oda pattogó korong: a sűrű mandinerből érkező „támadásokat” nem képes kivédeni. Persze ő majd megpróbál tenni arról, hogy sok lehetőségünk azért ne legyen lapos szögben passzolni... Lássuk, hogyan éleszthető fel a *TuxPuck*, *Linuxon*! Előre fordított verzió nem létezik, de olyan egyszerű dolgunk akad, hogy nem is lesz szükségünk rá. Látogassunk el a ➲ *http://home.no.net/munsuun/tuxpuck/* címre, és töltsük le a forrás *tarballt*. A kicsomagolt archívban adjuk a ki a `make` parancsot. Ha az *SDL* könyvtáraink rendben vannak, helyben felépül a *tuxpuck* nevű *ELF* bináris állomány. Ez a nagyjából 500 *Kbyte* méretű fájl önmaga tartalmazza a futásához szükséges összes adatot, így bárhová bemásolhatjuk (javaslatom szerint, az */usr/local/bin* mappát praktikus használni), majd felhasználóként indíthatjuk. Természetesen a forrás könyvtár a sikeres műveletek után törölhető.

PonGL

A *PonGL* igazi különlegesség. A képi világa elsőre puritánnak tűnik, azonban játék közben érkezik a „feketeleves”. Ha bárki nyeregben érezné magát passzolgatás közben, akkor az asztal látványosan elfordul *x,z(+;-90fok)* irányban. Mondanom sem kell, a kézre álló irányításnak ebben a pillanatban gyakorlatilag lőttek. Emiatt közepes fokozaton már elég nehéz nyerni a számítógép ellen.
A program több felbontásban használható, kérésünkre teljes képernyőn és ablakban egyaránt működhet, illetve még a nehézségi fokozata is állítható. Munkára fogható szóló üzemmódban (fallabda játék), gépi ellenfelet választva, illetve hús-vér játékos ellenében is.
Látogassunk el a ➲ *http://pongl.opengl-fr.com* oldalra, és töltsük le a forrás archívot! Csomagoljuk ki, majd a mappa gyökerében *root* jogkörrel adjuk ki a `make` parancsot. Ha a rendszerünk *SDL* könyvtárai, *Mesa* és *Glut* csomagjai naprakészek, akkor rövid időn belül felépül az indítható állomány. A létrejött *PonGL* fájlt (a mellette lévő */data* könyvtárral együtt) tegyük egy mindenki által olvasható területre (például a */usr/local/games/pongl* pont megfelel), majd a binárist linkeljük ki valamelyik alapértelmezett elérési útra (esetemben a láncolás így néz ki: `ln -s /usr/local/games/pongl/PonGL /usr/local/bin/pongl`).
A szimbolikus linket használva máris indítható meccs...

Konklúzió, öregesen

A négy, röviden bemutatott programot bűn lenne kihagyni. Ha valakinek vannak pici gyermekei, ezekkel a játékokkal akár hónapokra megoldhatja az egész család kikapcsolódását... Sőt! Szégyen, nem szégyen, én is hosszú órákig szórakoztam mindegyikkel, miközben lassan a harmincat taposom. A *Secret Maryo Cronicles* egészen odáig kalauzolt, ahol már a programkód fulladt ki (a tizenötödik pálya körül a fejlesztők még adósak némi munkával)... Nem is tudom másképpen leírni az önfeledt retro élményt adó a játékokat: „kis mókusok kenyérre is kenhetik, nagy mókusok sütéménybe is tölthetik!”.

Kovács Zsolt (kovi@linuxforum.hu)

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