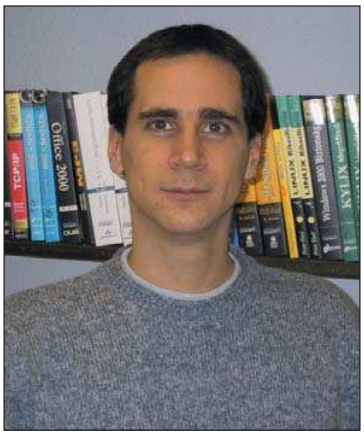
Az informatika világa rendkívül gyorsan változik. Sőt a Linux világa, ha lehet mondani, még gyorsabban.
A Linuxvilág kiadójánál, a Kiskapunál hosszú évek óta foglalkozunk angol szakszövegek fordításával, ezért is volt érdekes és külön kihívás a számomra a Linuxvilág magazin kiadása. Aki már igyekezett minőségi fordítást készíteni, tudja, hogy a linuxos angol szófordulatokat, kifejezéseket rendkívül nehéz helyesen átültetni a magyar nyelvbe. Olyannyira, hogy sok kiadó és szerző nem is igyekszik magyarul írni, hanem az angol szakszavakat használja (fonetikus vagy angol írásmód mellett), majd – mint aki jól végezte dolgát –, dülleszti a mellét. Kiadónk egyik fontos céljának tekinti, hogy a félképzett műszakemberek miatt óriásivá mélyült szakadék fölé, amely a magyar nyelv és a szakmai kiadványok nyelvezete között tátong, hidat állítson, vagy legalább kifeszítsen egy kötelet.
Az elmúlt években nagyon sok próbálkozásunk volt, melyek legtöbbjének kedves olvasóink voltak a „kísérleti nyulai". Ezúton is külön szeretném megköszönni azt a rengeteg támogató, vagy éppen támadó, gyakran fájóan őszinte véleményt, levelet, amelyekkel segítettetek nekünk abban, hogy egészséges egyensúlyt találjunk ezen az ingoványos területen. A szóhasználat mára eljutott oda, hogy szinte minden szakszónak van értelmes, pontos és jól használható magyar változata, viszont használatuk, pontos jelentésük, megfeleltetésük gyakran még nekünk is komoly nehézséget jelent. Gondolok itt olyan esetekre, amikor egy jövevényszó „forrásszava" több értelemben is szerepel a külföldi irodalomban (nem is beszélve azokról a szavakról, melyeket még az angol nyelvterületen is kevernek), vagy amikor egy idegen szó a szaknyelvben fonetikus leírás mellett az eredetitől akár homlokegyenest különböző értelemmel honosodott meg.
Olvasóink részéről teljesen jogos az igény, hogy pontosan értsék a leírtakat, ezért januártól ezzel a szótároldallal igyekszünk segíteni a lapban megjelenő szövegek értelmezését. Ez a szótár természetéből adódóan sok olyan szót tartalmaz, amit még nem lehet „ipari szabványnak" tekinteni, ezért kérek mindenkit, ha van rá ideje, kedve, írja meg nekem véleményét, ötletét, kérdéseit a Szy.Gyorgy@Linuxvilag.hu levélcímre.
Lássuk akkor az e havi szavakat!

administrator → ***rendszergazda***
átirányítás → ***redirection***
áttetsző – ***transparent***: a rendszer egy olyan eleme (általában külön rétegként beszélnek róla), amely a rendszer használója számára „láthatatlan".
betörő – ***cracker***: rosszindulatú behatoló, aki a célgépen kárt okoz, vagy onnan értékes adatot másol le. Nem keverendő össze a betyárral.
betyár – ***hacker***: magasan képzett számítógépes szakember, aki a károkozás szándéka nélkül hatol be idegen rendszerekbe. A betyár szóval kapcsolatban nagy vita alakult ki, több „hacker" kedveli az elnevezést, és büszkén vallja magát betyárnak, mások visszás érzelemmel viseltetnek a szó rosszalló mellékjelentése miatt.
címke – tag: a HTML nyelvben az elemeket címkének nevezzük. Az angol tag (ejtsd: teg) szóval azonos magyar tag szót is használják erre a fogalomra.
client → ***ügyfél***
cracker → ***betörő***
cursor → ***kurzor***
csomag → ***package***
firmware → ***gépszintű program***
folyam – ***stream***: adatfolyamnak is hívják. Olyan adatcsatornát jelöl (például csővezeték, tárolóeszköz vagy hálózati szórt műsor), amely folyamatos vagy nagy mennyiségű, nem darabokra bontott adatforrásként működik.
gépszintű program – ***firmware***: olyan program, amely az alkatrészekkel közvetlenül tartja a kapcsolatot, és feladata elsősorban programozási felületet biztosítani az alkatrész(ek)hez. Ilyen gépszintű program a PC-ben található BIOS. Használják még: gépi szinten futó program, a gép belső programja, gyári program, alkatrész- vagy eszközprogram.
hacker → ***betyár***
hardver – ***hardware***: sajnos a magyar szóhasználatban nem elég pontosan értelmezett gyűjtőfogalom. A szó eredeti jelentése: a gép azon „része", amelyet meg lehet fogni. Használata kizárólag általános gyűjtőszóként javasolt. Kiváltására általában a vas, a gép, az alkatrész, az eszköz, a ház vagy a tárgy szavak valamelyike alkalmas.
hardware → ***hardver***
héj – shell: a parancsértelmező, például bash, tcsh.
helyőr → ***kurzor***
hivatkozás – link: olyan bejegyzés a fájlrendszerben, ami egy fájlra vagy könyvtárra mutat. Kétféle hivatkozás van: a közvetett (symbolic, soft), illetve a közvetlen (hard). Közvetlen hivatkozás esetén a fájl többszöri bejegyzéséről is beszélünk.
hurok – loop: olyan eszköz, függvény vagy csatoló, amelynek a kimenete közvetlenül a saját bemenetére van irányítva, például ilyen az `lo` hálózati csatoló.
időbélyeg – timestamp: egy pontos dátumot (és időt) tartalmazó karaktersorozat vagy változótípus, naplózásnál, adatbázis-műveleteknél stb.
kapu – port: az adatátvitelnél használatos kapcsolódási pont, akár alkatrész, akár csak egy kiszolgálódémon. Soros kapu, ssh-kapu (22-es) stb.
kurzor – cursor: a billentyűzetről bevitt karakterek megjelenési helyét jelző négyzet vagy jel. Létezik magyar megfelelője, a helyőr, de sajnos kevesen használják. A beszúrási pont is elterjedt.
leképezés – rendering: a grafikában használatos kifejezés, amikor a számítógép egy adott képet a megadott képletek (stb.) felhasználásával kiszámol. Animációs filmeket például gyakran képkockánként képeznek le óriási géptelepek.
link → ***hivatkozás***
loop → ***hurok; ciklus***
mutató – pointer: az egérrel (vagy egyéb mutatóeszközzel) irányított nyíl (más alakzatot is felvehet) a munkafelületen; hívják egérmutatónak is. Van, hogy az egérmutató helyett hibásan a kurzor szót használják.
önhívó – ***recursive***: olyan folyamat vagy függvény a programozásban, amely valamilyen feladatot saját maga ismételt meghívásával old meg. Közismert példa a faktoriálist számító függvény: fakt(i) {return (i * if (i>1; fakt(i-1); 1)}
package → ***csomag***
pointer → ***mutató***
port → ***kapu***
recursive → ***önhívó***
redirection → ***átirányítás***
rendering → ***leképezés***
rendszergazda → ***administrator***
server → ***kiszolgáló***
shell → ***héj***
software → ***szoftver***
stream → ***folyam***
szoftver – software: a hardverhez hasonlóan rendkívül gyakran hibásan használt fogalom. Eredeti jelentése: a gép azon része, amely „kézzel nem fogható". Fogalmak, melyek helyett hibásan használják: program, programrendszer, (dobozos) termék, szolgáltatás stb.
tag → ***címke***
timestamp → ***időbélyeg***
transparent → ***áttetsző***

Beköszöntő



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

Boldog új esztendőt!

Nagy örömmel zárjuk le a második évet, hiszen egyre közelebb kerülünk egyik fő célunkhoz. A Linuxvilág elindításakor elsődleges célunk a hazai Linuxhasználat támogatása, megerősödésének elősegítése volt. A GNU/Linux világa pedig gyönyörűen fejlődik. Elindult és erősödik több Linuxszal foglalkozó hazai cég, országszerte találhatók felhasználói csoportok, az oktatásban szinte mindenütt felfigyeltek a Linuxra és így tovább. Nem mondom, hogy a helyzet rózsás, de mindenképpen ígéretes híreket hallani mindenfelől: teljes vállalatok állnak át Linuxra, komoly pályázatok születnek, és ismét kezd felélédni a linuxos közösség.

Természetesen a két év alatt mi is sokat fejlődtünk. A nyelvi viták például nagyon komoly változtatásokra késztettek, a szemközti oldalon lévő Szótár ennek ékes bizonyítéka. A szótár igazából nem pontos meghatározás, hiszen sem teljesnek, sem részletesnek nem nevezhető. Mégis igyekszünk minden olyan szót kigyűjteni, ami az adott lapszámban megtalálható és értelmezése nem pontos. Sajnálom, hogy nem maradt hely a rendes magyarításokkal még nem rendelkező szavak leközlésére és megvitatására, hamarosan azonban üzembe helyezzük a ➲ http://www.linuxvilag.hu weboldalt, ahol külön fórum keretében foglalkozunk a témával. Indítottunk egy kereshető szótárt is, melyben példákat is igyekszünk megadni, illetve egyes szavakkal részletesebben is foglalkozni. Ha van időtök, látogassátok meg a ➲ http://www.szotar.kiskapu.hu címen!

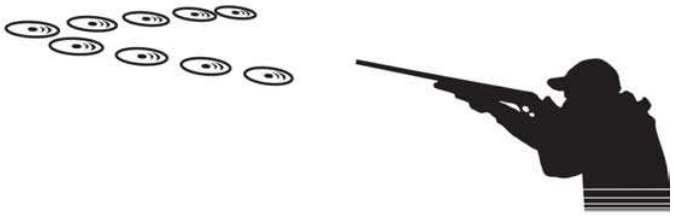
De nézzünk egy pillanatra körbe, hogy milyen érdekes cikkek kaptak helyet e havi számunkban! Nagyon sok levelet kaptam, hogy írjunk több megvalósításról. Szép és jó a Linux, de ha nem lehet éles irodai munkában használni, semmit nem ér. Igyekszünk tehát tényleges kiépítéseket, linuxos rendszereket használó vállalatokat bemutatni. Ebben a hónapban egy nemzetközi cég magyar leányvállalatában ***Lippai Gergő*** vezet körbe minket (Linux alapú irodai hálózat megvalósítása – 1. rész, 38. oldal), valamint egy amerikai cég kiépítésével is megismerkedhetünk, ***Craig Swanson*** és ***Matt Lung*** vezetésével (OpenLDAP mindenütt, 28. oldal).

Rendszergazdáknak szól néhány egyéb cikkünk is. E hónapban a biztonság kapcsán elővesszük az FTP-szolgáltatást, és ***Mick Bauer*** vezetésével elrejtjük egy helyi proxy mögé (a SuSE egyik megoldását használja). Vannak nehezebben emészthető cikkeink is. Ide tartozik például ***Greg Kroah-Hartman*** írása a soros illesztőprogram-rétegről (47. oldal). És ha elfáradtunk, foglalkozzunk egy kicsit játékosabb témákkal is. ***Garzó András*** írja le kedvenc játékait (Linux és játék, 62. oldal), és a Mustra rovatban egy négyfalas kivetítőrendszerrel is megismerkedhetünk, amit Linux vezérel (Négyfalas 3D-s megjelenítő Linuxra, 72. oldal).

Fontos és mindenképpen említésre méltó téma az 50. oldalon található PostgreSQL rendszert bemutató írás ***Nyíri Imre*** tollából. Nagyon sokan vannak ugyanis, akik a Linuxot elsősorban kiszolgálóként kívánják használni. Itt pedig igen gyorsan eljut az ember az adatbázisokkal kapcsolatos kérdésekig. Milyen adatbázis-kiszolgálót használjunk? Nálunk is hasonló helyzet állt fenn. A régi programrendszerek még Windows alatt készültek, ezek Delphivel kergették az InterBase-kiszolgálót. Azután az InterBase is sokat változott (fizetős termékből átváltott nyílt fejlesztésbe), de röviden összefoglalva most komolyan azon gondolkodunk, hogy milyen rendszerre váltsunk. Az egyik komoly esélyes éppen a PostgreSQL.

Jó olvasást kívánok mindenkinek!

Programvadászat



Az év első korongját kapták kézhez kedves olvasóink, amin az elmúlt évet mintegy összegezve közzéteszünk néhány „nagykorú” programot. Ezek között a programok között megtalálható az OpenOffice.org, az AbiWord és a Mozilla.

Linuxos iroda

OpenOffice.org

A mellékleten ennek a tekintélyes méretű irodai alkalmazásnak a magyarított változata kapott helyet, ezzel is segítve a kezdő linuxosokat, hogy legalább a nyelvi nehézségekkel ne kelljen megküzdeniük. Magyar nyelven telepíthető és állítható be.

Abiword

Ha engem kérdeznek, hogy a kis gépeken mivel célszerű szöveget szerkeszteni, szerintem az Abiword a válasz. A minap azonban egy érdekes hibára lettem figyelmes: az OpenOffice.org-ból mentett ***.doc***-fájlt – egy 30 soros táblázattal – hibásan nyitott meg. Szerencsére ezt, úgy tűnik, a fejlesztők ebben a változatban már kiküszöbölték.

Mozilla

A Mozilla böngésző méltán népszerű – ennek a programnak a magyar nyelvű változata és a magyarításhoz szükséges

.xpi-fájlok is helyet kaptak a korongon, természetesen a legújabb angol nyelvű csomagokkal egyetemben.

Rendszermag

Olvasói kérésre a teljes rendszermagfát a foltokkal együtt tesszük közzé, így azok az ínyencek is megtalálhatják a nekik megfelelő összeállítást, akik nem rendelkeznek széles sávú internetelérással.

Kiszolgálók

Ezentúl a kiszolgálók könyvtár szintén állandó vendége lesz korongunknak, ide kerülnek a mindenféle kiszolgálóprogramokkal kapcsolatos frissítések. Ebben a hónapban az Apache, a MySQL, a Samba és a PostgreSQL adatbázis-kezelő kapott helyet rajta.

Apache HTTP SERVER PROJECT

- Az Apache a legnépszerűbb webkiszolgáló program. Korongunkon az 1.3-as sorozatból az 1.3.27-es, illetve a 2-es sorozatból a 2.0.43-as forráskódja kapott helyet.
- A MySQL egy „kicsi” és gyors adatbáziskezelő-rendszer, webes alkalmazásokhoz azonban tökéletesen megfelel. Ennek a 3.23-as megbízható és a 4.0-s fejlesztői változatát adjuk közre.
- A Samba kiszolgálót úgy jellemezték, mint kapocs a rendszerek között – aki használja, az biztosan valami hasonlót gondol róla. Megbízható, jól testreszabható és „Windosból is látszik”.
- A másik adatbázis-kezelő, a PostgreSQL már nagyon összetett alkalmazás. Azok, akik az Oracle után valami ingyenesen használható programra szeretnének áttérni, és nem kívánják feladni a régi beidegződéseket sem, azok számára is nyugodt szívvel ajánlhatom.
- A kiszolgálók táborát bővíti a SUN ONE Application Server 7 is. Ez a program, ahogyan a neve is mutatja, egy alkalmazáskiszolgáló csomag. Egyelőre csak Solaris Sparc és Microsoft Windows operációs rendszerek alá érhető el, azonban gőzerővel folyik a linuxos változat fejlesztése is. A Sun alkalmazáskiszolgálója megoldást nyújt a fejlesztés és az üzemeltetés egyidejű kihívásaira, miközben megfelelő alkalmazásfelületet nyújt a kényes működésű rendszerek számára – mindezt úgy, hogy az iparági szabványként elfogadott J2EE alapokon működik. Magyarországon most először szerepel CD-mellékleten a Sun ONE Apps Server ingyenes változata. Amint elérhetővé válik a linuxos változat, természetesen azt is azonnal közreadjuk.

Magazin

Mint ígértük, az előző szám CD-mellékletéről lemaradt anyagok is helyet kapnak ezen a korongon. A Magazin/Elozo_szam könyvtárban lévő összes anyag az előző számunkhoz kapcsolódik. A könyvtárak tartalma az alábbiak szerint alakult:

LME_konf – Az idei LME konferencia anyagait tartalmazza pdf formátumban.

LTSP – A Linux Terminal Server Project fájljai a 75. oldalon kezdődő „Kóstoljunk bele a tisztán linuxos irodába” írásunkhoz.

QT – A 38. oldalon kezdődő „Grafikus felület programozása QT-ban” című írásunk példaprogramjainak forráskódjai itt találhatók meg.

xcdroast – A „CD-írás lépésről lépésre” című íráshoz kapcsolódóan az X-CD-Roast programot tartalmazza.

E havi magazinrészlegünk

3D – A 72. oldalon található „Linuxos teleppel meghajtott négyfalas kivetítő” című írásunk kapcsolódó címei és a szakirodalmi ajánlás érhető el itt.

Falatka – A Falatka rovatban leírt programok forráskódjait, illetve néhány előre elkészített csomagot tartalmaz.

Folyamat – A 34. oldalon kezdődő „Folyamat-nyilvántartás” cikkekhez tartozó forráskódlisták és a kapcsolódó címek vannak itt.

PostgreSQL – Erről a méltán népszerű adatbázis-kezelőről szóló cikkünk kapcsolódó anyagai, a forráskódok, könyvtárak (lib) és a lefordított programok találhatók itt.

OpenLDAP – A 28. oldalon található „OpenLDAP mindenütt” című írásunk kapcsolódó címeit rejti.

proxy-suite – Az 54. oldalon kezdődő „A biztonság növelése FTP proxy használatával” című írásunkban szereplő proxy-suite program található itt.

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

Zaurus az IBM támogatásával

Az IBM és a Sharp összefogása némi hátszelet adhat a sokat emlegetett Linux alapú kézi gép, a Zaurus elterjedéséhez. Az Enterprise Edition Zaurust közös megoldásként olyan vállalatoknak ajánlják majd, amelyek dolgozói nem köthetők le egy asztali számítógép elé, mégis folyamatos, átlátszó hozzáférést igényelnek a vállalati adatokhoz és programokhoz. Az alkalmazási lehetőségek között említhető a rendelések vagy a raktárkészlet szükség szerinti ellenőrzése, az elektronikus levelek olvasása, a címjegyzékek vagy a naptári bejegyzések központi kezelése – és természetesen bármi más, amire az érdekeltek megfelelő alkalmazást találnak vagy készítenek. Az alkalmazásfejlesztést mint a megoldás sikerének alapfeltételét a linuxos alap, a Java-futtatási lehetőségek és az IBM támogatása hivatott szavatolni. A két cég később a beszédfelismerés képességével is el akarja látni a készülékeket, illetve angoltól eltérő nyelvű változatok kiadását is tervezik. Az IBM és a Sharp közös terméke 2003 közepétől lesz elérhető, elsőként az amerikai, majd később egyéb piacokon is.

A Xeonok világa asztali gépekben is

Az Intel bejelentette 3,06 GHz-es Pentium 4 processzorát – ez az asztali számítógépekbe szánt lapkák közül az első, amely 3 GHz fölötti órajellel működik. A másik, ennél sokkal fontosabb újdonsága, hogy az eddig csak az Intel Xeon processzoroknál alkalmazott Hyper-Threading (HT) megoldással is felvértezték, ennek köszönhetően a teljesítménynövekedés – megfelelő környezetben – jóval nagyobb lehet annál, mint ami csupán az órajel megemeléséből következne.

A HT révén ugyanaz a processzor két programszálat tud párhuzamosan futtatni, így lényegében két fizikai processzornak látszik az alkalmazások és az operációs rendszer felé. Ha megfelelő alaplappal és a többszálúságot támogató operációs rendszerrel rendelkezünk, 20–30 százalékos teljesítménynövekedést tapasztalhatunk, feltéve, hogy több programot is futtatunk egyszerre, illetve többszálú alkalmazást használunk.

➲ http://www.intel.com/technology/hyperthread

Vedd ölbe a monitort!

Alig fogtuk fel, miért jó nekünk a táblagép (Tablet PC), a Microsoft és holdudvara máris újabb ügyes kis jószággal örvendeztet meg bennünket. Az okos kijelzők (Smart Display) nagyon hasonlítanak az LCD-kre, ám saját processzoruk, memóriájuk és akkumulátoruk van; 802.11b szabvány szerinti vezeték nélküli hálózati kapcsolatot képesek teremteni, így a lakás bármely zugába

bátran magunkkal vihetjük őket. Mivel érintőképernyők, a hozzájuk tartozó tollal vagy akár az ujjunkkal különféle műveleteket végezhetünk el rajtuk. Azt mindegyik gyártó lelkesen hirdeti, hogy az új kijelzők mennyire kényelmesek – ezt nem nehéz belátni –, és a hozzájuk tartozó asztali számítógépnek Windows XP Professional operációs rendszert kell futtatnia, de a műszaki részletek tekintetében sajnos elég szűkszavúak. Ha minden igaz, magán a kijelzőn a Windows CE .NET rendszer Windows CE for Smart Displays nevű változata fut, ám szolgáltatásaiban túlnyomórészt az asztali számítógépre, illetve a Citrix Metaframe vagy egyéb RPD (távoli asztal protokoll) megoldások segítségével valamilyen távoli gép erőforrásaira támaszkodik.

Mivel a kijelzők – gyártótól függően – például Intel X-Scale vagy National Geode processzort tartalmaznak, hálózati kapcsolatuk pedig szintén szabványos, nem sok képzelőerő kell ahhoz, hogy a nem Windows operációs rendszerek távoli munkaasztal-képességeit gondolatban átültessük az új eszközökre. Azon pedig csak egy csendes estén, egy pohár jó bor mellett érdemes elmélkedni, hogy az X-felület alatt kezdetektől fogva meglévő lehetőséget újra fel kellett találni…

Mindentudó UHU

Az UHU-Linux Kft. bővíti kínálatát, terjesztéséből egy Szerver és egy Tűzfal változat fejlesztése is megkezdődött. Az előbbivel a könnyű telepítést igénylő, elsősorban más operációs rendszerről átálló rendszergazdákat célozzák meg, szem előtt tartva az egyszerű felügyeleti lehetőséget, a sokoldalúságot és a szabványokhoz való igazodást. A kiszolgálókra szánt terjesztés első nyilvános változatának megjelenése decemberre várható. A Tűzfal változat szintén decemberben lát napvilágot. Alapja egy megerősített rendszermag lesz, sokoldalú szolgáltatásait pedig könnyen kezelhető beállítófelület segítségével felügyelhetjük.

Az UHU-Linux sikeréhez az is hozzájárulhat, hogy a munkaállomás-változatban helyet kapott a LafiSoft Kft. számlázóprogramja. A programot 100 bizonylatig vagy hárommillió forint forgalomig ingyenesen lehet használni, így a kisvállalkozások számára lényeges költségmegtakarítást jelenthet.

➲ http://www.uhulinux.hu

Apache-megoldások x86–64 gépekre

Az AMD bejelentette, hogy a Covalent, egy Apache alapú vállalati megoldásokat fejlesztő társaság és a Red Hat közösen fejleszt Apache alapú kiszolgálót az AMD elvileg jövő év első felében megjelenő, 64-bites Opteron processzorára. A fokozatos áttérés jegyében megoldásaik mind a 32-, mind a 64-bites processzorokat támogatni fogják, termékeik segítségével a felhasználók zökkenőmentesen, az igényelt teljesítmény függvényében térhetnek át 32-bites rendszerükről 64-bitesre.

➲ http://www.covalent.net

nVidia GeForce FX

Mozifilm-minőségű játékok ígéretével csábít az nVidia új, GeFoce FX nevű 3D lapkáinak bejelentésével. A GeForce FX lesz az első GPU, amely 500 MHz-es órajellel működik, a mellé kerülő 1 GHz-es DDR2 memóriáknak – ezeket egyébként a Samsung szállítja – és az AGP 8× felületnek köszönhetően másodpercenként négymilliárd képpont megjelenítésére képes. A lapka számos új vagy megújult megoldást támogat, teljes DirectX 9.0-támogatást biztosít, a színeket 128 biten ábrázolja és 0,13 mikronos eljárással készül. Természetesen máris számos játékgyártó és -forgalmazó cég felsorakozott az nVidia mögött, így várhatóan lesz mivel kihasználni a jövő februárban megjelenő grafikus kártyák képességeit.

➲ http://www.nvidia.com

ATI-hírek

Az asztali gépekbe szánt kártyái közül a frissen bejelentett ALL-IN-WONDER 9700 PRO az ATI csúcsmodellje. A DirectX 9.0 gyorsításra képes Radeon 9700 PRO VPU és Theater 200 lapkával szerelt kártyára 128 MB memória kerül. A kártya – többek közt – képes valós idejű, 720×480 felbontású, 30 képkocka/másodperc képfrissítésű MPEG-2 tömörítésre; sztereó vevője 125 tévécsatornát tud fogni; 10 méteres hatótávú rádiós távirányítóval rendelkezik; és további TV Wonder vagy TV Wonder VE kártyával párosítva egyszerre akár két tévécsatorna megjelenítésére is alkalmas. Bőséges a dobozban megbújó programcsomag is, videoszerkesztési igényeink kielégítésére a Pinnacle Studio 8 és a MatchWare Mediator 7 programot mellékelik. A kártya egyetlen szépséghibája hat számjegyű ára forintban.
Az ATI töretlenül támogatja kártyáinak Linux alatti használatát: megjelent egyesített illesztőprogramjának 2.4.3-as változata. Az új változat megbízható OpenGL 1.4-es támogatást biztosít, és a Radeon 8500, 8500 LE, ALL-IN-WONDER 8500 DV, 9000 PRO, 9500 PRO, 9700 PRO, ALL-IN-WONDER 9700 PRO és FIRE GL kártyákat ismeri.
➲ http://www.atitech.com

UnitedLinux-bejelentések

Megjelent a SuSE Linux Enterprise Server 8.0-s változata, az első olyan vállalati SuSE Linux-terjesztés, amely a UnitedLinux 1.0-s szabványon alapul. A rendszer mySAP.com és IBM DB2 tanúsítvánnyal is rendelkezik, Intel és AMD 32- és 64-bites processzorokon, HP ProLiant Server, HP Itanium 2 Server, Fujitsu Siemens Primergy Server és IBM eServer gépeken egyaránt használható. Az SCO Linux Server 4.0 az első olyan Linux-termék, amelyet az SCO Group az egységes Linux jegyében „Powered by UnitedLinux”-ként jelentet meg. Az SCO Linux Server egy kimondottan üzleti felhasználásra készült operációs rendszer szabványos Intel processzoros kiszolgálókra. Az SCO Linux Server UnitedLinux magja szabványos, megfelel az LSB és az Openi18n (korábban Li18nux) követelményeinek, amely a megbízhatóságra, a magas rendelkezésreállásra és a biztonságra helyezi a hangsúlyt.

Palm OS kicsiben

A Fossil neve inkább a divatszakmában, semmint a számítástechnikában lehet ismert, mégis érdemes néhány szót szentelni a cégnek, pontosabban új termékének, a 4.1-es Palm OS-t futtató karórának. Ez a világ első olyan karórája, amely Palm OS operációs rendszert használ, és fontos megjegyezni, hogy nem valami bumfordi próbapéldányról, hanem valóban hordható, divatos termékről van szó.

A karóra aprócska kijelzőjén lényegében egy kézi gép képernyőjének tartalma jelenik meg, a jól megszokott ikonokat és egyéb kellékeket láthatjuk rajta, és a kezelése sem fog meglepetéseket okozni. A karóra 2 MB memóriával gazdálkodik, 160×160 képpontos kijelzőjét háttérvilágítással látták el, tolla pedig a szíjba épül be. Egy aprócska infravörös kapunak is jutott rajta hely, használatát pedig az oldalán elhelyezett gombok segítik. További érdekessége, hogy használója testre szabhatja a kijelző karóra módban megjelenő tartalmát.
➲ http://www.fossil.com

GB méretű CF kártyák

A Pretec elsőként jelentett be 1,5, 2 és 3 GB kapacitású CompactFlash kártyákat – fizikai megjelenésüket tekintve a hasonló méretű memóriakártyák között még nem találni ekkora tárkapacitásút. Első hallásra feleslegesnek tűnik egy ekkora kapacitású kártya, hiszen ki akarna mondjuk hordozható MP3-lejátszójában sok száz órányi zenét magával vinni; ám arra gondolva, hogy például a 4–6 megapixel felbontású digitális fényképezőgépek elérhető áron kaphatók, és ha jó minőségű képekre vágyunk, gyorsan fogy a hely, már nem is olyan sok az az 1–2 GB.

A cég termékeit emellett katonai kivitelű változatban is gyártja, ezek -40 és +125 °C között használhatók, és a viszontagságos körülményeknek is ellenálló borítással bírnak.
➲ http://www.pretec.com

A mindent ismerő Verbatim DVD

Ha Nagy Sándor ma élne, Verbatimnak hívnák, a gordiuszi csomót pedig DVD-nek. A cég ugyanis rájött arra, hogy rengeteg felhasználót elriaszt az írható DVD-formátumok körül kialakult csetepaté. Mivel minden gyártó a saját ötletét tartja a legjobbnak – vagy inkább a saját fejlesztése után szeretne díjat szedni –, a gyártó cégek két csoportra oszlottak. Az egyik csoport a mínuszos, ők a DVD-R és DVD-RW lemezeket támogatják, a másik csoport a pluszos, ők DVD+R és DVD+RW jelzéssel futtatják termékeiket. A Verbatim új meghajtója mindkét csoportosulás termékeit kezelni fogja, illetve az összes CD-s formátumot is ismeri. Ára 80 ezer forint körül mozog majd, ha néhány hónap múlva megjelenik az üzletek kínálatában.

Vajon az egységes formátumoké a jövő?

Az XML-formátumok fejlesztését felvállaló szervezet, az Organization for the Advancement of Structured Information Standards, röviden OASIS újonnan létrehozott bizottsága egységes, gyártótól független irodai fájlformátumok kidolgozását tűzte ki céljául. Az Open Office XML Format Technical Committee a felhasználókat meg szeretné szabadítani a különféle gyártók saját formátumainak béklyójától, akik így irodai programcsomagjuk származásától függetlenül használni tudnák a szöveges dokumentumokat, táblázatokat és egyéb állományokat.
Az új bizottság maga mögött tudhatja a Sun, a Word Perfectet készítő Corel, több tartalomkezelő cég, illetve az OpenOffice.org támogatását – az utóbbi máris közös alapnak ajánlotta XML alapú fájlformátumait. Természetesen hiába fognak össze a kicsik, ha az irodai programok piacát uraló Microsoftot nem tudják megnyerni maguknak. A Microsoft ugyan tagja az OASIS-nak, ám az új bizottságban – noha elismeri annak munkáját – egyelőre nem képviselteti magát. Az óriáscégnek nyilvánvalóan nem érdeke feladni saját zárt formátumait, és az Office 11 formátumai ettől függetlenül is erősen XML alapúak lesznek; továbbá a Microsoft már támogatja a World Wide Web Consortium által kifejlesztett XSD 1.0 szabványt. Ha tehát az OASIS csapata netán a W3C ajánlásai mentén dolgozna, a MS Office-sorozat következő tagját is különösebb erőfeszítés nélkül beállíthatná a sorba.
➲ http://www.oasis-open.org

Bővülő iPaQ-kínálat

A világ egyik legnépszerűbb kéziszámítógép-családja, az iPaQ Pocket PC termékvonal két újabb taggal bővült.
A h1910-es készülék elsősorban kedvező, 70 ezer forint körüli árával és kis méretével hívhatja fel magára a figyelmet, míg a h5450-es típus a nagyobb igényű felhasználóik kívánságait hivatott kielégíteni.

A h1910-es típusba 64 MB memória került, SD-kártyafoglalattal, illetve színes kijelzővel rendelkezik. Belsejében egy 200 MHz-es Intel X-Scale processzor lapul, operációs rendszere gyári állapotban Windows for Pocket PC 2002.
A h5450-es készülékbe minden földi jót belezsúfoltak, képes Bluetooth és IEEE 802.11-es kapcsolatot létesíteni, 400 MHz-es processzort kapott – legérdekesebb jellemzője azonban beépített ujjlenyomat-olvasója. Segítségével, illetve az ujjlenyomatunk mellett használt jelszavas vagy PIN-kódos védelemmel gépen tárolt adatainkat az eddigieknél kicsit nagyobb biztonságban tudhatjuk.
Ára 280 ezer forint körül alakul majd.
➲ http://www.hp.com/go/ipaqsolutions

RS-232–Bluetooth barátság

A 3e Technologies International az első RS-232–Bluetooth átalakítót mutatta be, amely irodai és ipari célokra egyaránt használható.
Az átalakító talán nem is az irodai dolgozók számára jelenthet nagy könnyebbséget, hanem azoknak a műszaki szakembereknek, akik különféle tesztelő- és mérőműszerekhez, adatrögzítőkhöz még mindig a jó öreg RS-232-es kapukat használják – természetesen a jó öreg kábelekkel kiegészítve. Az átalakító használatához nincs szükség illesztőprogramra, Bluetooth 1.1 szabvány szerint működik, hatótávolsága 30 méter, adatátviteli sebessége 723 kbit/s. Tápellátását az RS-232 aljzat RTS és DTR lábairól veszi, illetve saját akkumulátorral is rendelkezik. A cég hamarosan egy RS-485-ös változatot is megjelentet.

➲ http://www.3eti.com

Egy telefon, két kamera

A japán NTT DoCoMo egyik új mobiltelefonja két digitális kamerát is tartalmaz. Azok a szerencsések, akik más digitális kamerával egybeépített telefonnal rendelkeznek, bizonyára súlyos gondként élik meg, hogy ha önmagukat próbálják lefényképezni, akkor a kijelző az átellenes oldalra kerül, és nem látják, hogyan fog sikerülni a kép. Ezt a tűrhetetlen és kényelmetlen állapotot szüntették meg a Matsushita mérnökei, akik az így is csupán 18,8 mm vastagságú telefon méreteivel kivívták a versenytársak elismerését.

A készülékre két darab 65 000 szín megjelenítésére képes TFT kijelző került, az egyik 2,1, a másik 1,1"-os. Kamerái CCD érzékelőkkel működnek, és 110 ezer képpontot tudnak rögzíteni.

Sony: másolunk is, meg nem is

A Sony Music Entertainment „LabelGate CD" név alatt olyan új másolásvédelmi eljárást mutatott be, amely – a korábbi módszerekkel ellentétben – lehetővé teszi a zenei CD-k tartalmának számítógépre történő másolását. A cég felmérései kimutatták, hogy a vásárlók több mint 60 százaléka vagy csak számítógépen, vagy azon is hallgat zenét – ha viszont különféle módszerekkel ebben megakadályozzák, jó eséllyel nem veszi meg az őt érdeklő korongot.
Az idén tavasszal bevezetett Copy Control CD (CCCD) módszer egyáltalán nem teszi lehetővé a zenék másolását, a LabelGate CD viszont igen – természetesen némi megkötéssel. A lemezre egy külön, kódolt sáv kerül, amit csak az Internetről beszerzett kulcs segítségével tud visszafejteni a tulajdonos. Amikor át szeretné másolni a tartalmat számítógépének merevlemezére, a központi kiszolgálótól megkapja az ehhez szükséges kulcsot, és ha első alkalommal végzi el a műveletet, fizetnie sem kell. A kulcsot az egyedi azonosítóknak köszönhetően más lemezhez nem lehet felhasználni. A második másolástól kezdődően viszont fizetőssé válik az engedélyezés, a díjat zeneszámonként róják ki a vásárlóra.

Medgyesi Zoltán (mz@rettesoft.hu)
A BMGE 24 éves informatika szakos hallgatója. Szabadidejét legszívesebben a barátnőjével tölti. Szeret autózni és bográcsban főzni. A Linuxot hat éve ismeri, de még nem volt lelkiereje, hogy áttérjen rá. A Linuxvilág hírszerkesztője.

Cégvilág

Újrafeldolgozási programot indított a HP

A HP Magyarország Kft. anyavállalata a „Return & Recycling" környezetvédelmi program részeként az idén hazánkban is bevezette a kiürült festékkazetták begyűjtésére szolgáló HP Take Back programot. A program pontgyűjtő akcióként működik, a kiürült festékkazettákért kapott pontok három kategória termékeire, akár HP-kellékanyagokra is beválthatók. A program során begyűjtött kellékanyagokat környezetre veszélyes részeiktől megszabadítják, és arra alkalmas összetevőiket alapanyagként forgatják vissza a termelésbe.
➲ http://www.hp.hu/takeback

Tévéadás mobilon

Magyarország első mobiltelefonos tévéadását láthatták a szerencsések november 24-én. A Pannon GSM megfelelő felszereltséggel rendelkező előfizetői a TV2 Tények és Napló című műsornak GSM-re átültetett változatát nézhették meg. A továbbiakban esténként a Tények teljes anyaga elérhető lesz, a böngészés segítése érdekében hírekre bontva. A közvetítéshez a világviszonylatban is újdonságnak számító mobil videofolyam (mobile videostreaming) megoldást használták.
Az adások megtekintéséhez megfelelő felbontású kijelző, internet-hozzáférés és nagysebességű, HSCSD adatátviteli képességgel rendelkező készülékre van szükség – a ma kapható telefonok közül a Nokia 9210i, a Nokia 9210 és a Nokia 7650 felel meg ezeknek a feltételeknek.
➲ http://www.korridor.hu/mobilvideo

Tízéves az Ericsson Galéria

Az Ericsson harmadik évezredet idéző épületében, a III. kerületi Laborc utca 1. szám alatt található a cég kortárs képzőművészeti galériája, amely idén decemberben ünnepli tízéves fennállását. A high-tech és skandináv hangulatú környezetben a jubileum alkalmával *Ráday Mihály* „Csend – élet" című tárlatát mutatják be. Az Ericsson Képzőművészeti Galéria 1992 decembere óta ad otthont kiállításoknak, a mostanit 2002. december 3 és 2003. január 7 között tekinthetik meg az érdeklődők, munkanapokon 8.00-tól 17.00 óráig.
➲ http://www.ericsson.hu

Sun Fire Link

A Sun Microsystems bemutatta Sun Fire Link nevű, nyolc felsőkategóriás Sun-rendszert támogató, belső optikai kapcsolati rendszerét. A korábbiaknál egy nagyságrenddel nagyobb sávszélességű (4,8 gigabájt/másodperc rövid időre és 2,8 gigabájt/másodperc tartósan), alacsony késleltetésű (kevesebb mint 4 mikroszekundum) belső kapcsolati rendszer segítségével több mint egy teraflop csúcsteljesítményű, akár 800 processzorból álló fürtözött rendszer építhető ki. A Sun az új Fire Link megoldásra épített szuperszámítógépeket új összeállításokban, Galaxy néven fogja forgalmazni oktatási, kutatási és kormányzati intézmények számára.
A kapcsolórendszer a legszigorúbb rendelkezésre állási követelményeknek is megfelel, a kapcsolatok összevonhatók (legfeljebb négy Sun Fire Link csatorna), redundanciája pedig háromszoros.

WLAN szolgáltatás Ferihegyen

A Westel a hazai mobilszolgáltatók közül elsőként Magyarországon a Budapest Airport Rt.-vel közösen kísérleti vezeték nélküli helyi hálózatelérés (WLAN) szolgáltatást nyújt az arra alkalmas készülékkel és kártyával rendelkezőknek. Az üzleti felhasználók számára bevezetett, nagysebességű, vezeték nélküli internet-hozzáférést kínáló szolgáltatás három hónapig díjmentesen érhető el. A Westel kísérleti WLAN szolgáltatását a felhasználók egyelőre csak a Budapesti Ferihegyi Repülőtér 2B terminálján vehetik igénybe, a későbbiekben viszont számos forgalmas ponton (például bevásárlóközpontokban, konferenciaközpontokban) is elérhető lesz.

Megújul a Magyar Rádió honlapja

„A Magyar Rádió az Interneten is meg kíván felelni közszolgálati küldetésének" – olvasható a rádió megújult honlapján. Ugyan a végleges változat még nem készült el, az élőben elhangzó adások egyre nagyobb része kerül fel RealAudio formátumban a honlapra, illetve a különféle kategóriákba – Itthon, Külhon, Gazdaság, Sport, Kultúra, Tudomány, Internet – szerveződő híreket is folyamatosan frissítik. A műsorok egy részének eddig is volt saját oldala, ezek fokozatosan egységes megjelenést kapnak, és közös tárhelyre, közös adatbázisba kerülnek. Érdekesség a Kincsestár, amelynek Hangos Pantheon című rovatába az elmúlt évtizedek jeles személyiségeinek– színészeknek, sportolók, költők – felvételei kerülnek be.
➲ http://www.radio.hu

Rátz Tanár Úr díj hat tanárnak

Immár második alkalommal vehette át ünnepélyes keretek között a Rátz Tanár Úr Életműdíjat hat középiskolai tanár pályafutása során nyújtott kiemelkedő teljesítményéért. A Graphisoft R&D Rt., az Ericsson Magyarország Kft., valamint a Richter Gedeon Rt. által létrehozott Alapítvány a Magyar Természettudományos Oktatásért kuratóriuma a díjat évente ítéli oda, összesen hatmillió forint értékben. Az Alapítvány díjazottjai azok a középiskolai tanárok, akik az alapítók tevékenységi köréhez szorosan kapcsolódó magyarországi matematika-, fizika- és kémiaoktatásban kimagasló szerepet töltenek be a tantárgyak népszerűsítésében és a tehetséggondozásban.
Az idei díjazottak:
Matematika: *Reiman István* és *Pálmay Lóránt*
Fizika: *Dr. Kopcsa József* és *Nagy Márton*
Kémia: *Szabó Lászlóné Hoffmann Ilona* és *Dr. Harka Katalin*

Medgyesi Zoltán (mz@rettesoft.hu)
A BMGE 24 éves informatika szakos hallgatója. Szabadidejét legszívesebben a barátnőjével tölti. Szeret autózni és bográcsban főzni. A Linuxot hat éve ismeri, de még nem volt lelkiereje, hogy áttárjen rá. A Linuxvilág hírszerkesztője.

A Szabadon projekt

2002. november 15-én a szabad programok felhasználóit képviselő három honi szervezet széles körű kampányt indított.

A Szabad Szoftverért Alapítvány (Free Software Fondation) magyarországi képviselői, azaz az FSF.hu alapítvány, a BSD egyesület, valamint az LME (Linux-felhasználók Magyarországi Egyesülete) közös kampányt indított azzal a céllal, hogy felhívják a számítástechnika iránt érdeklődők figyelmét arra, hogy az amúgy is drága gépeket nem feltétlenül kell még drágább programokkal felvértezni, ha az utóbbiaknak van nyílt forráskódú, szabadon felhasználható változata is.

„Az üzleti programgyártók szerzői jogait védő BSA célja az, hogy az európai uniós csatlakozás időpontjára az elmúlt év végi 48 százalékról az ottani átlagra, 30–32 százalékra csökkenjen a törvénytelenül felhasznált programok aránya Magyarországon" – hangsúlyozta *Sebők Erzsébet*, az egyesület főtitkára az informatika és távközlési bizottság előtt.

A fenti cél elérésében kíván szerepet vállalni a három hazai szervezet, természetesen olyan eszközök igénybevételével, amelyeket erkölcsi érzékük megenged. Véleményük szerint egy demokratikus világban nem a megtorlás és az elrettentés fegyverével kell harcolni a törvénytelen programhasználat ellen, sokkal inkább a meggyőzés eszközével.

Ezért a kampány kezdetétől számított hozzávetőlegesen három hónapon keresztül fővárosunkra összpontosítva igyekeznek minden olyan lehetőséget megragadni, amelynek révén igéjük hirdetése nyitott fülekre találhat. Jelenleg öt budapesti metróaluljáróban találkozhatunk a képen látható plakáttal, melynek célja, hogy vonzó módon válaszoljon a BSA „bilincses" kampányára.

„Számunkra fontos, hogy a végfelhasználók észrevegyék: létezik más lehetőség is. Ezért ez a kampány nem tartalmaz kellemetlen képzettársításra okot adó jelképeket. Mi lényegében ugyanazért a célért küzdünk, mint a BSA, csakhogy mi megválogatjuk az eszközeinket. Meggyőződésünk, hogy fenyegetés és terror nélkül is lehet eredményeket elérni. Elég értetlenül állunk az események előtt, hiszen nem hittük volna, hogy hazánkban a nyilvános terrort óriásplakátokon lehet népszerűsíteni, miközben az európai csatlakozásról beszélünk. Tehát mi arra akarunk példát szolgáltatni, hogy a törvénytelen programhasználat ellen lehet oly módon is küzdeni, hogy az a jó ízlés határain belül maradjon."
– mondta *Varga Csaba Sándor*, az FSF.hu alapítvány kuratóriumának elnöke.

Magosány Árpád-tól, az LME elnökségi tagjától, aki jelentős részt vállalt a szervezésben, megtudhattuk, hogy a jövőben az eddig kihelyezésre került 70×100 cm-es plakátok B5-ös, azaz szórólap méretben is legyártásra kerülnek. Ezeket a kampányban résztvevő cégek partnereiken és azok üzlethálózatain keresztül fogják a felhasználókhoz eljuttatni. Terveznek még tájékoztató, ismeretterjesztő előadásokat is, ahol a szabadon

felhasználható programokban rejlő lehetőségeket kívánják bemutatni.

A kampány résztvevői

LME ➲ http://www.lme.hu

Magyar BSD Egyesület ➲ http://www.bsd.hu

FSF.hu ➲ http://www.fsf.hu

FSN.hu ➲ http://www.fsn.hu

UHU-Linux Kft. ➲ http://www.uhulinux.hu

SuSE Linux AG magyarországi iroda
➲ http://www.suselinux.hu

Mission Critical Linux Kft. ➲ http://www.mclx.hu

Software Station ➲ http://www.swsbooks.hu

Linux Support Center Kft. ➲ http://www.lsc.hu

A kampány hivatalos weboldala
➲ http://www.lme.hu/szabadon/

Gibizer Tibor (gibzo@linuxmania.hu)
Újságíró, immár hét éve a Linux elkötelezett híve. Imádja a kutyákat, a kerékpározást és az autós csavargást.

Küldetés igény szerint

Háztűznézőben jártunk a hazai linuxos piac egyik meghatározó cégénél, a Mission Critical Linuxnál. ***Bodnár Csabát***, az ügyvezető igazgatót a cég múltjáról, jelenéről és jövőjéről kérdeztük.

Csontos Gyula: Hogyan jött létre a magyarországi Mission Critical Linux?

Bodnár Csaba: Egy barátom, *Moiz Kohari* 1999-ben – a Compaq céget otthagyva – alapította meg a Mission Critical Linux céget az Egyesült Államokban. Főként nagyvállalati termékekkel és ezek támogatásával foglalkoznak – ilyenek például a nagy számítási teljesítménynyel rendelkező telepek, illetve a magas rendelkezésreállást (High Availibility – HA) biztosító rendszerek.

Az ötlet, hogy saját linuxos céget hozzunk létre, 1999-ben kezdett érlelődni – én személy szerint már többéves unixos és linuxos tapasztalattal rendelkeztem –, az elképzelést később tett követte, és a 2000-es CeBiten az amerikai vállalattal megszületett a megállapodás, így létrejöhetett a magyarországi leányvállalat. 2000 májusában jelentünk meg először úgy, mint a magyarországi Mission Critical Linux, egy kiállításon. 2000 júniusában pedig a tényleges cégbejegyzés is megtörtént.

***Cs. Gy.**: Hogyan alakult ezek után a cég élete?*

B. Cs.: Ügyfélkörünk a megalakulástól kezdődően egyfolytában bővült, így a 2002-es évben már nyugodtan elmondhatjuk: a cég nyereségessé vált. Jelenleg 8–10 főt foglalkoztatunk folyamatosan.

***Cs. Gy.**: Mik a fő termékvonalaitok?*

B. Cs.: Rólunk az terjedt el, hogy telepekkel és nagyvállalati megoldásokkal foglalkozunk. Ez az átlagfelhasználó számára szükségtelen, szolgáltatásaink azonban nem csak nagyvállalati környezetben használhatók. Szívesen megoldunk bármilyen Linuxszal kapcsolatos feladatot, legyen az irodai fájlkiszolgáló, levélkiszolgáló telepítése, karbantartása vagy bármi, amit az ügyfelek kitalálnak.

Szolgáltatásaink széles körűek, így minden igényt ki tudunk elégíteni. Ha csoportosítani szeretném őket, akkor négy különféle termékvonalat említhetek.

Elsőként a „telepítési szolgáltatás”-t emelném ki, ami akár a tűzfal, a webkiszolgáló, a levélkiszolgáló, az állomány- és nyomtatókiszolgáló, az adatbázis-kiszolgáló, az alkalmazáskiszolgáló, a magas rendelkezésreállású rendszerek, a tudományos célú telepek, az útvonalválasztás, a sávszélesség-megosztás és a VPN-megoldások telepítését is magában foglalja, ezen kívül annak személyre szabott beállítását, üzembe helyezését és a szükséges operátori oktatást is.

A második csoport a „technikai támogatás” – ez a szolgáltatás szintén személyre szabottan vehető igénybe, így költség- és energiatakarékos. Ügyfeleink főként a heti 7×24-es (a hét minden napján mindig elérhető) és a napi 5×8-as (munkanapokon munkaidőben elérhető) rendelkezésreállás-szolgáltatást kérik, természetesen ezektől eltérő szolgáltatásformákat is kidolgozhatunk. A rendszerfelügyeletet azok az ügyfeleink veszik igénybe, akik nem rendelkeznek a megfelelő szakembergárdával. Az ügyfelek a fentebb említett két szolgáltatást együtt szeretik igénybe venni, ilyenkor természetesen a kedvezmények is nagyobbak.

Kínálatunk harmadik nagy csoportját a telepmegoldások (cluster technology) adják. Megalakulásunk óta foglalkozunk ezzel a területtel, bármilyen telepeket ki tudunk építeni, ami a megrendelő igényeinek megfelel. Az egyik ilyen rendszer a nagy rendelkezésreállást nyújtó telep, ennek az összeállításához szabványos PC-alkatrészeket és saját fejlesztésű Kimberlite/Convolo HA programunkat használjuk. A kettő közül a Kimberlite ingyenesen letölthető. A tudományos-technikai célú telepeket, mint amilyen a Beowulf-telep is, ügyfeleink hatalmas számítási teljesítményükért választják szívesen. Segítségükkel „könnyedén” elvégezhető szinte bármilyen nagyságú számítás (természetesen ez függ a telepbe épített gépek számától és teljesítményétől). Terheléselosztó telepeket elsősorban a webfarmoknál és az elektronikus kereskedelemben használnak. Az SSI típusú telep több számítógépet foglal magába, de mind a felhasználók, mind a rajta futó programok egyetlen számítógépnek látják őket.

A negyedik csoport a különféle Linux-kompatibilis hálózati eszközök forgalmazása. Az MCLX a közép- és kelet-európai terjesztője két éve a Western Telematic Inc. (USA) termékeinek. Ezek között a termékek között távoli újraindító kapcsolók és távoli hálózatkezelési eszközök találhatók, amelyek változatos kiépítésben érhetőek el.

***Cs. Gy.**: Kik és mire használnak telepeket jelenleg Magyarországon*?

B. Cs.: A nagy számításigényű feladatokat igénylő helyeken, a tudományos kutatásban, esetleg az autók töréstesztjét is ilyeneken futtatják. Jelenleg is folyamatban van egy ilyen telep építése az ELTE Elméleti Kémiai Tanszéke számára. Ez a telep teljesen egyszerű és olcsó PC-alkatrészekből fog állni, mindegyikben két-két AMD Athlon processzor dolgozik majd, és húsz csomópont lesz egyetlen Beowulf-rendszerré összeállítva. A telep egyik feladata az elektronburok kirajzolása lesz.

***Cs. Gy.**: Állami cégekkel van-e kialakult kapcsolatotok?*

B. Cs.: A kapcsolatok ebből a körből sajnos egyelőre hiányoznak, ezen a téren is igyekszünk javítani. Jelenleg a Bull Magyarország kínálja termékeinket a közbeszerzési projektekhez.

***Cs. Gy.**: Mik a terveitek a jövőre nézve?*

B. Cs.: Mint minden cégnek, természetesen nekünk is a folyamatos fejlődés fenntartása a célunk – mindenképpen erősíteni szeretnénk a rendszerépítői vonalat.

***Cs. Gy.**: Köszönöm a beszélgetést!*

A cég honlapja ➲ http://www.missioncritical.hu

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

UHU-Linux-pályázatok

A hazai Linux-változatnak szánt UHU-Linux fejlesztője az UHU-Linux Kft. Jelenleg a rendszernek még csak az RC1-es próbaváltozata érhető el, de máris ígéretes fejlesztés. A társaság vezetőjét, *Körmendi Andrást* kérdeztem az UHU helyzetéről.

Gibizer Tibor: *Az elmúlt hónapokban szinte folyamatosan érkeztek az izgalmasabbnál izgalmasabb hírek az UHU háza tájáról. Sok a kérdés és sok a találgatás az új rendszerrel kapcsolatban. Milyen friss híreket tudnál megosztani olvasóinkkal?*

K. A.: Mint azt már tudjátok, jelen voltunk az LME IV. GNU/Linux Szakmai Konferenciáján, és ott bejelentettük, hogy készül az UHU-Linux Tűzfal és Kiszolgáló változata, ami természetesen szintén szabadon felhasználható lesz. Örömmel mondhatom, hogy a fejlesztés nagyon szépen halad. Közel állunk az első próbaváltozat közzétételéhez, de bevallom, igyekszünk egy kicsit késleltetni a kiadást, hogy már az első változatok is a lehető legjobbak legyenek. Emellett nagyon fontos, hogy a terméket a hazai felhasználók igényeinek megfelelőre kívánjuk kialakítani, hiszen kimondottan nekünk, magyaroknak készül. Mire ez az írás megjelenik, feltehetőleg már mindkét változatból a kezünkben lesz egy-egy használható próbapéldány.

G. T.: *Már hosszabb ideje elérhető az UHU Linux RC1-es próbaváltozata. Van-e már pontos hír az RC2-es kiadásról? Ezt talán többen várják, mint az előző fejlesztéseket.*

K. A.: Azt sajnos nem tudom, hogy mennyien várják az UHU-Linux 1.0 RC2-s változatának megjelenését, de ha csak annyian fogják használni, mint az RC1-et, nyugodtak lehetünk, hiszen a számok egyértelműen azt mutatják, hogy jó irányba haladunk. Érdekességként megemlíthetem, hogy az UHU-Linux asztali változatát szinte teljesen átfésültük, az összes csomag frissült benne. De erről majd akkor beszélgessünk részletesebben, ha már készen leszünk vele – ami valójában már nincs is messze. Remélem, a következő hónapban már a magazin oldalain is részletes beszámolót tudtok lehozni az új változatról.

G. T.: *Nagy érdeklődéssel hallgattuk a konferencián, hogy a társaság új pályázatokat ír ki. A hazai linuxos életben egyedülálló pályázati kiírásotokról szeretnélek kérdezni. Mit lehet erről tudni? Mi a háttere a pályázatok kiírásának?*

K. A.: A nyílt forráskódú fejlesztések általában nem haszonelvűek, hanem egyéni alkotóerők összeadódásából és lelkesedésből építkeznek. Ahhoz, hogy a hatékonyságot ugrásszerűen növelhessük, az erőforrásokat egybe kell vonni.

Az UHU-Linux Kft. az általa összefogott fejlesztések mellett figyelmet fordít a fentiekben megfogalmazott célokkal nem szükségszerűen egybeeső, ugyanakkor a fejlesztői, illetve alkalmazási szempontokat figyelembe vevő, társadalmilag hasznos fejlesztések támogatására is.

Ez alatt azt értjük, hogy a társaság közvetlenül összehangolt saját fejlesztőcsapata mellett támogatást kíván nyújtani olyan, nem a céghez tartozó fejlesztői csoportok számára is, amelyek a magyarországi egyéni Linux-használat elterjedése érdekében vagy éppen egy más nagyságrendű felhasználói hozzáférést elősegítő fejlesztésen dolgoznak.

Mit is jelent ez? Lényegében azt, hogy véleményünk szerint ahhoz, hogy a Linux valóban széles körben el tudjon terjedni, támogatni kell azokat a fejlesztőket, programozókat, akik megvalósításra váró ötletekkel bírnak, megvan hozzá a szakmai hozzáértésük, de eddig erőforráshiányok miatt nem tudtak megfelelő munkát fordítani a fejlesztésre. Tudjuk, hogy ma már hihetetlen mennyiségű program áll a felhasználók rendelkezésére, de ezek rendszerint nem hazai fejlesztések, így sok esetben nem is igazán ültethetők át a mi igényeinknek megfelelő formába. Sokat segít a fordítás, de kimondottan a mi – itt természetesen a magyar viszonyokra gondolok – igényeinknek, kívánalmainknak vagy mondhattam volna azt is, néha jogszabályainknak megfelelő programok csak itt születhetnek, azokat fordítások útján nem lehet beszerezni. Számunkra nagyon fontos, hogy összefogjuk a fejlesztőket és támogassuk a munkájukat, hiszen a Linux ereje mindig az összefogásban rejlett. Csak gondolj bele, milyen módszerrel vehetjük fel a versenyt a nagy, illetve a multinacionális cégekkel? Van nekünk erre pénzünk? Szerintem nincs, és sokáig nem is lesz. Akkor mi marad számunkra a járható út? Csak az összefogás. Ha a linuxos emberek végre felébrednek, és tudásukat, tapasztalataikat megosztják a közösséggel, abból csodálatos dolgok fognak születni, sőt születnek is, jó példa erre az MPlayer fejlesztése, vagy Gnome2 fordítócsapatának munkája. A Linux rohan előre. Már nemcsak a környező országok számára egyértelmű, hogy a helyes irány a nyílt forráskód felé vezet, de Kína után már India is nemet mondott a zárt forráskódra – jó, tudom, hogy a tényleges felhasználói tábort ez feltehetően nem igazán fogja befolyásolni, de ha már egy közismerten elmaradott állam is látja a helyes utat, talán majd egyszer mi is erre fordulunk.

G. T.: *Tudom, hogy még nem történt meg a pályaművek elbírálása, de gondolom, belekukkantottál már a beérkezett anyagokba. Így első ránézésre hogyan ítéled meg a pályaműveket?*

K. A.: Amikor az imént azt mondtam, hogy jó úton haladunk, akkor ezt a területet tudatosan nem említettem, hiszen a legnagyobb meglepetés pont itt ért minket. Nem titok, hogy a vártnál is sokkal több anyag érkezett, és véleményem szerint a többség valóban eléri azt a szintet, sőt több esetben meg is haladja, amire számítani mertünk. Sejtettük, hogy sokan fognak jelentkezni, de azt álmunkban sem hittük, hogy ennyien. Pontos számokat inkább nem mondanék, hiszen még nem értük el a leadási határidőt, de ha gondolod, örömmel „pletyizek” egy keveset.

G. T.: *Ezek szerint váratlanul ért benneteket a sok pálya-*

mű. Eredetileg ötmillió forintot különítettetek el erre a célra. Ha lényegesen több támogatásra érdemes pályázat érkezik, milyen elveket követtek a támogatások megítélésekor? Minden arra érdemes kap egy keveset, vagy csak a tényleg kiemelkedők kerülnek díjazásra?

K. A.: A gondok azért vannak, hogy megoldjuk őket. Jó, tudom, ez így közhely, amivel én személy szerint utálok dobálózni, de erre tényleg nem tudok mást mondani. Ötmillió nem kevés pénz, de a jelek szerint legalább húszra lenne szükség, viszont annyi nincs. Ezért feltehetően azokat fogjuk támogatni, akiket bírunk. Semmiképpen nem szeretnénk viszont, hogy a többi pályázat kárba vesszen. Ezért a mostani kiírás tanulságait levonva arra gondoltunk, hogy az UHU-csapat a jövőben teljes erőbedobással azon lesz, hogy minél nagyobb keretet tudjon ezekre a területekre fordítani. Amint sikerült a megfelelő pénzösszeget összeszedni, azonnal megismételjük a mostani kiírást, tehát senki ne csüggedjen, ha most kimaradt vagy csak lemaradt, hiszen igyekszünk mindent megtenni, hogy ez a folyamat ne álljon le.

G. T.: *Bele tudunk kukkantani, hogy a különböző témakörökben milyen anyagok érkeztek?*

K. A.: Természetesen igen, de csak futólag, hiszen addig, amíg a pályázat nincs lezárva, tisztességtelen lenne őket bemutatni. A magazin megjelenésekor már remélhetőleg a kiértékelésen is túl leszünk, remélem, nagy titkokat nem fogok kikotyogni. Fontos megjegyeznem, hogy a kiértékelést egy független szakmai zsűri végzi majd. Lássuk, mire vagy kíváncsi!

G. T.: *Hány pályamű érkezett, és meg lehet-e határozni, hogy túlnyomórészt honnan érkeztek?*

K. A.: Eddig több mint harminc anyagot kaptunk, és ha most azt akartad finoman megfogalmazni, hogy ezek mennyire korlátozódnak a fővárosra, azt tudom mondani, hogy egyáltalán nem, tehát az ország szinte minden szegletéből csodálatos anyagokat kaptunk. Nagyon örülünk, hogy nemcsak ötletszinten élő, hanem futó projektek fejlesztői is jelentkeztek, például az Ispell, a Freddy, vagy az MPlayer fejlesztőitől, szakembereitől.

G. T.: *Valóban erre gondoltam, ugyanis többször tapasztalom, hogy sokan azt hiszik, csak Budapest belvárosára összpontosul a hazai linuxos élet. Ismert alkalmazásokat említettél - készítőik miben kérték a segítségeteket?*

K. A.: Az egyik sokak által ismert alkalmazás weblapjának a készítője el szeretné készíteni az alkalmazás magyar leírását; míg egy másik egy olyan program, ami már többszörösen bizonyított, és amely már most felülmúlja a Microsoft-termékekhez használt változatot. A szerző csupán be szeretné fejezni az eddig végzett munkát, azaz végre eljutni az 1.0-s változathoz. Ezek a programozók már bizonyítottak, és tényleg csak azért van ránk szükségük, hogy hatékonyabban tudják a munkájukat végezni. Mi ebben természetesen örömmel leszünk a partnerük, mint ahogy több esetben már azok is vagyunk. Ne értsd félre, ez nem azt jelenti, hogy ők a pályázati elbírálástól függetlenül máris támogatást kaptak, de gondolom, a szakmai zsűri is értékelni fogja eddigi erőfeszítéseiket, és a hasznosság mérlegelése mellett a lehetőségekhez mérten a támogatás mellett fognak szavazni. Természetesen az is előfordulhat, hogy nem így döntenek, de akkor még mindig ott lesz a lehetőség, hogy a következő körben ismét megpróbálják pályázni, tehát semmi sincs veszve.

G. T.: *Régi meggyőződésem, hogy a Linux népszerűsítését az irodákban és az iskolákban kell kezdeni. Az, aki a munkahelyén Linuxot használ, nagy valószínűséggel otthon is azt fog, illetve aki már az iskolapadban megismeri ezt a remek rendszert, a való világban is ezt akarja majd használni. Ezen a területen is érkeztek használható megoldások?*

K. A.: Szinte minden területre érkezett pályamű, de az irodai alkalmazások, illetve az oktatás témakörét érdemes kiemelni, hiszen ezek a területek valóban nagyon fontosak, és a jelek szerint ezt nem csak mi látjuk így. Igazán remek anyagok érkeztek, amik szerintem szintén érdemesek arra, hogy az UHU-Linux mögéjük álljon. Most mosolyogsz, de tényleg nem tudok mást mondani, hiszen szinte az összes anyag olyan, amit vétek lenne nem megvalósítani. Mondok pár példát: a távoktatás hazánkban elég mostoha helyzetben van. Érkezett egy anyag, ami ezt szeretné bővíteni, és ami kimondottan megnyerte benne a tetszésemet, az az, hogy nem a fellegekben jár, jó érzékkel azt a kérdéskört feszegeti, ami a fejlesztés alapjaihoz szükséges. Jelen esetben a programozáshoz. A tervező a saját tapasztalatai alapján szeretné bevonni az embereket a C-programozás alapjainak megismerésébe. Ez fontos, hiszen annak ellenére, hogy erről a témakörről számtalan mű született már, olyan, ami magyar szerző tollából kimondottan távoktatási céllal íródott volna, ráadásul saját tapasztalatokon alapulva, legjobb tudomásom szerint még nem. Természetesen léteznek fordított anyagok, de ezek rendszerint közel sem annyira „emberiek", mint az olyanok, amelyeket hozzánk hasonló kezdő jegyzett le folyamatosan, ahogy a tudás megszerzésének rögös útján haladt.

Oktatást segítő alkalmazásra is érkezett pályamű. Dióhéjban az a lényege, hogy az iskolák számára (főként az iskolatitkári munkákhoz nagy segítséget nyújtó) alkalmazást készítsen, ami lehetővé teszi a diákok adatainak pontos, naprakész nyilvántartását, kezelését, megkönnyítse a különféle kimutatások elkészítését, egyszerűvé tegye az adatok felvitelét, módosítását, visszakeresését. A programhoz kiegészítő modulok is társulnának, például könyvtári adatbázis, órarend.

Egy ilyen program kifejlesztése – ha kiadod egy cégnek – rémisztő összegbe kerülne. Abban az esetben, ha egy cég felvállalja ennek az elkészítését, de nem a Linux oltalma alatt, tehát nem nyílt forrással, szabadon hozzáférhetően, jogdíjat kell utána fizetni. Nem tudom, jelenleg hány iskola van hazánkban, de ha mindegyik megrendelné ezt a programot, abból az összegből biztosan lehetne készíteni pár UHU-Linuxot – vagy ha a valóság

talaján akarok maradni, akkor azt mondom, inkább vegyen belőle az iskola labdát vagy amit akar. Ezt nem akarják sokan megérteni, hogy az a rengeteg pénz, amit most a jogdíjakra szétszórnak, mennyivel hatékonyabban felhasználható, ha a nyílt forráskódot támogatják. Hosszútávon hihetetlen megtakarítások érhetők el. Annyit még hozzátennék, hogy ez a program lényegében minden rabszolgamunkát le szeretne venni a tanárok válláról, tehát egy igazán széleskörű alkalmazásról beszélhetünk, ami természetesen hálózati munkára is alkalmas. Ennek erényeit most inkább nem ecsetelgetem, hiszen a Linuxvilág olvasói bizonyára tisztában vannak azzal, hogy a központilag kezelt adatbázisok milyen lehetőségeket rejtenek magukban.

Felmerült olyan ötlet is, ami kimondottan az UHU-Linux általános és középiskolák számára készítendő változatának támogatását tűzte ki célul. Ennek érdekében vállalnák olyan programok keresését, adott esetben magyarítását vagy támogatását, használati leírások készítését, melyek az általános és középiskolák számára különösen jól használhatók, és Linuxról lévén szó, gazdaságosak.

Te elég sokat foglalkoztál az oktatás kérdésével, így nem kell külön magyaráznom, hogy mennyire siralmas a helyzet. Vannak gépek, vannak géptermek, de csak kevés. Ha több lenne azonban, akkor sem tudnának mit kezdeni vele, mivel például a történelem vagy biológia tantárgyakhoz – és sorolhatnám – lényegében nincs olyan anyag, amivel a gépeket ténylegesen ki lehetne használni. Már régen nem igaz, hogy a számítógép csak a számítástechnika vagy jelen esetben hazánkban az ámítástechnika oktatására használható. Elenyésző számban léteznek ugyan programok, de azok is rendszerint valamilyen idegen nyelvről lettek átültetve, ráadásul meglehetősen borsos áron kínálják őket. Ismét csak azt tudom mondani, hogy ha a döntéshozók végre belátnák, milyen lehetőségek rejlenek a szabad programokban, viszonylag gyorsan komoly javulást lehetne elérni az oktatás anyagi helyzetében.

Tehát ez a kezdeményezés komoly hangsúlyt fektetne a műszaki szakközépiskolákra, hiszen ezt a területet sokkal hamarabb meghódíthatná. Távlati céljuk egy olyan információs kiszolgáló létrehozása, mely adat- és programforrás lehetne más iskolák számára, ahol UHU-Linuxot használnak.

Tényleg csak a példa kedvéért kiemelnék még egy pályaművet, aminek a fontosságával, gondolom, mindenki egyetért majd. Sajnos sok gyermek küszködik tanulási gondokkal, ezért fontos lenne egy az ő nevelésüket segítő rendszer kifejlesztése. A tanulási zavar „állapot”, ami olyan gyermekeknél fordul elő, akik átlagos vagy átlagon felüli intelligenciával rendelkeznek és mindenekelőtt az jellemző rájuk, hogy elmaradásuk a tanulás egy vagy több részterületén életkorukhoz képest jelentős. Ilyen területek a beszéd, a beszédfelismerés, a figyelem, a gondolkodás, a magatartás, illetve a mozgás. A tanulási zavarral küzdő gyermekek esetében, mivel egyedi, az átlagostól eltérő adottságokkal bírnak, egy-egy tanuláshoz szükséges tevékenység gyengébb működésével vagy sokszor hiányával találkozhatunk. Ennek kezeléséhez különleges pedagógiai módszerekre van szükség. Nagyon fontos, hogy ez nem betegség és nem is fogyatékosság. Egyes vizsgálatok szerint ma minden tizedik gyermek valamilyen tanulási- vagy teljesítményzavarral küzd. Nem szabad elfelejteni, hogy ezek a gyermekek, ha megfelelő oktatásban részesülnek, teljesen egyenrangú építőtagjai lehetnek társadalmunknak.

Nem akarom az olvasókat a részletekkel fárasztani, ezért csak annyit tennék még hozzá, a pályázó célja, hogy abban segítse a szakembereket hogy, minél hamarabb felfedezzék az esetlegesen tanulási gondokkal küzdő gyermekeket, valamit a tanulóknak játékosan szórakoztató segítséget nyújtson a tanulásban, miközben igyekszik megkönnyíteni a nevelő munkáját.

Ez a téma természetesen nem kapcsolódik szervesen az UHU-Linuxhoz, de nem is baj, hiszen egy olyan fontos területen próbál segíteni, aminek a fejlesztése mindannyiunk érdeke.

Még számtalan példát fel tudnék hozni ebben a témakörben, de gondolom, lassan már külön melléklet méretűvé fog válni a cikked.

G. T.: *A terjedelem valóban korlátozott, de olyan öröm volt látni a lelkesedést, hogy nem volt szívem megszakítani a beszámolódat.*

K. A.: Köszönöm, pedig eszembe jutott még egy nagyon hasznos projekt, egy többkulcsos növényhatározó, de ebben most már tényleg ne mélyedjünk el.

G. T.: *Az oktatás fontosságát botorság lenne megkérdőjelezni, de mint tudjuk, a „belbecs” mellett fontos szerepet kap a „külcsín” sem. Érkezett anyag grafikai megvalósításra is?*

K. A.: Abban a szerencsés helyzetben vagyok, hogy most bármire rákérdezhetsz, mert semmire sem fogom azt mondani, hogy nem. Természetesen érkezett. Megemlítenék egy számomra nagyon rokonszenveset, mely egységesített témákat készítene a GDM-hez, KDM-hez, Gnome-hoz, KDE-hez, Gimphez, Mozillához és sorolhatnám. Itt a grafikai munka kiterjedne a háttérképekre, az ikonokra, és lényegében az egész kinézetre, ami valóban nem elhanyagolható.

G. T.: *Akkor minden túlzás nélkül kijelenthetjük, hogy a pályázat sikeres volt?*

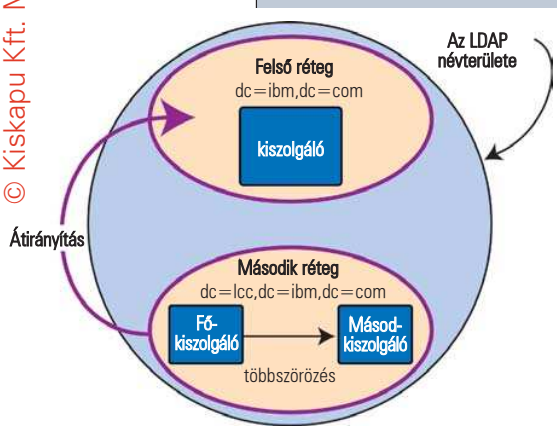
K. A.: Feltétlenül. Nem győzöm hangsúlyozni, hogy messze felülmúlta a várakozásunkat.

G. T.: *Köszönöm a beszélgetést. Remélem, hamarosan részletesen is bemutathatjuk a nyertes pályázatokat.*

Gibizer Tibor (gibzo@linuxmania.hu)
Újságíró, immár hét éve a Linux elkötelezett híve. Imádja a kutyákat, a kerékpározást és az autós csavargást.

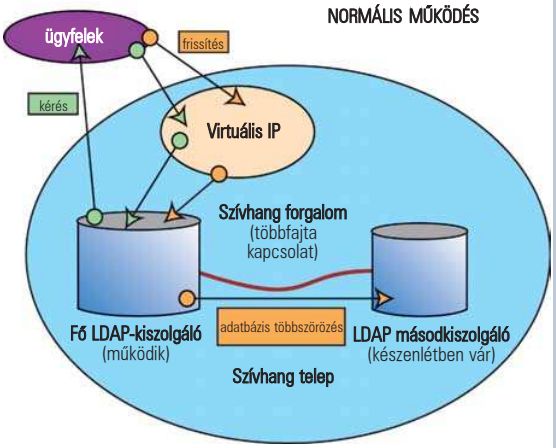
Magas rendelkezésre állású LDAP

Vegyünk egy LDAP-kiszolgálót, majd adjunk hozzá pár evőkanálnyi fejlesztést a Linux-HA projektből, és máris kész egy magas rendelkezésre állású azonosítótelep.



1. ábra
Az LDAP névterenként egyetlen mestert engedélyez

Minél több programot és szolgáltatást használ egy vállalat, annál nagyobb segítséget jelent a rendszergazdák számára a központosított azonosítás. Ráadásul egy központi adatbázis használatával a biztonság is nagymértékben növelhető. Gondot jelenthet viszont, ha minden hitelesítést egyetlen gépre helyezünk. Ebben az esetben egyetlen gép meghibásodása esetén az egész vállalat leállhat. Itt kerülnek a képbe a hibatűrő telepek. Ebben a cikkben azt tárgyaljuk, hogyan hozhatunk létre felhasználók azonosítására szolgáló megbízható kiszolgálótelepet. Példánkban egy LDAP-kiszolgáló (Lightweight Directory Access Protocol, vagyis pehelysúlyú könyvtárhozzáférési/címtárhozzáférési protokoll) látja el a központosított felhasználóazonosítást, melyre a különböző alkalmazások feliratkozhatnak. A szolgáltatást két gépre bízzuk, ehhez a Linux-HA kezdeményezés (➲ http://www.linux-ha.org) heartbeat csomagját használjuk fel.



2. ábra
A slurpd küldi ki az LDAP adatbázis változásait a mesterről a segédre

Az LDAP háttere

Munkánk során az OpenLDAP kiszolgálót használjuk fel, mely sok különféle Linux-terjesztésben megtalálható. A Red Hat Linux 7.1-es változata a 2.0.9-es OpenLDAP-t tartalmazza. Ezen cikk írásának idején az alkalmazás legfrissebb változata a 2.0.11-es. Az OpenLDAP Alapítvány azért jött létre, hogy megalkossanak egy nagy teherbírású, kereskedelmi minőségű, nyílt forráskódú LDAP-csomagot, amely különböző LDAP-alkalmazásokat, fejlesztőeszközöket tartalmaz. Az OpenLDAP 1.0-s változata 1998 augusztusában jelent meg, jelenleg pedig már a 2.0-s sorozatnál tartanak, melyet 2000 augusztusában kezdtek el, és ami már támogatja az LDAPv3-at is.

Az LDAP, mint bármely más jól működő hálózati szolgáltatás, képes egyszerre több kiszolgálón futni. Ezt adatbázis-többszörözéssel (replication) és átirányítással (referral) valósítja meg. Az átirányítások segítségével válik lehetségessé, hogy az LDAP-névterület több kiszolgáló között legyen szétosztva, és hogy több LDAP-kiszolgálót egyetlen többszintű rendszerbe szervezzük. Az LDAP egy névtérhez (namespace) csak egyetlen főkiszolgálót enged hozzárendelni (*1. ábra*).

A többszörözést az OpenLDAP többszörözője végzi, a `slurpd`, mely meghatározott időközönként felébred, és ellenőrzi a főkiszolgálón a naplófájlt, hogy megtudja, történt-e változás. A módosítások ezt követően a másodkiszolgálókra is átkerülnek. Az olvasási kéréseket bármely kiszolgáló megválaszolhatja, frissítéseket azonban csak a főkiszolgálón lehet elvégezni. Ha egy alárendelt kiszolgálón próbálunk meg frissíteni, akkor egy olyan átirányítást kapunk vissza, amelyből megtudhatjuk a főkiszolgáló címét. Ezt követően az ügyfél feladata, hogy megtalálja a főkiszolgálót és elvégezze rajta a frissítést.

Az OpenLDAP-csomagban nincs terhelésmegosztásra alkalmas eszköz, erre a célra valamilyen IP-szintű kérelemszóró programot használhatunk, mint amilyen például a `balance`.

Feladatunk tehát egy hibatűrő rendszer kialakítása. Ehhez két kiszolgálót kapcsoltunk össze. Használhattunk volna osztott tárolóegységet is a két gép között, így az adatokból csak egyetlen másolatot kellene kezelni, mi azonban a „megosztott-semmi" elméletet követtük, ezért minden kiszolgáló saját tárolóeszközzel rendelkezik. Az LDAP-adatbázisok alapesetben kisméretűek, és csak ritkán frissülnek (tanács: ha *tényleg* nagy LDAP-adatbázissal kell dolgoznod, gondold végig, hogy névteredet átirányításokkal feloszthatod-e kisebb egységekre).

A megosztott-semmi elrendezés némi odafigyelést igényel, mikor egy korábban leállított csomópontot indítunk újra: minden időközben történt változást hozzá kell adni az adatbázishoz, még mielőtt a kiesett csomópontot újra elindítjuk. Erre az esetre láthatsz majd példát a későbbiekben.

A géptelep kiépítése

Mielőtt belekezdenénk, oszlassunk el egy félreértést. A legtöbb HA (high availability – magas rendelkezésre állású) géptelep kiépítésének alapja egy üzemelésfigyelő („létfenntartó") szolgáltatás, amelyet szívhangnak (heartbeat) hívunk. Ezt a szívhangot használjuk, hogy a géptelepben található csomópontok egészségi állapotát ellenőrizzük. A Linux-HA (➲ http://www.linux-ha.org) kínál alkalmazást erre a célra, melyet – mily meglepő – Hearbeatnek hívnak. A hasonló elnevezések itt olykor félreértéshez vezethetnek, ezért a cikkben a Linux-HA csomagjára Heartbeatként hivatkozunk, az általános fogalomra pedig szívhangként.

A Linux-HA projekt 1998-ban a Linux-HA HOGYAN-ból

kifejlődve indult útjára, amit *Harald Milz* készített. A munkát jelenleg *Alan Robertso* irányítja, de a csapatnak még megannyi résztvevője van. A Hearbeat 0.4.9-es változata 2001 elején jelent meg. A Heartbeat a csomópontok állapotát vizsgálja valamilyen hálózati közvetítőn keresztül, mely leggyakrabban soros vonal vagy hálózat. Hasznos több egymástól független közvetítővel („szívhangvezetékkel") is ellátni a rendszert. A csomópontok mindegyikén fut egy Heartbeat nevű program. A főprogram olvasó és író alprogramot hoz létre minden egyes közvetítőhöz, valamint elindít egy ellenőrző alprogramot is. Ha egy csomópont kiesik, akkor a Heartbeat az indító és leállító parancsfájlokat lefuttatja a megfelelő csomóponton. A programterveknek megfelelően ezek a parancsfájlok ugyanazokat a kapcsolókat használják, mint a rendszerindító parancsfájlok (amelyek alapesetben a */etc/init.d/* könyvtárban találhatók). A mintafájlokban a fájlrendszer, a webkiszolgáló és az IP-beállítások kezelésére találunk példát.

Két LDAP-kiszolgálóval sokféle különböző kiépítés használható. Először is létrehozhatunk egy úgynevezett „készenléti" rendszert, ahol a mester (a főcsomópont) egy látszólagos IP-cím, ami egy futó kiszolgálóra mutat. Ilyenkor a másodkiszolgáló csak várakozik. Ha a főkiszolgálón hiba lép fel, a virtuális IP és a szolgáltatás átköltözik a várakozó („cold", „hideg") csomópontra. Ez egy viszonylag egyszerű kiépítés, de az adatok elosztása, és egyeztetése gondot okozhat a két csomópont között.

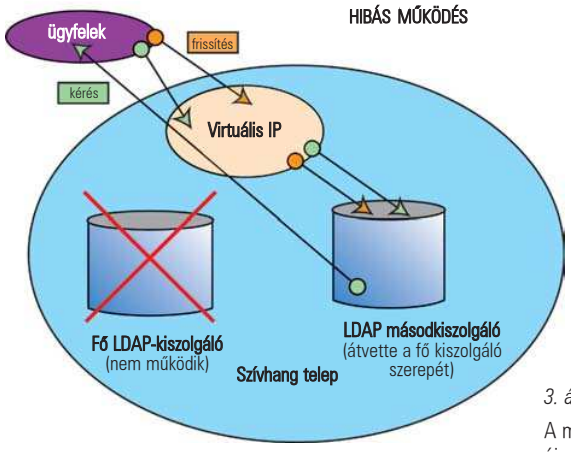
A másik lehetőség, hogy mindkét kiszolgáló „él". Ebben az esetben a főcsomópont futtatja az LDAP-kiszolgáló mesterpéldányát, míg a másodcsomópont csak egy segédpéldányt (egy „szolgát"). A mesteren elvégzett frissítések ennél a felállásnál a `slurpd` segítségével azonnal továbbítódnak a másodkiszolgálóra (*2. ábra*).

A mestergép (főcsomópont) hibája esetén a szolga teljes egészében átveheti kérések kiszolgálását, de mivel a szolgán nem végezhetünk adatfelvitelt, csak lekérdezni tudunk. Hogy teljes értékű szolgáltatásunk legyen, a másodkiszolgálón mesterpéldányként újraindítjuk az LDAP-szolgáltatást (*3. ábra*).

Ez a második felállás teljes körű LDAP-szolgáltatásokat biztosít, azonban egyetlen dologgal még mindig számolnunk kell. Ha a másodkiszolgáló adatbázisa frissül, a főkiszolgáló adatbázisát a hiba elhárítása után, még szolgálatba állítás előtt frissíteni kell.

A Hearbeat rendelkezik egy nagyszerű hibavédelmi lehetőséggel (failback), mely megakadályozza, hogy a kiesett csomópont engedély nélkül ismét erőforrásokat foglaljon le egy leállást követően. Ezt a beállítást érdemes bekapcsolni, és az újraindítást kézzel elvégezni.

A mi kiépítésünk emellett a Heartbeat által biztosított látszólagos IP (Virtual IP) szolgáltatást is használja. Ha erős terhelésre kell felkészülnünk, a látszólagos IP helyett egy IP-szórót vagy terheléselosztó rendszert is használhatunk, ami szétosztja a kéréseket a kiszolgálók között. Ebben az esetben, ha egy másodkiszolgáló adatbázisát próbáljuk módosítani, egy átirányítást (referral) kapunk vissza. Az átirányítások követése nem történik meg önműködően, azt az ügyfélprogramnak kell megvalósítania. Az elsődleges és másodlagos csomópontok ugyanúgy vannak beállítva, eltekintve a többszörözésért felelős kapcsolóktól (a Linuxvilág honlapján ➲ http://www.linuxvilag.hu/HA-LDAP – és a Linux Journal FTP-kiszolgálóján – ftp.ssc.com/pub/lj/listings/issue104/5505.tgz fájlban – elérhető). A főkiszolgáló beállításait tartalmazó állományban találjuk meg a



3. ábra
A másodkiszolgáló újraindul mesterként

többszörözés naplójának a helyét, és a másodkiszolgálók listáját, vagyis azokat a címeket, melyeknek az azonosítási adatokat tartalmazniuk kell:

```
replica host=slave5:389
binddn="cn=Manager,dc=lcc,dc=ibm,
➥dc=com";
bindmethod=simple credentials=secret
```

A másodkiszolgálók beállításai között nincs utalás a főkiszolgálóra. Ehelyett felsorolja a sokszorosításhoz szükséges követelményeket:

```
updatedn
➥"cn=Manager,dc=lcc,dc=ibm,dc=com"
```

A következő részben a Hartbeat beállításával folytatjuk cikkünket.

Linux Journal 2002. december, 104. szám

Cliff White (cliffw@osdl.org)
Az OSDL műszaki munkatársa (www.osdl.org). 1989 óta dolgozik különböző Unix- és Linux-változatokkal. Hogy biztos legyen a dolgában, minden reggel azonosítja önmagát.

Jay D. Allen (allen5@us.ibm.com)
Nappal a legújabb IT-megoldásokon dolgozik programfejlesztőként az IBM Linux for Service Providers Labnél (LSPL), ahol Linuxot használ Intelen és RS/6000-en. Éjszakánként elmaradott megoldásokkal foglalatoskodik, főként a DEC PDP-11-ével, és egyéb régiségekkel.

A IV. GNU/Linux Szakmai Konferencia tapasztalatai

A Linux-felhasználók Magyarországi Egyesületének (LME) szervezésében novemberben sor került hazánk legfontosabb linuxos megmozdulására.

Az LME megalakulásakor célul tűzte ki a Linux népszerűsítését. Az egyesület ennek megvalósítása érdekében minden évben megrendezi immár szokásossá váló szakmai konferenciáját. Az évek során a rendezvényt sikerült olyan színvonalra emelni, hogy minden olyan hazai cég, amelyik úgy érzi, hogy jelenléte a linuxos piacon fontos, kiállítóként vagy látogatóként megtalálható rajta. Egy ilyen nagyszabású megmozdulás megszervezése meglehetősen komoly feladat. Ezzel kapcsolatban beszélgettem Sári Gáborral, az LME elnökével.

Gibizer Tibor: *Az eddigi évek során egyre színvonalasabbra sikeredett a konferencia. Ez jól érzékelhető, ha a megjelent kiállítók névsorára tekintünk. Mennyi idő kellett a szervezéshez?*
Sári Gábor: Sajnos nagyon kevés idő volt a konferencia szervezésére. Természetesen tapasztalatainkat megpróbáljuk összegyűjteni, hogy a következő évben még gördülékenyebben zajlódjék le a rendezvény. Az LME mint egyesület állandó erőforráshiánnyal (értsd: aktív tag) küzd. A vezetőséget egy évre választjuk, ennek az a következménye, hogy mire mindenki belerázódik a feladatába, már jön is az új vezetőség. Talán majd egyszer ezen is tudunk változtatni. A szervezés sajnos idén is későn indult, mivel az elnökség megbízható adatok híján nem akart egy ekkora vállalkozásba belevágni, ezért az egyesület aránylag későn fogott bele a rendezvény megszervezésébe.
Ma már tudom, hogy nagy vállalkozás egy ilyen konferencia megrendezése, de induláskor (körülbelül augusztus elején kezdődött a szervezés) még egy kicsit jóhiszemű rácsodálkozásokkal vágtam bele.
G. T.: *Egy fáradt, de boldog ember ül velem szemben. Milyen tapasztalataid, érzéseid vannak az egyesület idei tevékenyégeivel kapcsolatban?*
S. G.: Nem tagadom, az elmúlt hónapok, amióta az egyesület irányítását elvállaltam, valóban kifárasztottak egy kicsit, de ugyanakkor van bennem némi büszkeség is, hogy eddig sikerült a feladatoknak megfelelni, és itt természetesen elsősorban a konferencia szervezésére gondolok. Rögtön hozzá is kell tennem, hogy ez nem egyedül az én érdemem, hiszen egy kis létszámú csapat, lényegében 3–4 fő vett részt a feszített tempójú szervezésben. Szerencsére a tényleges lebonyolításnál több tag is segített. Fontosnak tartom kiemelni, hogy ez a konferencia nem jött volna létre, ha az UHU-Linux nem segít. Természetesen nélkülük is meg tudtuk volna oldani, de hihetetlenül nagy segítséget jelentett már az a tudat is, hogy ők ott állnak a háttérben, és szinte mindenben a segítségünkre tudnak lenni. Ők voltak a fő támogatóink, amivel talán példaértékűen bizonyították, hogy a Linux ereje az összefogásban és egymás segítésében rejlik.
G. T.: *A tanácskozás a Grand Hotel Hungáriában került megrendezésre. Miért ezt a helyszínt választottátok?*
S. G.: Szerettem volna végre komoly üzleti környezetben látni a Linux-konferenciát. A cél az volt, hogy végre mi is eljussunk arra a szintre, ami a környező országokban hasonlóan fontos rendezvényeknél tapasztalható. Ehhez természetesen a kiváló szakmai előadókon túl egy olyan helyszínt kellett találnunk, ahol a kiállítók is örömmel jelennek meg, jól megközelíthető, megfelelően színvonalas, kényelmes, megoldott az ebédeltetés, a technikai feltételek adottak, így látogatóink is megelégedéssel vehetnek részt a rendezvényen. A szervezésből közel egy hónap csak arra ment el, hogy választ kapjunk a következőre: érdemes-e és lehetséges-e a konferenciát egy hotelben megrendezni? A legfontosabb mozzanatra is ekkor derült fény: melyik hotel szabad még? Megnehezítette a dolgunkat az is, hogy az előadásokat párhuzamosan kívánták tartani, akár 3–4 előadást is egy időben, amihez megfelelő méretű termekre volt szükség, illetve a kiállítói helyek mérete is meghatározó volt.
G. T.: *Tudnál tényszerű adatokat említeni?*
S. G.: A végleges pénzügyi elszámolás még folyamatban van, de annyit talán már elárulhatok, hogy nagy valószínűséggel jól gazdálkodtunk, a jelek szerint sikerült a rendezvényt ráfizetés nélkül megtartani, ami nagyjából annyit jelent, hogy csak annyit költöttünk, amennyi a rendezvény bevétele volt. A két legnagyobb kiállítónk a LafiSoft és a Magyar BSD Egyesület volt. A belépéskor elvégzett regisztráció adatai szerint a rendezvénynek 366 fizető látogatója volt, ami tökéletesen megfelelt az eredeti terveknek.
G. T.: *Milyen visszhangja volt a rendezvénynek?*
S. G.: A visszhang általánosan roppant kedvező volt. Természetesen – mint mindig – most is akadtak, akik a rendezvényen nem találták meg a számításukat. A jövő évi szervezés során mind a kedvező, mind a sérelmező véleményeket fokozott mértékben figyelembe vesszük, pontosabban átadjuk az LME jövő évi vezetőségének, mivel a szervezést minden évben az LME elnöksége egyezteti össze.
Felmérésünk alapján az előadók és a rendezvényt támogató, illetve kiállító cégek szintén kedvezően vélekedtek a rendezvényről.
G. T.: *Személyesen is részt vettem a konferencián, és teljes nyugalommal állíthatom, hogy az eddigi éveknek megfelelően a nap igazán jó hangulatban telt. Ha hasonló érdeklődésű embereket zárunk össze, ott gyorsan kialakul egy többéves barátságra utaló légkör. Belülről – mint a szervezők egyike – te miként értékelnéd ilyen szemszögből a rendezvényt?*
S. G.: Elfogult vagyok, még így utólag is. Az én hangulatom kimondottan vidám volt. Az eddig tudomásomra jutott vélemények is nagyon jó hangulatról árulkodtak. Tegyük hozzá, hogy az azonos érdeklődési kör valóban összehozza az embereket, de ne felejtsük el, hogy hazánkban még mindig kevesen vannak a linuxosok, ezért lépten-nyomon ismerős arcokkal vagy valamelyik gyakran hallott becenév viselőjével találkozhatunk. Ezért is

öröm ide eljönni, mert rengeteg ismerőssel, sőt ismeretlen ismerőssel találkozhatunk.
Egy apró, de annál kellemesebb történetet megosztanék meg a Linuxvilág olvasóival:
Az egyik látogatónk letette a kabátját és a szatyrát egy székre. A pénztárcája feltehetően kiesett a kabát belső zsebéből, ami többek között fizetésének nagy részét is tartalmazta. Azért mondom azt, hogy „feltehetően", mert a nap végére a pénztárca átvándorolt a szatyorba, természetesen hiánytalan tartalommal. Erre a legkézenfekvőbb magyarázat, az hogy valaki észrevette, de nem tudta kié az otthagyott holmi, ezért betette a pénztárcát a kabát melletti szatyorba. Gondolom, nem akart a kabátban turkálni. Tehát csak azt tudom mondani, hogy a kiállítók, a résztvevők, tehát lényegében mindenki, aki megjelent, jól érezte magát.

G. T.: *Szinte minden cég, amelyik közvetlenül vagy közvetetten érintett a Linux világában, képviselte magát. Az eddigi évekkel ellentétben az idén nem találkozhattunk a SuSE nevével. Mi volt ennek az oka?*

S. G.: Több céget is megkerestünk, akik az elmúlt években részt vettek a konferencián. Minden cég magánügye, hogy meg kíván-e jelenni a rendezvényen. Természetesen több megkérdezett cég is volt, akik úgy döntöttek, hogy idén nem látogatnak el a konferenciára. Ezek közé a cégek közé tartozott a „kaméleon-Linux" cége is.

G. T.: *Sokan panaszkodtak, hogy nem értesültek időben az eseményről. Hol hirdettétek meg?*

S. G.: Az „esemény" beharangozása szeptember elején indult a linuxos témájú levelezőlistákra küldött körlevélben és a Linux.hu híroldalon. A konferencia hivatalos weboldala pedig szeptember végén indult be.
A számítástechnikai sajtóban összesen négy hirdetés tudósított róla, de emlékeim szerint hírt adtak a nevesebb linuxos portálok (linux.index.hu, LinuxMánia stb.), valamint a Linuxvilág is, és ha jól emlékszem, a Fix.tv is többször hírt adott róla.

G. T.: *Tervez az LME hasonló nagyszabású megmozdulást?*

S. G.: Természetesen tervezünk, viszont a szervezésben résztvevők kis száma miatt nagyon meg kell gondolnunk, mibe vágunk bele a jövőben.
A konferencián is elhangzott az a tervünk, hogy 2003 első félévében rendezünk egy Szabad Szoftver Konferenciát, amelyet az FSF.HU, a Magyar BSD Egyesület és az LME közösen rendezne, de erről közelebbi részleteket még nem tudok mondani.

G. T.: *Sokak szerint elég körülményes, nehezen értelmezhető volt a jelentkezési lap – ez nem jelentett hátrányt?*

S. G.: Igen, ez egy olyan részlete volt az idei konferenciának, ami újdonságnak számított, így talán egy kicsit szokatlan is, de minden komolyabb szakmai konferencián ez a gyakorlat. Szükség volt rá, hogy elkerüljük a 2001. évi konferencia elején megesett torlódást. Ezt a kitűzött célt maradéktalanul sikerült megvalósítani, ugyanis a belépés gördülékenyen lezajlott. Ezt a fajta jelentkezést a jövőben is alkalmazni fogjuk; amin változtatnunk kell, az az oldallal kapcsolatos egyértelmű tájékoztató összeállítása.

G. T.: *Ma már szinte természetes, hogy minden LME-rendezvényen teljes valójában megjelenik a „CD-író Projekt", azaz Balázs Tibor (covek@tux.linux.hu). Talán ennek a természetességnek köszönhető, hogy hajlamosak vagyunk megfeledkezni értékes munkájáról. Idén is volt igény a helyben írott lemezekre?*

S. G.: Balázs Tibor természetesen ott volt és folyamatosan írt, körülbelül 30–40 CD-t készített az LME CD-író projekt vezetője. Ő az egyike azoknak, akiknek a munkájára feltétlen lehet számítani. Köszönet érte, és engedd meg, hogy megragadjam az alkalmat, és köszönetet mondjak mindenkinek, aki közvetlenül vagy közvetett módon részt vállalt a rendezvény sikeres lebonyolításában.

G. T.: *Akkor nekem már csak annyi feladatom maradt, hogy hasonlóan kiváló rendezvényeket és további jó munkát kívánjak! Köszönöm a beszélgetést.*

Gibizer Tibor (gibzo@linuxmania.hu)
Újságíró, immár hét éve a Linux elkötelezett híve. Imádja a kutyákat, a kerékpározást és az autós csavargást.

Ők mondták

Amíg az ember elfogadja a megszokottat, addig állandó kétségek rabja lesz, de ha helyet ad egy józan kétely csábításának, végül rátalál az igazságokra. *(Sir Francis Bacon)*

Az emberiséget három nagy csoport alkotja: a mozdíthatatlanok, a mozdíthatók, és azok, akik mozdulnak. *(Benjamin Franklin)*

Azért van két fülünk és egy nyelvünk, hogy többet halljunk, és kevesebbet beszéljünk. *(Diogenész)*

Ha a kvantummechanika nem rázott meg, bizonyára nem értetted meg. *(Nils Bohr)*

Minden nemzedék azt képzeli magáról, hogy intelligensebb az előtte járónál és bölcsebb az utána következőnél. *(George Orwell)*

A legnagyobb szerencsétlenség az, amikor az elmélet túlszárnyalja a gyakorlatot. *(Leonardo da Vinci)*

A tévedés híd a tapasztalatlanság és a tudás között. *(Phyllis Theroux)*

Több szót fecsérel az ember arra, amit vél, mint arra, amit tud. *(Cullen Hightower)*

Az ember akkor ember, ha az összes választási lehetőség közül a legnehezebbet választja. *(Bolyai János)*

öröm ide eljönni, mert rengeteg ismerőssel, sőt ismeretlen ismerőssel találkozhatunk.
Egy apró, de annál kellemesebb történetet megosztanék meg a Linuxvilág olvasóival:
Az egyik látogatónk letette a kabátját és a szatyrát egy székre. A pénztárcája feltehetően kiesett a kabát belső zsebéből, ami többek között fizetésének nagy részét is tartalmazta. Azért mondom azt, hogy „feltehetően", mert a nap végére a pénztárca átvándorolt a szatyorba, természetesen hiánytalan tartalommal. Erre a legkézenfekvőbb magyarázat, az hogy valaki észrevette, de nem tudta kié az otthagyott holmi, ezért betette a pénztárcát a kabát melletti szatyorba. Gondolom, nem akart a kabátban turkálni. Tehát csak azt tudom mondani, hogy a kiállítók, a résztvevők, tehát lényegében mindenki, aki megjelent, jól érezte magát.

G. T.: *Szinte minden cég, amelyik közvetlenül vagy közvetetten érintett a Linux világában, képviselte magát. Az eddigi évekkel ellentétben az idén nem találkozhattunk a SuSE nevével. Mi volt ennek az oka?*

S. G.: Több céget is megkerestünk, akik az elmúlt években részt vettek a konferencián. Minden cég magánügye, hogy meg kíván-e jelenni a rendezvényen. Természetesen több megkérdezett cég is volt, akik úgy döntöttek, hogy idén nem látogatnak el a konferenciára. Ezek közé a cégek közé tartozott a „kaméleon-Linux" cége is.

G. T.: *Sokan panaszkodtak, hogy nem értesültek időben az eseményről. Hol hirdettétek meg?*

S. G.: Az „esemény" beharangozása szeptember elején indult a linuxos témájú levelezőlistákra küldött körlevélben és a Linux.hu híroldalon. A konferencia hivatalos weboldala pedig szeptember végén indult be.
A számítástechnikai sajtóban összesen négy hirdetés tudósított róla, de emlékeim szerint hírt adtak a nevesebb linuxos portálok (linux.index.hu, LinuxMánia stb.), valamint a Linuxvilág is, és ha jól emlékszem, a Fix.tv is többször hírt adott róla.

G. T.: *Tervez az LME hasonló nagyszabású megmozdulást?*

S. G.: Természetesen tervezünk, viszont a szervezésben résztvevők kis száma miatt nagyon meg kell gondolnunk, mibe vágunk bele a jövőben.
A konferencián is elhangzott az a tervünk, hogy 2003 első félévében rendezünk egy Szabad Szoftver Konferenciát, amelyet az FSF.HU, a Magyar BSD Egyesület és az LME közösen rendezne, de erről közelebbi részleteket még nem tudok mondani.

G. T.: *Sokak szerint elég körülményes, nehezen értelmezhető volt a jelentkezési lap – ez nem jelentett hátrányt?*

S. G.: Igen, ez egy olyan részlete volt az idei konferenciának, ami újdonságnak számított, így talán egy kicsit szokatlan is, de minden komolyabb szakmai konferencián ez a gyakorlat. Szükség volt rá, hogy elkerüljük a 2001. évi konferencia elején megesett torlódást. Ezt a kitűzött célt maradéktalanul sikerült megvalósítani, ugyanis a belépés gördülékenyen lezajlott. Ezt a fajta jelentkezést a jövőben is alkalmazni fogjuk; amin változtatnunk kell, az az oldallal kapcsolatos egyértelmű tájékoztató összeállítása.

G. T.: *Ma már szinte természetes, hogy minden LME-rendezvényen teljes valójában megjelenik a „CD-író Projekt", azaz Balázs Tibor (covek@tux.linux.hu). Talán ennek a természetességnek köszönhető, hogy hajlamosak vagyunk megfeledkezni értékes munkájáról. Idén is volt igény a helyben írott lemezekre?*

S. G.: Balázs Tibor természetesen ott volt és folyamatosan írt, körülbelül 30–40 CD-t készített az LME CD-író projekt vezetője. Ő az egyike azoknak, akiknek a munkájára feltétlen lehet számítani. Köszönet érte, és engedd meg, hogy megragadjam az alkalmat, és köszönetet mondjak mindenkinek, aki közvetlenül vagy közvetett módon részt vállalt a rendezvény sikeres lebonyolításában.

G. T.: *Akkor nekem már csak annyi feladatom maradt, hogy hasonlóan kiváló rendezvényeket és további jó munkát kívánjak! Köszönöm a beszélgetést.*

Gibizer Tibor (gibzo@linuxmania.hu)
Újságíró, immár hét éve a Linux elkötelezett híve. Imádja a kutyákat, a kerékpározást és az autós csavargást.

Ők mondták

Amíg az ember elfogadja a megszokottat, addig állandó kétségek rabja lesz, de ha helyet ad egy józan kétely csábításának, végül rátalál az igazságokra. *(Sir Francis Bacon)*

Az emberiséget három nagy csoport alkotja: a mozdíthatatlanok, a mozdíthatók, és azok, akik mozdulnak. *(Benjamin Franklin)*

Azért van két fülünk és egy nyelvünk, hogy többet halljunk, és kevesebbet beszéljünk. *(Diogenész)*

Ha a kvantummechanika nem rázott meg, bizonyára nem értetted meg. *(Nils Bohr)*

Minden nemzedék azt képzeli magáról, hogy intelligensebb az előtte járónál és bölcsebb az utána következőnél. *(George Orwell)*

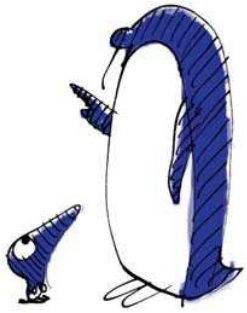
A legnagyobb szerencsétlenség az, amikor az elmélet túlszárnyalja a gyakorlatot. *(Leonardo da Vinci)*

A tévedés híd a tapasztalatlanság és a tudás között. *(Phyllis Theroux)*

Több szót fecsérel az ember arra, amit vél, mint arra, amit tud. *(Cullen Hightower)*

Az ember akkor ember, ha az összes választási lehetőség közül a legnehezebbet választja. *(Bolyai János)*

A hónap szakmai tanácsai



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

Szia, Világ: miért nem megy?

Nemrég telepítettem a Linuxot, és a következő hibaüzenetet kapom, ha a rendszeren végrehajtható állományokat akarok futtatni:

```
bash: a.out command not found
```

Hogyan javítható ki a hiba?

Manuel Sevilla, slickspick@yahoo.com

Úgy tűnik, a pillanatnyi könyvtár nincs benne az elérési útban. Egy frissen fordított C-programot a pillanatnyi könyvtárból a `./a.out` paranccsal lehet elindítani.

Robert Connoy, rconnoy@penguincomputing.com

Grafikus felület adatbázishoz oktatási célra?

Ausztráliában tanítok, és Linux-hálózatunkhoz adatbázis-kezelőt keresek. Olyan adatbázis-kezelőt szeretnék, amilyen a Microsoft Access, mert egyszerű használni és könnyű megtanítani a diákoknak.

Ken Jordan, kwjordan@cedars.nsw.edu.au

Van egy pgaccess nevű program (➲ http://pgaccess.org), ami PostgreSQL-lel működik, és nagyon hasonlít a Microsoft Accessre. A másik megoldás, hogy az OpenOffice.org adatbázis-kapcsolati lehetőségeit használod, ODBC-n keresztül.

Felipe E. Barousse Boué, fbarousse@piensa.com

Hiba vagy biztonsági óvintézkedés?

Red Hat 7.1-es rendszert használok, és IMAP, POP3, illetve POP2 fut rajta. Ezek a szolgáltatások nem fogadják el a rendszergazda nevét és jelszavát, de a többi felhasználóét igen.

Pedro Guedes, pmg01@netc.pt

A legegyszerűbb a rendszergazda számára egy nem a rendszergazdára mutató álnevet beállítani a */etc/aliases* állományban. Ne felejtsd el futtatni a `newaliases` parancsot a */etc/aliases* frissítése után!

Christopher Wingert, cwingert@cwingert.qualcomm.com

Soha ne próbálj bejelentkezni az IMAP vagy POP kiszolgálóra rendszergazdaként, különösen akkor ne, ha nem használsz SSL-t. Utóbbi esetben a rendszergazda jelszava egyszerű szövegként halad át a hálón, és bárki ellophatja.

Marc Merlin, marc_bts@google.coml

Frissítés RPM-mel

A KDE 3.0.3-at RPM-csomagokból próbáltam meg telepíteni egy újonnan telepített rendszerre, amelyen nem volt sem KDE, sem Gnome. A `qt-3.0.5-16.i386.rpm` telepítésénél a következő hibaüzenetet kaptam:

```
[root@yeller rpms]# rpm -Uvh
  ➥qt-3.0.5-16.i386.rpm
error: failed dependencies:
  libcups.so.2   is needed by
  ➥qt-3.0.5-16
   libpng12.so.0  is needed by
   ➥qt-3.0.5-16
```

Ezután ezzel próbálkoztam:

```
[root@yeller rpms]# rpm -qa
➥| grep libpng
libpng-1.0.14-0.7x.3
libpng-devel-1.0.14-0.7x.3
[root@yeller rpms]
➥# rpm -Uvh libpng-1.2.2-6.i386.rpm
+libpng-devel-1.2.2-6.i386.rpm
```

Megint a ki nem elégített függőségekre vonatkozó hibaüzenetet kaptam. Próbáltam kielégíteni a függőségeket a legalsó szintről kiindulva, de belegabalyodtam az RPM-csomagok pókhálójába – egyre több függőség jelent meg. Hogyan lehet anélkül csomagokat telepíteni vagy frissíteni, hogy a más csomagok által megkövetelt dolgokat ne töröljünk le vagy ne írjunk felül?

James Weisensee, itjayw@yahoo.com

A Red Hat által készített `up2date` segédprogram ezt próbálja megoldani. Az `up2date` használatához az `rhn_register` parancs segítségével fel kell iratkoznunk a Red Hat Network felhasználói közé. Az `up2date` frissíti a rendszerre telepített csomagokat, ha új változatuk jelenik meg. Ezek legtöbbször a biztonsági hibajavításokat tartalmazzák.

Az általad próbált `rpm -qa` parancsok csak a már telepített RPM-csomagok adatbázisát kérdezik le. Ha az érdekel, hogy egy adott könyvtárban található RPM-ek közül melyik tartalmaz egy bizonyos fájlt (például a libcups.so.2 vagy a libpng12.so.0), használd inkább a következő parancsot:

```
for f in  libcups.so.2 libpng12.so.0; do
  for i in *.rpm; do
      rpm -qpl | grep
➥-q $f && echo $i;
  done
done
```

Ez a parancs a megadott állományneveket az összes RPM-csomagban keresi, és kiírja azoknak a csomagoknak a nevét, amelyekben valamelyik megtalálható. Ez nem mindig működik (egyes esetekben a keresett állományt a csomag telepítése után lefutó parancsfájl hozza létre). Ezenkívül bizonyos függőségeknek nincs állományneve, csak elvont azonosítója, amelyet több helyettesítő csomag is biztosíthat.

Jim Dennis, jimd@starshine.org

Gyakran megszabadulhatunk az efféle RPM-frissítési bonyodalomtól, ha az összes lényeges RPM-csomagot egyszerre, ugyanabban a parancssorban frissítjük. Bővítgessük addig a parancssort az RPM-csomagok neveivel, amíg az `rpm -Uvh` parancs sikeres nem lesz. Ez a módszer az egymással összefüggő csomagok eltávolítására is alkalmas.

Don Marti, dmarti@ssc.com

Linux Journal 2002. december, 104. szám

Új termékek

white dwarf linux v1.2

Az EMJ Embedded Systems bejelentette a *white dwarf linux* legújabb változatát, amely 10 MB flashmemóriát és 16 MB DRAM-ot használó készülékekhez tervezett beágyazott operációs rendszer. A white dwarf linux minden alaplapot vagy egykártyás számítógépet támogat, amelyben van IDE-vezérlő és legalább 8 MB RAM. Támogatja a CD-s és a hálózati telepítést is. Az 1.2-es változat tartalmazza a 2.4.19-es rendszermagot, a grafikuscsomag-alapú telepítőt és a GCC fejlesztőeszközöket a glibc 2.2.5-höz. A white dwarf linux a következő kártyákkal működik: DIMM-PC 486 és 520, MOPS686+, CoolMONSTER és Tri-M MZ104.

Adatok: EMJ America,
220 Chatham Business Drive,
Pittsboro, North Carolina 27312,
telefon: 800-548-2319,
➲ http://www.emjembedded.com

EnCore M3

Az Ampro Computers új, beágyazott rendszerekben használható MIPS alapú modullal jelent meg. Az EnCore M3 az AMD 400 MHz-es Alchemy Au1500 lapkáját és a szabványos EnCore-jellemzőket kombináló termék, ami teljesen szabványos CPU-alrendszert képez a kisméretű modulban. Az EnCore M3 480 Dhrystone MIPS sebességet ért el, miközben fogyasztása átlagosan 2,5 watt alatt maradt. A modul rendelkezik egy 32 bites, 66 MHz-es PCI csatolósínnel. Az EnCore M3 100×150 mm méretű, két 10/100 ethernetkaput és egy AC97 hangrendszert tartalmaz. Legfeljebb 256 MB SODIMM SDRAM-ot támogat, van benne 2 MB flashmemória, két soros kapu, két USB-kapu, hajlékonylemez-vezérlő, PS/2 billentyűzet- és egérkapu, IrDA-kapu és egy ECP/EPP kétirányú párhuzamos kapu.

Adatok: Ampro Computers, Inc.,
5215 Hellyer Avenue #110,
San Jose, California 95138,
telefon: 800-966-5200,
➲ http://www.ampro.com

Black Lab v2.1

A Black Lab v2.1 Yellow Dog Linuxot futtató HPC-telepek felépítését és kezelését segítő programcsomag. Egy-két kattintással telepíthető és beállítható, az önműködő frissítést az `apt-get` végzi, grafikus felülete van, több telep is kezelhető vele, és minden szolgáltatás a parancssorból is vezérelhető. Az új változat tartalmazza a BProc 3.0-t, ami az alkalmazásokat és az osztott programkönyvtárakat a kiszolgálóról önműködően átköltözteti a kijelölt csomópontokra. Ez a beállítás teszi lehetővé, hogy az összes csomópont a helyi merevlemezéről csak a legszükségesebb összetevőkkel működjön. A rendszergazda a csomópontokat úgy is testreszabhatja, hogy csak a szükséges programok legyenek telepítve; továbbá saját kiszolgálóoldali csomópont-lemezképek készíthetők.

Adatok: Terra Soft Solutions, Inc.,
117 West Second Street,
Loveland, Colorado 80537,
telefon: 970-278-9243,
presales@terrasoftsolutions.com,
➲ http://www.terrasoftsolutions.com

NAG C Library Mark 7

A Numerical Algorithms Group (NAG) – a matematikával, statisztikával és háromdimenziós ábrázolással foglalkozó számítógéptudósok nemzetközi szervezete – kiadta 850-nél is több függvényt tartalmazó C-programkönyvtárának legújabb változatát. A NAG C Library függvényei között megtalálható a modellezés és a szimuláció, az idősorok elemzése és a statisztikai függvények, amiket a legkülönfélébb programokban fel lehet használni. A függvények Linux és más rendszerek alatt is használhatóak, sokféle programnyelvből, beleértve a Javát és a C++-t.

Adatok: Numerical Algorithms Group, LTD, Wilkinson House,
Jordan Hill Road,
Oxford OX2 8DR, United Kingdom,
➲ http://www.nag.co.uk

miniHiPerCam

Az America ELTEC bemutatott egy olyan beágyazott Linuxszal működő kamerát, ami nem nagyobb, mint egy közönséges megfigyelő kamera. A miniHiPerCam színes és fekete-fehér változatban is elérhető, és az ELINOS beágyazott változatán alapul.

A CMOS-képérzékelő 640×480 képpontos felbontással bír, és 15 vagy 30 képet vesz fel másodpercenként. A beépített vezérlőkártya 50 MHz órajelű PowerPC823 processzort használ. A képek és a futásidejű adatok tárolására 16 MB memória áll rendelkezésre, míg az operációs rendszer a beépített HTTP-kiszolgálóval és az alkalmazásokkal a 8 MB méretű flashmemóriában tárolódik. A kamera két RS-232-es kapuval és egy 10 Mb-es ethernet-kapuval rendelkezik.

Adatok: American ELTEC,
2810 West Charleston Avenue,
Suite 57, Las Vegas, Nevada 89102,
telefon: 702-878-4085,
➲ http://www.americaneltec.com

Lindows 2.0

A Lindows 2.0 a Lindows.com legújabb operációs rendszere.

A 2.0-s változat újdonságai közül kiemelhető az újratervezett grafikus felület, több mint 800 nyomtató támogatása és az SMB-nyomtató-kiszolgálók használatának lehetősége. A Lindows 2.0 a Netscape 7.0-t használja elektronikus levelezésre és webböngészésre, az utóbbi támogatja a többlapos böngészést és a felbukkanó reklámablakok megjelenésének letiltását. A hordozható gépek felhasználói számára a Lindows 2.0 lehetővé teszi az energiakezelést és az akkumulátor vezérlését. A hálózat terén az új kiadás a továbbfejlesztett WiFi-támogatást és a windowsos fájlkiszolgálók használatának lehetőségét nyújtja.

Adatok: Lindows.com, Inc.,
9333 Genesee Avenue, 3rd Floor,
San Diego, California 92121,
telefon: 858-587-6700,
➲ http://www.lindows.com

Linux Journal 2002. 104. szám

Általános PC személyazonossággal

Elsőként *Scott McNealy* említette meg a LinuxWorld Expón, 2002 augusztusában. A Sun Microsystems elnöke és ügyvezető igazgatója programbeszédében elmondta, hogy a vállalat új linuxos asztali gépet fog bejelenteni szeptemberben a SunNetwork tanácskozáson. Amikor megtörtént a bejelentés, úgy tűnt, másról szól, mint amit ígértek. Nem egyszerűen egy újabb, Linuxot futtató gépet tettek elénk, hanem egy olyan asztaligép-tervet, aminek az a célja, hogy kiaknázza az olcsó és rugalmas, névtelen x86-os számítógépek óriási piacának lehetőségeit, illetve általában a költségmegtakarításból eredő előnyöket.

Az új stratégia álneve Mad Hatter. *Curtis Sasaki*, a Sun asztaligép-megoldások fejlesztéséért felelős alelnöke így mutatta be: ez a bejelentés egy teljes csomagról szól, nem pusztán egy gépről. Magában foglalja a számítógépet, a programokat, szolgáltatásokat, valamint a kiszolgálóoldali középréteget is. Maga a számítógép egy asztali gép, amelyen megtalálható a Linux-rendszermag, a Gnome GUI, beépített Java 2-felület, a Mozilla, a StarOffice és a Ximian alkalmazáscsomagja, az Evolution. A hangsúly a beépítettségen van. Ez a gép azonnal használható, méretezhető, van rajta címtár, naptár, üzenetkezelő, illetve a hozzáférés-felügyelet érdekében Java-kártyákkal is rendelkezik (a Java-kártya Java nyelven programozható kártyát jelent).

A termék a költségtudatos, nagyszámú üzletkötőt foglalkoztató vállalatokat célozza meg. Ám nem pusztán zümmögő dobozokat akarnak eladni a kocka formájú kaptáraknak: a Mad Hatter mindenekelőtt az egyének azonosítására helyezi a hangsúlyt.

„A vezetők manapság pontosan tudni akarják, mekkora költséget jelent az egyes felhasználók elektronikus levelezése, naptár- és címtárhasználata, illetve mennyi a biztonság ára" – jelentette ki Sasaki. A Sun elképzelése szerint a vállalatoknak olyan Linux PC-kkel kell felszerelkezni, amelyek hitelesítéskor a Java-kártya alapján egyértelműen azonosítják a felhasználót, és valóban mutatkozik erre igény. Sasaki szerint a vásárlók nemcsak a nyílt szabványon alapuló rendszerek kedvező árának köszönhetően mutatnak komoly érdeklődést, az ajánlatunkban szereplő különleges alkatrész szintén csábító: a rendszergazda a Java-kártya segítségével a honlap és az alkalmazás alapján elvégezheti a felhasználó hitelesítését.

Ezzel elérkeztünk egy teljesen új besoroláshoz, amelyhez egy új betűszó is társul. Sasakit idézve: „Ez nem PC, hanem IdC – identity computer, vagyis azonosító számítógép. Az azonosítás nagyon fontos. Ez teszi lehetővé, hogy igazoló adatai alapján az ember bárhonnan hozzáférhessen az asztali gépén tárolt adatokhoz. "

Eric Norlin, a Digital ID World egyik elemzője így vázolta körülményeket: a vállalati informatikai részleg mára jóformán puszta költséghellyé vált, és gyakorlatilag az azonosításfelügyelet az egyetlen terület, ahol – a titoktartás és az adatvédelem fokozása mellett – pénzt takaríthat meg. Az azonosítási megoldások várhatóan akkor is fejlődni fognak, amikor a többi szolgáltatás helyben jár, mivel egyedül az azonosítástól remélhető, hogy az informatikai részleg költség helyett nyereséget termel. A Digital ID segítségével valóban lehetségessé válik az informatikai beruházások megtérülése.

A Sun ezen túlmenően azt is meg kívánja valósítani, hogy egyetlen azonosításra legyen szükség a vállalaton kívül (vállalatok között), csakúgy, mint a felhasználó munkahelyén. A részleteket Sasaki ekképpen magyarázta el: a Liberty Alliance 115, különböző területen működő vállalat összefogásával kidolgozott egy nyílt szabványt, amely megoldást kínál arra az igényre, hogy a felhasználók egyetlen bejelentkezéssel sok szolgáltatást érjenek el. Hamarosan számos különféle Liberty szabványú szolgáltatás jelenik meg, amelyek biztonságos módon fogják kezelni a személyazonosságot.

Amikor üzleti kapcsolatba kerül egy olyan vállalattal, ami szintén Liberty típusú azonosítási rendszert alkalmaz (legyen például a United Airlines vagy az American Express), akkor újbóli azonosítás nélkül léphet át az egyik rendszerből a másikba.

Mivel a Mad Hatter egy Linuxra épülő keretrendszer, ráilleszthető a vásárlók meglévő x86-os gépeire, illetve „tiszteletben tartja" az olyan országok igényeit, mint például Kína, ami ragaszkodik a hazai gyártású dolgokhoz. A számítógép mindössze az egyik cserélhető összetevője. *Shahin Khan*-t, a Sun versenyügyekért felelős igazgatóját idézve: „Olyan a felépítése, hogy bármelyik összetevője tetszés szerint eltávolítható vagy kicserélhető."

Miért éppen most? Sasaki szerint azért, mert végre elkészültek a linuxos asztali gépekre tervezett programok: „Az utóbbi 12–18 hónapban valóban kifejlődött a „Linux az asztali gépen" (Linux on the Desktop – LOTD). Egy évvel ezelőtt még hiányoztak az alapok. Nem tudtunk egy teljes asztaligép-megoldást felmutatni – az irodai programcsomagunk nem volt készen. A Gnome és a KDE sem volt elég fejlett, a Mozilla még az 1.0-s változatnál sem tartott. Most azonban a Gnome 2.0 kifejezetten menő, a StarOffice 6 iránt is nagy az érdeklődés, az Evolution pedig igazán figyelemre méltó termék. Most már készen állunk."

A prototípusokat a Sun iForce központjaiban fogják még ez év vége előtt összeállítani. Az első IdC-ket várhatóan 2003 első negyedévében szállítják ki.

Linux Journal 2002. december, 104. szám

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

Rendszermag-fejlesztési hírek

Több logikai processzor egyetlen fizikai processzoron – ez más néven a sokszálúság (hyperthreading, azaz HT); ez a viszonylag új fogalom az Intel Pentium 4 Xeon processzorával vált ismertté, amelyet újabban nagymértékben támogat a Linux-rendszermag. Elsőként *Molnár Ingo* alkalmazta 2002 augusztusában egy javítófájlban, ami több szál kezelésére tette alkalmassá az ütemezőt. Azóta sok javítófájl született, és a sokszálúság általánosan bevett szolgáltatás lett.
Búcsút mondhatunk a `khttpd`-nek. A megjelenése óta folyamatosan vitatott rendszermagbeli webkiszolgáló a 2.5-ös változatban végérvényesen megszűnik. A szolgáltatást övező viták ellenére nem a heves szócsaták pecsételték meg a `khttpd` sorsát, hanem az a tény, hogy a felhasználói területet használó Tux2 sokkal gyorsabb nála. Még az is szóba került, hogy a Tux2 esetleg szintén bekerül a rendszermagba – bár ezt nagyon sokan ellenzik, ugyanabból az okból, amiért eredetileg a `khttpd`-t támadták. Az ötlet felvet bizonyos szabadalmazással kapcsolatos aggályokat is, és úgy tűnik, igazából senki sem száll érte síkra.
A rendszermag 2002 augusztusa és szeptembere folyamán számos új rendszerhívással bővült. Ezek közül a `clone_startup()` előnye, hogy használatával az x86-os gépeken egyetlen rendszerhívással lehet majd új szálat létrehozni. A glibc új pthread kódja nagymértékben használja ezt a hívást.
Egy kis érdekesség: a 2.5.31-es változat utáni rendszermagok esetén a PC hangszórója mikrofonként használható. Ez teljesen új, szokatlan megoldás. Ahogyan *Jos Hulzink* a linux-kernel levelezőlistán megfogalmazta: „a 2.5.32-es változat úgy vonul be a történelembe, mint a rendszermag, amely megvalósította a hangfelismerést az AT típusú számítógépeken…”
Az új rendszermag-beállítási rendszerért (kernel configuration system) folytatott küzdelem tovább tart. Mivel a CML2-nek láthatóan nincs folytatása, helyette egy halom új beállítási rendszer bukkant fel, melyek közül a `kernel conf` tűnik a legesélyesebbnek. *Roman Zippel* kitartóan dolgozott rajta, és állítása szerint a kód csaknem teljesen kész. A szóbeszéd szerint a `kernel conf` valószínűleg benne lesz a 2.5-ös fában, hivatalosan azonban még nem jelentették be.
Gary Lerhaupt jóvoltából új eszközzel gazdagodtunk, ennek neve `devlabel`. A program lehetővé teszi a tárolóeszközök folyamatos elérését – dinamikus közvetett hivatkozások (dynamic symlinks) révén, emellett a működés közbeni befűzés kezelésére is képes. Egyszerűen fűzd be (mont) az eszközt, és azonnal megjelenik egy közvetett hivatkozás, amelyen keresztül az eszköz elérhető. Ha az eszközt leválasztod, a közvetett hivatkozás is eltűnik.

Zack Brown

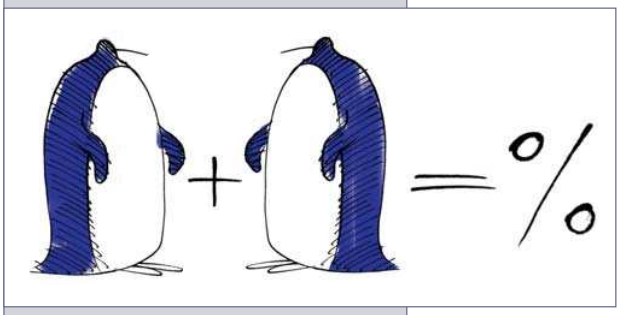
Linux Journal 2002. december, 104. szám

Ő mondta

Az újításnak mindenki ellensége, aki a régi rendszerben sikeres volt, és csak csekély támogatást várhat azoktól, akik az új rendszerben lennének sikeresek. Közömbös viselkedésük oka részben a félelem, részben az, hogy általában véve hitetlenek, sosem bíznak igazán az új dolgokban, hacsak saját maguk ki nem próbálták őket. *(Nicolo Machiavelli)*

Linux-index

1. Az ausztráliai Perth felett repülő betyárok (hacker) ennyi vezeték nélküli számítógépet észleltek egy iPaq segítségével egy négyüléses Grumman kisrepülőgépről: 95



2. Ugyanők Toshiba laptoppal ennyi vezeték nélküli számítógépet észleltek: 92
3. A Grumman repülési sebessége: 402 km/óra
4. A Grumman repülési magassága az út során: 457 méter
5. Ennyi vezeték nélküli számítógépet észlelt *Phil Windley*, Utah állam informatikai kormánybiztosa Salt Lake City felett egy Piper Turbo Arrow repülőgéppel repülve: 27
6. Ezek közül ennyi volt titkosított: 5
7. A Piper repülési sebessége: 201 km/óra
8. A Piper repülési magassága az út alatt: 450 méter
9. A Linux részaránya a 110 millió darabot számláló asztaligéppiacon: 2,7%
10. A linuxos asztali gépek eladásának növekedése 2001-ben: 47%
11. Az új Linux-eladásokból származó bevétel növekedési aránya a múlt évben: 28%
12. Ezzel egy időben az új Unix-eladásokból származó bevétel csökkenésének aránya: 25%
13. Ennyi Zumiez üzletben állítanak össze linuxos asztali gépeket: 100
14. A Zumies üzleteiben átlagosan ennyivel olcsóbb egy ilyen gép, mint egy Microsoft rendszert futtató: 500 dollár

Források
1–4.: E3 ➲ http://www.e3.com.au
5–8.: Phil Windley
➲ http://www.windley.com
9–14.: International Data Corp.

Linux Journal 2002. december, 104. szám

Debian GNU/Linux magyarítási tippek

A továbbiakban igyekszem röviden összefoglalni, miként tudjuk a Debian GNU/Linuxot a magyar nyelv használatára rábírni. Az itt található megoldásnál lehet, hogy létezik egyszerűbb, de mivel nem vagyok a téma szakértője, csak a nálam bevált módszert tudom ismertetni.
Ahhoz, hogy beavatkozásunk sikeres legyen, rendszergazdaként kukkantsunk bele a rendszermagba, majd fordítsuk bele a következőket:

- File system – Native Language Support:
- Default NLS Option: "iso8859-2"
- Codepage 437 (United States, Canada)
- Codepage 852 (Central/Eastern Europe)
- NLS ISO 8859-1 (Latin 1; Western European Languages)
- NLS ISO 8859-2 (Latin 2; Slavic/Central European Languages)

Az esetek többségében bőven elég a fenti csomagokat modulba tenni, de természetesen szabadon csemegézhetünk a lehetőségek között. Következő lépésként a csomagkezelőt (`dselect`, `aptitude` stb.) érdemes elindítani, és a következő csomagokat telepíteni: *locales, icu-locales, ispell, manpages-hu, ihungarian, myhungarian, freefont, xfonts-scalable-nonfree, xfonts-100dpi, xfonts-100dpi-transcoded, xfonts-75dpi, xfonts-75dpi-transcoded, xfonts-base, xfonts-base-transcoded,* valamint általában az *xfonts-biznet* csomagokat, különös tekintettel azokra, amelyek ISO-8859-2-re készültek. Ha rendszerünkre telepítve van az X, valószínűleg a fentebb említett csomagok közül is telepítve van már néhány. Amennyiben KDE grafikus környezetet szeretnénk használni, a *kde-i18n-hu* csomagról se feledkezzünk meg!
A telepítést az

```
apt-get install locales
➥... telep tendı csomagok ...
➥xfonts-biznet-* kde-i18n-hu
```

parancs kiadásával is megtehetjük, ha nem kívánunk csomagkezelő keretprogramot használni.
Ezt követően kedvenc szövegszerkesztőnké lesz a főszerep, amit rendszergazdaként kell használnunk. Karakteres felületen is van némi dolgunk. Ameddig a konzol által használt betűtípus nem támogatja a latin2-es, azaz az iso-8859-2 kódolású karaktereket, ne is ringassuk magunkat abban a hitben, hogy gyönyörű ékezetes betűink lesznek. Ezen a gondon két módon segíthetünk: az első a `fonty` csomag telepítése, a másik, hogy a kívánt változtatásokat kézzel készítjük el. Ehhez nyissuk meg szerkesztésre a */etc/console-tools/config*-ot, majd minden sor elé tegyünk #-ket, és illesszük be a következő sort:

```
SCREEN_FONT=iso02g
```

A következő lépésben a nyelvi változókat állítjuk magyarra. Itt először szintén a csomagokat érdemes használni, a *locales* telepítése közben a rendszer jó esetben felkészíti a gépet a magyar környezetre.
Ha a beállításokat kézzel kívánjuk megtenni, a következő főbb lépéseket kövessük:
A `LANG` és az `LC_ALL` változókat `"hu"` vagy `"hu_HU"` értékre kell állítanunk. Próbáljuk ki mindkettőt, rendszerre válogatja, hogy melyik jó. Az esetek többségében a `"hu_HU"` szokott tökéletes eredményt adni. Mivel a Woodyban az utóbbi időben rendszeresen nem találom a */etc/environment* állományt, a biztonság kedvéért létrehozom a következő tartalommal:

```
LANG=hu_HU
```

A */etc/profile* állományba illesszük be a következő sort:

```
export LC_ALL=hu_HU
```

A `loadkeys hu` parancs segítségével tölthetjük be a magyar billentyűkiosztást akkor, ha a *locales* telepítésekor nem kértük, hogy rendszerindításkor is ez a kiosztás éljen.
A */etc/locale.gen* állományban tárolja a rendszer, hogy milyen környezeteket kell elkészítenie:

```
en_US ISO-8859-1
hu_HU ISO-8859-2
```

Ha a fájlban (vagy a rendszer fontosabb elemeiben) valamit megváltoztattunk, a gépet a `locale-gen` paranccsal utasíthatjuk a környezet újbóli elkészítésére. Végezetül a */etc/inputrc*-be is érdemes belekukkantanunk. Az `input-meta` és az `output-meta` bekapcsolásával engedélyezzük az egyedi karakterek (ő, ű stb.) közvetlen használatát. A `convert-meta` változót akkor érdemes kikapcsolnunk, ha az AltGr gomb nem működik megfelelően:

```
set convert-meta off
```

Ezután máris magyar nyelven élvezhetjük rendszerünket, és ékezetes karaktereinket is használhatjuk.
A KDE grafikus környezet a fentiek ellenére sem lesz mindig önműködően magyar, hiszen ott a megfelelő nyelvet a KDE vezérlőközpontjában, a *Personalization -> Country & Language* menüpont alatt kell kiválasztanunk.

Mutt

Magyarítási folyamatunk következő lépéseként következzék egy sokak számára rendkívül fontos program.
A Mutt levelező hihetetlenül okos, így most nem is kívánok elmélyedni a lehetőségek ismertetésében, csupán a */etc/Muttrc* állományban javasolnék néhány kiegészítést:

```
set charset="iso-8859-2"
charset-hook iso-8859-1 iso-8859-2
set send_charset="us-ascii:iso-8859-
2:iso-8859-1:utf-8"
```

A `vi` szövegszerkesztőtől idegenkedőknek javaslom a sor begépelését:

```
set editor="mcedit"
```

Gibizer Tibor (gibzo@linuxmania.hu)
Újságíró, immár hét éve a Linux elkötelezett híve. Imádja a kutyákat, a kerékpározást és az autós csavargást.

Rendezvénynaptár 2003

Nyakunkon az új esztendő, és mint mindig, most is igyekszünk az év folyamán megrendezendő fontosabb eseményeket jó előre betervezni, feljegyezni, majd ugyanazzal mozdulattal elfelejteni. Társlapunk, a Linux Journal rendszeresen közli az Államok területén megrendezésre kerülő linuxos összejöveteleket, bemutatókat. Ezt a szokást kívánjuk mi is bevezetni, remélve, hogy saját kis „naptárunkkal" is hozzájárulhatunk, hogy minél több és minél látogatottabb GNU- vagy Linux-közeli konferencia, bemutató, vagy más érdekesség kapjon szárnyra.
A Rendezvénynaptár sorozat beköszönő részében megjelentetjük a tengerentúl rendezvényeit is, remélve, hogy idővel lényegesen több magyar megmozdulásról adhatunk le hírt (ha valaki tud egyéb rendezvényekről, kérjük, írja meg nekünk!).

Január

LinuxWorld Expo
Időpont: január 21–24.
Helyszín: New York

Consumer Electronics Show
Időpont: január 9–12.
Helyszín: Las Vegas

Március

CeBit
Időpont: március 12–19.
Helyszín: Hannover

Game Developers Conference
Időpont: március 4–8.
Helyszín: San Jose

Usenix Symposium on Internet Technologies
Időpont: március 26–28.
Helyszín: Seattle

Május

Usenix Mobile Systems, Applications & Services Conference
Időpont: május 5–8.
Helyszín: San Francisco

Június

Usenix Annual Technical Conference
Időpont: június 9–14.
Helyszín: San Antonio

Augusztus

LinuxWorld Expo
Időpont: augusztus 4–7.
Helyszín: San Francisco

Usenix Security Symposium
Időpont: augusztus 4–8.
Helyszín: Washington DC

Szeptember

PC Expo
Időpont: szeptember 16–18.
Helyszín: New York

INFOmarket
Időpont: szeptember 5–11.
Helyszín: Budapest, Hungexpo

Október

System Administration Conference – LISA
Időpont: október 26–31.
Helyszín: San Diego

November

INFOtrend
Időpont: november 27–29.
Helyszín: Budapest, Hungexpo

Usenix System Administration Conference
Időpont: november 3–8.
Helyszín: Philadelphia

Comdex
Időpont: november 22.
Helyszín: Las Vegas

December

Annual Computer Security Applications Conference
Időpont: december 19.
Helyszín: Las Vegas

Ők mondták

Az újításnak mindenki ellensége, aki a régi rendszerben sikeres volt, és csak csekély támogatást várhat azoktól, akik az új rendszerben lennének sikeresek. Közömbös viselkedésük oka részben a félelem, részben az, hogy általában véve hitetlenek, sosem bíznak igazán az új dolgokban, hacsak saját maguk ki nem próbálták. *(Nicolo Machiavelli)*

A Microsoft úgy döntött, hogy a nyílt forrás elleni küzdelmet vallásháborúba fordítja át. Ezzel azonban csak még élesebb megvilágításba helyezte: most már azok az informatikai vezetők is fontolgatják az áttérést, akiknek korábban eszébe sem jutott volna. *(Dan Kusnetsky, International Data corp.)*

Néhány ember olyannak látja a dolgokat, amilyenek a valóságban, és azt kérdezi: „Miért?" – Én viszont nem létező dolgokról álmodozva kérdem: „Miért is ne?" *(George Bernard Shaw)*

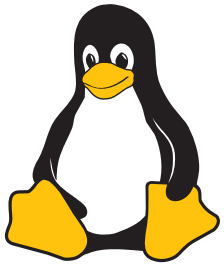
Az ember az a csomó, amelyben a kapcsolatok hálója összefut. Csak ezek a kapcsolatok számítanak az életben. *(Antoine De Saint-Exupéry)*

Kételkedő vagyok, de nem azért, mert semmiben sem hiszek, hanem mert mindent lehetségesnek tartok. *(Popper Péter)*

Linux Journal 2002. december, 104. szám

Linux-totó eredményhirdetés

A visszaérkezett kérdőívek alapján kisorsolt első három helyezett:
1. *Szűcs Tamás*, Budapest (nyereménye: 1 db 5000 Ft-os könyvutalvány, 1 db Linuxvilág-póló, 1 db Linuxvilág-poszter)
2. *Szalánki Ferenc*, Debrecen (nyereménye: 1 db Linuxvilág-póló, 1 db pingvin – Tux, 1 db Linuxvilág-poszter)
3. *Hirth Tibor*, Bácsalmás (nyereménye: 1 db Linuxvilág-póló, 1db Linuxvilág-bögre, 1 db Linuxvilág-poszter)
A nyerteseknek gratulálunk, nyereményeiket postai úton juttatjuk el. A többi nyertest elektronikus levélben értesítjük.



Üzleti lehetőség az azonosításban?

Doc azt javasolja, hogy a nyílt vevőazonosítási szabványt használjuk a távközlési és tartalomszolgáltatói iparágak megteremtésére.

Egy az AT&T-nél széles körben terjesztett belső dokumentumban (➲ http://www.rageboy.com/stupidnet.html) ***David Isenberg*** a következőt írta: „Az Internet szakít a telefontársasági modellel, ugyanis az ellenőrzést átadja a végfelhasználónak, mégpedig azáltal, hogy kiveszi a képből a rejtett hálózati részleteket."
Írása miatt David elveszítette az állását az AT&T-nél. De igaza volt. A Hálózat közvetítő nélkülisége számtalan új hatékony módszert támogat, amelyek észrevétlenek maradnak, hiszen a háttérben zajlanak.
A GXS elnök-vezérigazgatója, ***Harvey Seegers*** elmondta nekem, hogy a Hálózat erőegyensúlyt hozott létre a kínálat és a kereslet között. Szavai szerint régebben a GE uralta a vevőihez fűződő kapcsolatokat, de a Hálózat színre lépésével ennek vége szakadt. Azt tapasztalják, hogy többé nem nyomhatják le erőből a vevőiket – és ez tetszik nekik, mert most sokkal egészségesebb kapcsolatot építenek ki velük.
A Cluetrain (➲ http://www.cluetrain.com) megjelenése után sok más nagy cégtől is ugyanezt hallottuk, például a Wal-Marttól, a Johnson & Johnsontól, a Prudentialtól és az Ericssontól. De gyanús csöndet tapasztaltunk azoknak a cégek háza táján, amelyek révén az Internet ténylegesen működik. A Verizon, az AT&T, a Qwest, a Time Warner Cable, a Comcast, a Cox és az SBC mind megpróbálnak internetnyelven beszélni, de továbbra is úgy tűnik, mintha arról álmodoznának, hogy az értékes tartalmakat közvetítő csövek végén található előfizetéses csapok a kezükben lennének. Ez az álom rémálom, mióta a távközlési cégek széles sávú szolgáltatásaikat ugyanazon egyenlőtlen erőeloszlás alapjaira építik fel, mint amit a fogyasztói marketing alakított ki.
A szemükben az Internet is csak ugyanolyan volt, mint a televízió: a nagyok szólnak a kicsikhez, kevesen a sokakhoz. A szórakoztatóipar nagy „tartalomközlő" cégei is a legkevésbé azt akarták látni, hogy a buta fogyasztók milliárdjaiból okos vásárló legyen, vagy ami még rosszabb: okos szolgáltató.
De hosszú távon (és ez nagyon hosszú táv is lehet) az Internet meg fogja verni a nagyokat. És amikor ez megtörténik, a távközlési és tartalomszolgáltató cégek újra szembekerülnek ugyanazzal a kérdéssel, mint 1995-ben: hogyan kereshetünk itt pénzt?
A közvetítőnélküliségről folytatott vita nem vezet célhoz a kongresszusnál, és valószínűleg az üzleti életben sem lesz eredményes – a valóságba ültetett megoldásokra van szükség. Én az azonosítást (ID) javaslom. A legkülönfélébb üzleti lehetőségek nyílnak meg, amint az azonosítási protokollok készek a rájuk alapuló üzleti találmányok támogatására.
Az azonosítás jelentőségének felméréséhez figyeljük meg saját azonosítóink, más szóval az üzleti világban bennünket képviselő személyiségek kérdését. Ezek közül mennyit kapunk külső szervezetektől? Hány olyan van, amelyet nem elsősorban mi ellenőrzünk? Mindkét esetben közel száz százalék az arányuk? Milyen új üzleti lehetőségek nyílnának meg, ha mi magunk tartanánk uralmunk alatt a bennünket képviselő azonosítókat? Mi lenne, ha azonosítónk olyan protokollok alapján működne a hálózatokkal összekötött világban, amelyek a mi oldalunknak is épp akkora hatalmat adnának, mint amilyen a termékek előállítóinál és a tranzakciók feldolgozóinál van? Milyen új üzleti lehetőségek nyílnának meg, ha azonosítókészletünknek saját API-jai lennének, amelyek megengednék, hogy eldöntsük, mi történjen személyes adatainkkal: helyünkkel, céljainkkal, a legkülönfélébb dolgokkal kapcsolatos érdeklődésünkkel? Melyek azok a már létező kereskedők, akik mindenféle új, tartalmas módszert ki tudnának találni, hogy kapcsolatba kerüljenek vásárlóikkal? Könnyű elképzelni a számtalan új üzleti lehetőséget – és azt, hogy sokkal kevesebb lesz a nem kívánt reklám, PR, hirdetés, reklámlevél és a többi bosszantó találgatás arról, hogy vajon mit szeretnénk mi, fogyasztók.
Ennek megvalósítása a célja a ➲ http://www.PingID.com-nak immár egy éve, amióta ***Andre Durand*** megalapította.
A PingID.org egyenrangú ügyfeleknek szóló böngészőn vagy keskeny (beágyazott) ügyfélalapú nyilvános digitális azonosítási rendszeren dolgozik. Kiterjedt mértékben használja az XML, SOAP, XML-RPC és más olyan szabványokat, amelyeket a Jabber is igénybe vesz. A gyakorlati megvalósításra csak akkor születik meg az indíték, ha az emberek megpróbálják elképzelni, milyen nagyszerű módokon lehetne ezeket felhasználni. Íme néhány olyan felhasználási mód, amelyet Andre Durand gyűjtött össze:

1. azonosítási szolgáltatók (alapjában véve azonosítót tároló szolgáltatások);
2. azonosságellenőrző, illetve -igazoló szolgáltatások (tanúsítványokat, illetve aláírásokat szolgáltatnak);
3. azonosságot védő szolgáltatások (olyan cégek, amelyek azt védik, illetve ellenőrzik, hogy hogyan használják fel a rólunk szóló adatokat);
4. azonosítási módok közti átjárhatóságot biztosító szolgáltatások (olyan cégek, amelyek az egyik rendszerben már létező azonosítót egy másik rendszerben létező azonosítóvá alakítják át);
5. az azonosság hírnevén őrködő szolgáltatások (olyan cégek, amelyek segítenek megőrizni digitális azonosságunkat, és afelett őrködnek, hogy tiszta és pontos maradjon);
6. azonosítási hálózatok (ilyet épít ki a PingID hálózata).

Ha a végfelhasználóknak megfelelő hatalmat adó digitális azonosítási rendszer létrejön, a fogyasztók vásárlókká fognak továbbfejlődni. Amikor ez megtörténik, a fogyasztói viselkedés ellenőrzésével kapcsolatos érvelés vitatottá válik, és végre a közvetítő nélküliség érve nyer majd, és minden téren tarolni fog.

Linux Journal 2002. december, 104. szám

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

Erkölcs és törvényesség

Szerkesztőm nemrégiben egy zavarba ejtően nehéz kérdést tett, fel, amire igyekszem jó választ találni. A kérdés arra kényszerített hogy a jogi szempontokon kívül etikai szempontokat is megvizsgáljak. Azt kérdezte, mit tegyen egy rendszergazda, ha tudomására jut, hogy a cégénél egyesek a szerzői jogokat megsértve programokat másolnak?
A legtöbb állam jogrendje szerint senkinek sem kötelessége, hogy irgalmas szamaritánus legyen. Emlékezhetünk a bibliai példabeszédre, amelyben egy kirabolt, megvert és az út mellett elhagyatva haldokló zsidón egy arra járó szamaritánus segített, ami azért volt különösen tiszteletre méltó jócselekedet, mert a zsidók és a szamaritánusok alapvetően ellenségeknek számítottak. Jogi értelemben irgalmas szamaritánus az, aki egy bajban lévő segítségére siet akkor is, amikor ez nem követelhető meg tőle.
Mivel a rendszergazdának nem törvényes kötelessége, hogy irgalmas szamaritánus legyen, nincs semmiféle, a törvénytelen másolás megakadályozására vonatkozó jogi kötelezettsége a programot szállító cég felé.
Sok cégnél van érvényben olyan viselkedési kódex, amelyik előírja, hogy az alkalmazottnak kötelessége jelenteni a törvényszegéseket a cég vezetőinek. Amennyiben egy rendszergazda nem jelenti a programok törvénytelen másolását, megsértheti a cég szabályzatát, amivel az állását is kockáztathatja.
Mindig azt tanácsolom az alkalmazottaknak, hogy a cég saját, közzétett szabályainak megfelelően járjanak el. Ha egy cégnek nincs írott viselkedési kódexe, a rendszergazda próbálja a hallgatólagos szabályokat megfigyelni. Elnézik-e a cégnél a leplezetlen csalást? A felsőbb vezetők elítélik-e vagy bátorítják a törvénytelen magatartást?
Egyszer bepereltem egy céget egy alkalmazott nevében, akit azért rúgtak ki, mert nem volt hajlandó veszélyes hulladékot önteni a San Franciscó-i öbölbe vezető lakossági szennyvízcsatornába. Később sajnos világossá vált a számomra, hogy a cég legfelsőbb vezetői az efféle eljárást jóváhagyták és bátorították. Egy ilyen cégnél nem lenne értelme jelenteni a törvénysértéseket.
Fölösleges jelenteni, hogy a cégünk felhatalmazás nélkül másol programokat? Ha egy ilyen cégnél lennénk rendszergazdák, vajon akarnánk-e továbbra is ott dolgozni? Akarnánk-e egy olyan cégnél maradni, amelyik nyíltan támogatja a szerzői jogok megsértését?
Mi a helyzet akkor, ha maga a rendszergazda készít törvénytelen másolatokat?

> A jogi tanácsadás megfelelő kerete egy jogász-ügyfél kapcsolat, amely egy adott helyzet minden tényállását figyelembe veszi és a helyileg érvényes jognak felel meg. Bár ezt a cikket egy jogász írta, a benne foglalt adatok nem helyettesíthetik az esetre szabott, bejegyzett jogásztól származó tanácsadást.

Törvénysértő cselekedetét nem mentheti azzal, hogy a munkáltatója utasította – ha egy alkalmazott tudatában van annak, hogy megszegi a törvényt, nem háríthatja a felelősséget a munkáltatóra.
A gyakorlatban természetesen – amennyiben a törvénytelen másolás ügyében polgári peres vagy büntetőjogi eljárásra kerülne sor – a felelősök közül a legfelső vezető lenne az, akit a büntetés súlytana, nem pedig egy beosztott, aki pusztán utasításokat követ. Ennek ellenére is azt tanácsolnám a rendszergazdáknak, hogy ne kockáztassák meg a felelősségre vonást törvénysértő tettek elkövetésével. Biztosabb, ha egyszerűen visszautasítják vagy kilépnek.
Sok állam törvénye védi az olyan alkalmazottat a munkáltató válaszlépéseitől, aki megtagadja, hogy törvénysértést kövessen el. Azokat az alkalmazottakat, akik magasabb beosztású vezetőiknek vagy a megfelelő állami hatóságnak törvénysértésről tesznek jelentést, gyakran tanúvédelmi törvények védik. A bejelentést tevő alkalmazottakkal szemben fellépő céget nagy összegű kártérítés fizetésére is kötelezhetik.
De mit gondoljunk szerkesztőm kérdésének etikai oldaláról? Miért érdekel minket, ha egy cég törvénytelenül másol le egy jogdíjas számítógépprogramot? Hiszen nem vagyunk kötelesek irgalmas szamaritánusként segíteni rajtuk, akkor miért foglalkozzunk az ő gondjukkal?
Végül is a szabad és nyílt kód alapelve éppen azt a célt szolgálja, hogy a programok szabadon másolhatók, módosíthatók és terjeszthetők legyenek. Ha a jogdíjas programokat terjesztő cégek nem osztoznak az elképzeléseinkben, az az ő bajuk, nem a miénk, nem igaz? Nem vagyok meggyőződve róla, hogy ez az etikus hozzáállás.
A nyílt forrású programterjesztési modell működőképességének egyik alapja az, hogy a szerzői jogi törvények segítségével érvényt szerezhetünk a terjesztési engedélyeknek. Amikor egy cég egy GPL engedéllyel terjesztett kódot jogdíjas programmá alakít, a szerzői jogi törvény alapján beperelhetjük a szerzői jogok megsértéséért, megakadályozva, hogy a programot ne a megfelelő módon használhassa. Hogyan hagyhatnánk tehát figyelmen kívül ugyanezt a törvényt azáltal, hogy jogdíjas programokat másolunk törvénytelenül? Etikus lehet-e egy törvényre hagyatkozni akkor, amikor a saját érdekeinket védjük, és megszegni ugyanezt a törvényt, ha más érdekeiről van szó?
Ezért mindenkit arra buzdítok, hogy ***ne*** másoljon törvénytelenül jogdíjas programokat. Hajlandó vagyok a szerzői jogi törvényre hagyatkozni annak ellenére is, hogy nagy programfejlesztő cégek ugyanezt a törvényt használják fel arra, hogy ellenem versenyezzenek.
Mint etikus jogász – vagy legalábbis egy olyan jogász, aki törekszik az etikusságra – mindenkinek azt kell tanácsolnom, hogy tartsa be a törvényeket.
Ne másoljunk programokat törvénysértő módon, és ne üljünk ölbetett kézzel, ha a cégünknél mások ezt teszik.

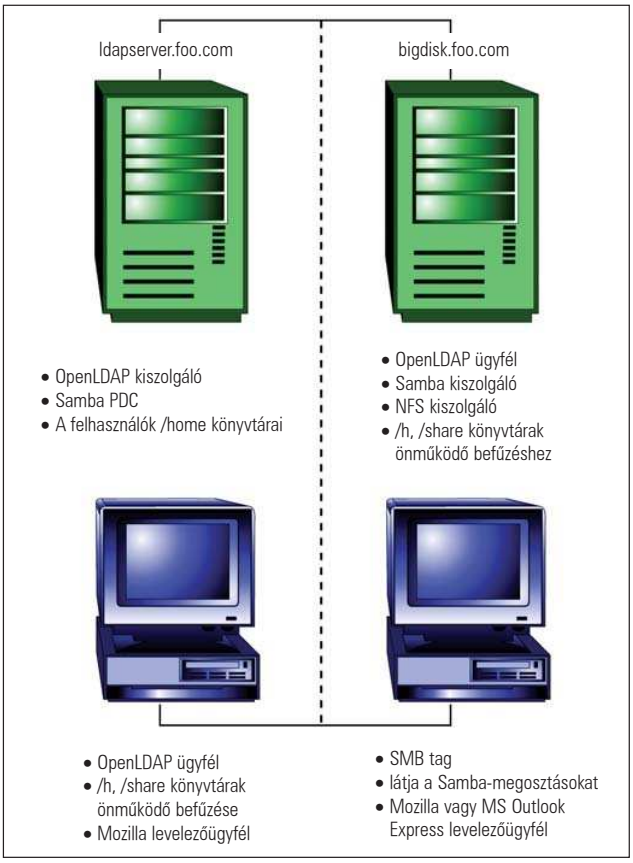
Linux Journal 2002. december, 104. szám

Lawrence Rosen
(www.rosenlav.com)
Magángyakorlatot folytató jogász. A nyílt forrás igazgatatója és jogtanácsosa.
➲ www.opensource.org

OpenLDAP mindenütt

Elektronikus leveleket tartalmazó könyvtárak megosztása, egységesített beléptetés és kevert környezetű fájlmegosztás – lépésről lépésre.

Írásom célja bemutatni, hogyan lehet az OpenLDAP-ot nem egységes környezetben működő könyvtári alapszolgáltatásként felhasználni. Az LDAP-kiszolgáló adhat osztott levélkönyvtár-szolgáltatást, egységes bejelentkezési lehetőséget a Linux- és Windows-felhasználóknak, elvégezheti a saját könyvtárak önműködő befűzését, illetve a fájlmegosztást linuxos és windowsos ügyfeleken egyaránt.
A Midwest Tool & Die immár három éve használja az OpenLDAP-ot, ami eddig kifogástalan teljesítményt nyújtott.
A könyvtárak esetében százszázalékos rendelkezésreállást tapasztaltunk. A cég vezetése szemében az első nagy előny a levélkapcsolatok könyvtárának megosztási lehetősége volt. Mára minden hálózati számítógépen egységesített beléptetéssel rendelkezünk. Számítógép-felhasználóink ugyanazt a fájltárhelyet érhetik el Windows/Samba, illetve Linux/NFS/önműködő befűzés segítségével. Eredményül végső soron a hálózati források zökkenőmentes elérését kaptuk.
A cikkben bemutatott egyszerű kevert környezet az *1. ábrán* látható. Az itt tárgyalt beállítások nem térnek ki az SSL használatára. Elképzelhető, hogy az általunk használt `ldapsync.pl` program felfedheti az LDAP üzemeltetői jelszót. A Windows-ügyfelek a felhasználói jelszavakat gyorstárazhatják, ezzel újabb linuxos biztonsági kockázatot teremtenek. Figyelmesen és megfontoltan vizsgálják meg biztonsági igényeiket és a leírt beállítást csak a saját felelősségükre használják! Sem a szerzők, sem munkáltatónk nem vállal semmilyen felelősséget az esetleg felmerülő biztonsági gondokért!



1. ábra Kevert OpenLDAP-környezet

Az LDAP-kiszolgáló telepítése és beállítása

Az itt tárgyalt LDAP-kiszolgálót RPM bináris csomagként (`openldap-2.0.11-8`) telepítettük Red Hat 7.1-es rendszeren. E csomagon kívül szükségünk lesz még az `auth_ldap` és az `nss_ldap` csomagokra is. A cikkben feltételezzük, hogy az általunk használt tartomány neve ***foo.com***. Ha a legfrissebb forrást szeretnénk használni, kövessük a
➲ http://www.openldap.org/doc/admin/quickstart.html oldalon olvasható utasításokat, töltsük le és telepítsük az OpenLDAP-ot. Javítsuk át az OpenLDAP-kiszolgáló beállítófájlját (***/etc/openldap/slapd.conf***) a következők szerint:

```
# Schemas to use
include   /etc/openldap/schema/core.schema
include   /etc/openldap/schema/cosine.schema
include
/etc/openldap/schema/inetorgperson.schema
include   /etc/openldap/schema/nis.schema
include   /etc/openldap/schema/redhat/
          ➥rfc822-MailMember.schema
include
/etc/openldap/schema/redhat/autofs.schema
include   /etc/openldap/schema/redhat/
          ➥kerberosobject.schema
database        ldbm
suffix          "dc=foo,dc=com"
rootdn          "cn=Manager,  dc=foo,dc=com"
rootpw          {crypt}sadtCr0CILzv2
directory       /var/lib/ldap

index default                                   eq
index objectClass,uid,uidNumber,gidNumber    eq
index   cn,mail,surname,givenname          eq,sub

# Access Control
access to attr=userPassword
   by self          write
   by anonymous     auth
   by dn="cn=manager,dc=foo,dc=com"      write
   by *     compare
access to *
   by self write
   by dn="cn=manager,dc=foo,dc=com"      write
   by * read
```

A bejegyzések csoportosításához használt ou eljárás

ou	bejegyzéstípus
ou=people	személyek
ou=contacts,ou=people	levélkapcsolatok
ou=group	csoportok a Linux-rendszeren
ou=auto.master	az automount /etc/auto.master parancsot helyettesíti
ou=auto.home	az automount /etc/auto.home parancsot helyettesíti
ou=auto.misc	az automount /etc/auto.misc parancsot helyettesíti

Az LDAP-sémák könyvtárbejegyzéseket alkotó objektumosztályokat és jellemzőket határoznak meg. A fenti változtatások elvégzésével az igényeinket leíró sémameghatározások nehezét már el is végeztük. A szükséges sémák, amelyeket a ***slapd.conf*** első része sorol fel, RPM-telepítésünkbe csomagolva már eleve meg vannak adva.
Amennyiben új `objectClass`-t vagy jellemzőt kellene hozzáadnunk a könyvtárhoz, olvassuk el az OpenLDAP felügyeleti útmutatót a ➲ http://www.openldap.org/doc/admin20/schema.html címen. Mi most az alapértelmezett adatbázistípust, az ldbm-et használjuk, példáink pedig az LDAP tartomány-összetevőket használják. Így aztán a ***foo.com***-ból ***dc=foo,dc=com*** válik. Továbbá a felügyelőnek (manager) teljes írási joga lesz az LDAP-bejegyzésekre.
A Red Hat 7.3 kézikönyv javaslata szerint felügyelő jelszavát érdemes a crypttel titkosítani:

```
perl -e "print crypt('jelsz ',
➥'s _karakterek',);"
```

Az előző Perl-sorban a „só_karakterek”-et cseréljük le valamilyen két karakteres sóra (eredetileg a salt: kriptográfiai kifejezés, a jelszavak kódolásakor felhasznált tetszőleges karaktereket jelenti, amelyek a kódolás „véletlenségét” biztosítják – a ford.), a jelszót pedig a jelszó egyszerű karakteres változatával. Az eredményül kapott kódolt jelszót a fentieknek megfelelő módon másoljuk a ***slapd.conf*** fájlba.
Az indexsorok a gyakran lekért jellemzők esetében növelik a teljesítményt. Az ***Access control*** (eléréskorlátozás) az `userPassword` bejegyzés hozzáférhetőségét szabályozza, de a felhasználó és a kezelő ezt a bejegyzést is módosíthatja. Az összes egyéb könyvtár esetében csak a kezelőnek van írási joga, mindenki más csak olvasási joggal rendelkezik.

A könyvtárszerkezet létrehozása

Az LDAP-ot faszerkezetként képzelhetjük el, amelynek a ***foo.com*** a törzse. Az ágakat szervezeti egységekként (`ou`, azaz organizational units) hozhatjuk létre, ahogyan azt a *2. ábrán* bemutatjuk.
A könyvtár minden egyes bejegyzését egyedileg azonosítja a megkülönböztető név (`dn`, vagyis distinguished name). Az LDAP-felügyelő `dn`-je tehát a következőképpen nézne ki: ***dn: cn=manager, dc=foo, dc=com.***
Az `ou` lehetőséget nyújt a bejegyzések csoportosítására, ahogy azt *táblázatunkban* megfigyelhetjük.
Az egyes bejegyzéseket LDIF-ben (LDAP Interchange Format, azaz LDAP csereformátum) készítettük el, majd a ***top.ldif*** fájlba mentettük:

```
dn: dc=foo, dc=com
objectclass: dcObject
objectclass: organization
o: Foo Company
dc: foo

dn: cn=manager, dc=foo, dc=com
objectclass: organizationalRole
cn: manager

dn: ou=people, dc=foo, dc=com
ou: people
objectclass: organizationalUnit
objectclass: domainRelatedObject
associatedDomain: foo.com

dn: ou=contacts, ou=people, dc=foo, dc=com
ou: contacts
ou: people
objectclass: organizationalUnit
objectclass: domainRelatedObject
associatedDomain: foo.com

dn: ou=group, dc=foo, dc=com
ou: group
objectclass: organizationalUnit
objectclass: domainRelatedObject
```

Az `ldapadd` segítségével legfelső szintű bejegyzéseinket adjuk a könyvtárhoz:

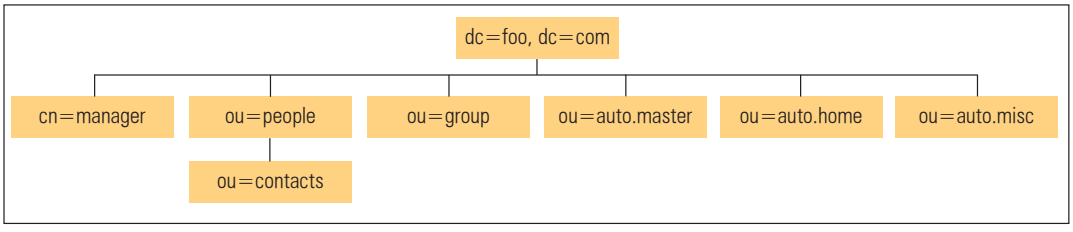
```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f top.ldif
```

Majd munkánkat az összes bejegyzés lekérésével próbáljuk ki:

```
ldapsearch -x -b 'dc=foo,dc=com'
```

Levélkapcsolatok megosztása

Immár elegendő LDAP-szerkezetet építettünk fel ahhoz, hogy ténylegesen használatba vehessük őket. Kezdetnek osszuk meg levélkapcsolatainkat, amelyeknek szintén LDIF formátumban kell lenniük.
A folyamat egyszerűsítése érdekében jó, ha címlistánkat LDIF formátumban exportálni tudjuk. Mozilla 1.0 alatt például az



2. ábra A szervezeti egységek mint az LDAP-fa ágai

Addressbook (címlista) ablak *Tools* menüjéből exportálhatunk LDIF formátumban. A Microsoft Outlook Express úgyszintén lehetővé teszi a címlisták ilyen formátumú mentését. Az eredményfájlt majd fel kell dolgoznunk, hogy úgy nézzen ki, mint az alábbi kapcsolatok példánk; e feladat megoldásához a Perlt tudnám ajánlani.
A kapcsolatokat címük egyedileg azonosítja. Íme egy példa kapcsolat dn-je:

```
dn: uid=someone@somewhere.com,ou=contacts,
    ou=people, dc=foo,dc=com
```

Az összes jellemzővel a teljes kapcsolatbejegyzés a következőképpen néz ki:

```
dn: uid=valaki@valahol.hu,ou=contacts,
    ou=people, dc=foo,dc=com
cn: Valaki Akit Ismer nk
mail: valaki@valahol.hu
uid: valaki@valahol.com
givenname: Valaki
sn: AkitIsmer nk
objectclass: person
objectClass: top
objectClass: inetOrgPerson
```

Minden egyes kapcsolatbejegyzést egy üres sorral válasszunk el egymástól, majd a fájlt ***contacts.ldif*** néven mentsük. A kapcsolatokat az `ldapadd`-dal adhatjuk a könyvtárhoz:

```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f contacts.ldif
```

Ismét ellenőrizzük a műveletet egy `ldapsearch`-csel, ami az összes bejegyzést visszakéri:

```
ldapsearch -x -b 'dc=foo,dc=com'
```

Levélügyfelek beállítása

Ideje beállítani a Mozillát, hogy az új LDAP-kiszolgálónkat használja (lásd a *3. ábrát*). A Mozilla ***Mail and News*** ablakából válasszuk az ***Edit*** menüpontot, és kattintsunk a ***Mail & Newsgroup Account Setting***-re. Az ***Addressing*** fülön válasszuk az ***Use a different LDAP server***-t, kattintsunk az ***Edit Directories***-re, majd az ***Add***-ra. Töltsük ki a ***Directory Server Properties*** űrlapot:

```
Name: FOO
Server: ldapserver.foo.com
base DN: ou=people,dc=foo,dc=com
```

Ezt követően meg kell mondanunk a Mozillának, hogy a címeket a mi könyvtárunkban keresse. A ***Mail and Newsgroups preferences*** alatt válasszuk az ***Address Autocompletion***-t, a ***Directory Server***-hez pedig írjuk be: `FOO`.
Próbáljuk ki beállításainkat; írjunk levelet egy olyan személynek, aki LDAP könyvtárunkban szerepel. A címnek a gépelés során önműködően ki kell egészülnie. A másik kipróbálási lehetőség, ha LDAP könyvtárunkban a ***Mozilla Mail Address Book***-ból keresni kezdünk. Egy olyan keresésnek, ami a ***Name*** (név) vagy az ***E-mail*** mezőben *-t (csillagot) tartalmaz, az összes kapcsolatbejegyzést vissza kell adnia. A Microsoft Outlook Express alatt az LDAP könyvtár használatát hasonlóképpen be tudjuk állítani.

Egységesített Linux-bejelentkezés LDAP-val

Ha felhasználóink bejelentkezési adatait LDAP alatt tároljuk, ugyanazt a felhasználónevet és jelszót bármely linuxos gépen használhatjuk. Első lépésben el kell döntenünk, hogy milyen felhasználóneveket akarunk bevinni az LDAP-ba. A mi felhasználói UID/GID-sémánk a következő volt:

- Rendszerazonosítók: UID < 500
- Valós emberek LDAP alatt: 499 < UID < 10 000
- Helyi felhasználók, csoportok (akik nincsenek az LDAP-ban) > 10 000

Ez a felhasználói séma 9500 LDAP-felhasználót és csoportbejegyzést engedélyez, ugyanakkor lehetővé teszi, hogy helyi, rendszerenkénti felhasználók és csoportok is létezzenek, amelyek nem kavarodnak össze az LDAP UID/GID-bejegyzésekkel.

Felhasználói bejegyzések készítése a helyi gépen

A helyi számítógép-felhasználók bejegyzéseit az `uid`-ként megadott felhasználói név azonosítja. A helyi gépfelhasználók az ***ou=people*** tagjai lesznek:

```
dn: uid=gomerp,ou=people,dc=foo,dc=com
```

A teljes bejegyzés az azonosítók elérésvezérléséhez szükséges jellemzőket is tartalmazza:

```
dn: uid=gomerp,ou=people,dc=foo,dc=com
uid: gomerp
cn: Gomer Pyle
givenname: Gomer
sn: Pyle
mail: gomer.pyle@foo.com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: kerberosSecurityObject
objectClass: shadowAccount
userPassword: useradd_ldap_flag
shadowLastChange: 11547
shadowMax: 99999
shadowFlag: 0
krbname: gomerp@FOO.COM
loginShell: /bin/bash
uidNumber: 531
gidNumber: 531
homeDirectory: /h/gomerp
gecos: Gomer Pyle
```

Az átállás megkönnyítése érdekében az OpenLDAP egy átalakítóeszközt tartalmaz, ami képes kikeresni a felhasználói adatokat – olvassuk el a ***/usr/share/openldap/migration***-t.
Először is szerkesszük át a `migrate_common.pe`-t:

```
# Default DNS domain
$DEFAULT_MAIL_DOMAIN = "foo.com";

# Default base
$DEFAULT_BASE = "dc=foo,dc=com";

# turn this on to support more general object
```

```
# classes such as person.
$EXTENDED_SCHEMA = 1;
```

Majd gyűjtsük ki a felhasználói azonosítókhoz tartozó adatokat:

```
/usr/share/openldap/migration/
➥migrate_passwd.pl /etc/passwod >people.ldif
```

Ha ez megvan, ellenőrizzük az eredményül kapott LDIF állományt. A rendszergazdához és a helyi rendszerfelhasználókhoz tartozó azonosítókat érdemes eltávolítani, hiszen ezeknek nem kell megjelenniük az LDAP alatt. Végül a felhasználói bejegyzéseket adjuk az LDAP-hoz:

```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f people.ldif
```

Mint mindig, most is ellenőrizzük munkánkat az összes bejegyzést visszaadó `ldapsearch`-csel:

```
ldapsearch -x -b "dc=foo,dc=com" "(objectclass=*)"
```

Minthogy a számítógép-felhasználók az `ou=people`-höz tartoznak, levelezőprogramunkból egyúttal a címeiket is elérhetjük.

Csoportbejegyzések készítése

Minden egyes olyan csoporthoz, ami több linuxos gép közt oszlik meg, egy-egy csoportbejegyzést kell létrehoznunk. Minden felhasználónak szüksége lesz továbbá a felhasználó saját csoportjához tartozó csoportbejegyzésre is. A csoportbejegyzéseket a `cn` azonosítja, és valamennyi csoport az ***ou=group*** alá tartozik, például:

```
dn: cn=gomerp,ou=group,dc=foo,dc=com
```

A felhasználó saját csoportja a következőképpen néz ki:

```
dn: cn=gomerp,ou=group,dc=foo,dc=com
objectClass: posixGroup
objectClass: top
cn: gomerp
userPassword: {crypt}x
gidNumber: 531*

VØg l  gy fest egy osztott csoport:

dn: cn=web_dev,ou=group,dc=foo,dc=com
objectClass: posixGroup
objectClass: top
cn: web_dev
gidNumber: 502
memberUid: gomerp
memberUid: goober
memberUid: barneyf
```

A csoportbejegyzések létrehozása után gyűjtsük ki a csoportadatokat:

```
/usr/share/openldap/migration/
➥migrate_passwd.pl /etc/group >group.ldif
```

Ellenőrizzük az eredményül kapott LDIF állományt, távolítsuk el belőle azokat a rendszercsoportokat és helyi felhasználókat, akiknek nem kell megjelenniük az LDAP-ban. Ezt követően a csoportbejegyzéseket adjuk az LDAP-hoz:

```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f group.ldif
```

Ellenőrizzük munkánkat az összes csoportbejegyzést lekérdező `ldapsearch` kéréssel:

```
ldapsearch -x -b 'dc=foo,cd=com'
```

Automount beállítása a saját könyvtárak megosztásához (és az NFS-megosztásokhoz)

Az egységesített beléptetés keretében felhasználóink egyetlen saját könyvtárral rendelkeznek, amelyet NFS-sel osztunk meg. A dolgokat egyszerűsítendő a saját könyvtárakat az ***ldapserver.foo.com***-on tároljuk, és a ***/home*** könyvtárat NFS-sel osztjuk meg. Az NFS rendszer ismertetése meghaladja e cikk kereteit, de segítségül megadunk egy rendkívül egyszerű de működőképes ***/etc/exports*** sort:

```
/home *.foo.com(rw)
```

A Linux LDAP-ügyfél a bejelentkezés során az `automount` és az NFS segítségével befűzi a felhasználó saját könyvtárát. Az LDAP alatti `automount` a NIS (Network Information Service) `automount` befűzési listákat váltja fel. Az ***auto.master, auto.home*** és ***auto.misc*** `automount` listákat fogjuk helyettesíteni. Az ***auto.master***-hez egy új szervezeti egységet is létrehozunk:

```
dn: ou=auto.master,dc=foo,dc=com
objectClass: top
objectClass: automountMap
ou: auto.master
```

Az ***auto.master*** bejegyzést a `cn` határozza meg. Az `automountInformation` kapcsoló utasítja az ***automount***-ot, hogy a listát az LDAP-ból vegye:

```
dn: cn=/h, ou=auto.master,dc=foo,dc=com
objectClass: automount
automountInformation: ldap:ou=auto.home,
                      dc=foo,dc=com
cn: /h
```

Ha már itt vagyunk, készítsünk egy ***auto.master*** bejegyzést mindjárt a másik NFS osztott könyvtárhoz is:

```
dn: cn=/share, ou=auto.master,dc=foo,dc=com
objectClass: automount
automountInformation: ldap:ou=auto.misc,
                      dc=foo,dc=com
cn: /share
```

Az ***automount*** bejegyzéseket LDIF formátumban hoztuk létre, azután ***auto.master.ldif*** néven mentjük:

```
dn: ou=auto.master,dc=foo,dc=com
objectClass: top
objectClass: automountMap
ou: auto.master
```

```
dn: cn=/h, ou=auto.master,dc=foo,dc=com
objectClass: automount
automountInformation: ldap:ou=auto.home,
                      dc=foo,dc=com
cn: /h

dn: cn=/share, ou=auto.master,dc=foo,dc=com
objectClass: automount
automountInformation: ldap:ou=auto.misc,
                      dc=foo,dc=com
cn: /share
```

Az ***auto.master*** bejegyzéseket adjuk az LDAP-hoz:

```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f auto.master.ldif
```

Következő lépésben új szervezeti egységet készítünk az ***auto.home*** számára: ***ou=auto.home*** A saját könyvtárat a `cn` adja meg:

```
dn: cn=gomerp,ou=auto.home,dc=foo,dc=com
```

Készítsünk LDIF formátumú ***auto.home*** bejegyzéseket minden egyes felhasználóhoz, és ***auto.home.ldif*** néven mentsük őket:

```
dn: ou=auto.home,dc=foo,dc=com
objectClass: automountMap
objectClass: top
ou: auto.home

dn: cn=gomerp,ou=auto.home,dc=foo,dc=com
objectClass: automount
automountInformation:
    ldapserver.foo.com:/home/gomerp
cn: super3
```

Az ***auto.home*** bejegyzéseket adjuk az LDAP-hoz:

```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f auto.home.ldif
```

Amikor a Linux LDAP-ügyfél önműködően befűzi a saját könyvtárunkat (***ldapserver.foo.com:/home/gomerp***), a ***/h/gomerp*** alá fog kerülni. Más NFS-megosztásokat is beírhatunk az LDAP-ba, amelyek aztán szükség szerint önműködően befűződnek. Ezeket az `automount` térképeket az ***ou=auto.misc*** formátumú ***auto.misc*** szervezeti egység tárolja.
A korábbiakban már elkészítettük a ***/share***-hez tartozó ***auto.master***-bejegyzést. Most az NFS-megosztásokhoz készítsük bejegyzéseket az ***auto.misc*** alatt, és ***auto.misc.ldif*** néven mentsük őket:

```
dn: ou=auto.misc,dc=foo,dc=com
objectClass: top
objectClass: automountMap
ou: auto.misc

dn: cn=redhat,ou=auto.misc,dc=foo,dc=com
objectClass: automount
automountInformation:
    bigdisk.foo.com:/pub/redhat
cn: redhat

dn: cn=engineering,ou=auto.misc,dc=foo,dc=com
objectClass: automount
automountInformation:
    bigdisk.foo.com:/data/engineering
cn: engineering
```

Az ***auto.misc*** bejegyzéseket adjuk az LDAP-hoz:

```
ldapadd -x -D 'cn=manager,dc=foo,dc=com'
➥-W -f auto.misc.ldif
```

Amikor a Linux LDAP-ügyfél osztott könyvtárunkat önműködően befűzi, a ***bigdisk.foo.com:/data/engineering*** a ***/share/engineering*** alá kerül.

Az LDAP-ügyfél beállítása a linuxos gépeken

Először is fel kell telepítenünk az `auth_ldap` azonosításkezelő csomagot, és az `nss_ldap` névváltás-szolgáltatás csomagot. A Red Hat `/usr/bin/authconfig` eszköze nagyon hasznos lehet az ügyfél beállítása során. Válasszuk ki a ***Use LDAP→Server: ldapserver.foo.com, base DN: dc=foo,dc=com*** bejegyzést. Az `authconfig` ezekbe a fájlokba fog írni: ***/etc/ldap.conf***, ***/etc/openldap/ldap.conf*** és ***/etc/nsswitch.conf***.

Ellenőrizzük, hogy a ***/etc/nsswitch.conf*** az alábbiakban látható bejegyzésekhez hasonlókat tartalmaz-e:

```
passwd:         files ldap
shadow:         files
group:          files ldap
automount:      files ldap
```

Nézzük meg, hogy a ***/etc/ldap.conf***-ban ilyen bejegyzéseket látunk-e:

```
host ldapserver.foo.com
base dc=foo,dc=com
```

illetve szerepel-e a ***/etc/openldap/ldap.conf***-ban az alábbiakhoz hasonlót

```
HOST ldapserver.foo.com
BASE dc=foo,dc=com
```

A Linux-kiszolgáló végső beállításai

Az LDAP-kiszolgáló egyben LDAP-ügyfél. Az LDAP-kiszolgálón ki kellene kapcsolnunk a ***/home*** önműködő befűzését ***/h***-ként. Az `nsswitch`-et úgy állítottuk be, hogy először a fájlokat ellenőrizze, és az önműködő befűzés adataiért csak ezután forduljon az LDAP-hoz. Ezért az ***ldapserver.foo.com:/etc/auto.master*** alatt egy álbejegyzést hozunk létre:

```
/h /etc/auto.null
```

A sajátkönyvtár-kiszolgálón a felhasználók jelszó- és csoportbejegyzéseit el kell távolítani a ***passwd*** és ***group*** fájlokból. Készítsünk mentéseket, majd szerkesszük át a ***/etc/passwd***, ***/etc/shadow***, ***/etc/group*** és ***/etc/gshadow*** fájlokat, eltávolítva az LDAP valós embereihez tartozó bejegyzéseket. Próbaképpen jelentkezzünk be a Linux LDAP-ügyfélre egy LDAP felhasználónévvel. A megfelelő felhasználóhoz tartozó bejelentkezési héjprogramot és a saját könyvtárat kell látnunk. Az ***auto.misc***-megosztások kipróbálásához a megosztást név

szerint kell elérnünk:

```
cd /share/redhat
```

Az `automount` az NFS-megosztásokat csak akkor fűzi be, amikor valaki használni akarja őket, így a */share/redhat* mindaddig nem látható, ameddig senki sem akarja elérni.

Egységesített Microsoft Windows-beléptetés Samba és LDAP segítségével

Ha egységes beléptetést szeretnénk Windowshoz és Linuxhoz, először is állítsuk be a Samba Primary Domain Controllert (PDC). A felhasználók saját könyvtárait SMB-ügyfeleken keresztül osztjuk meg. A Samba beállításainak részletei sajnos már túlmutatnak e cikk keretein.

Az ldapsync.pl és a Samba beállítása

A felhasználói jelszavakat MS Windows alól is meg lehet változtatni a Samba és a `ldapsync.pl` Perl-program segítségével, amelyet a ➲ http://www.mami.net/univr/tng-ldap/howto/#how_to_change_password címen érhetünk el.
Az `ldapsync.pl` parancsfájl a `/bin/passwd` program helyettesítésére használható, amelyet a Samba hív meg, amikor a felhasználók jelszavát megváltoztatjuk, és összehangolja őket a Samba-jelszavakkal. Az `ldapsync.pl` parancsfájlt a Samba hívja meg, amikor a felhasználó Windowsból változtatja meg a jelszavát, és ugyanúgy rendszergazdai jogosultság alatt fut, akárcsak a `/bin/passwd`, amelyet a módosítatlan Samba használ. Az LDAP által kezelt felhasználókhoz szükség lesz az `ldapsync.pl` parancsfájlra. Minthogy a felhasználók jelszavait az LDAP-ban tároljuk, és nem helyileg a */etc/passwd* fájlban, az LDAP könyvtárat és módosítókat az `ldapsync.pl` parancsfájl fogja összekapcsolni a felhasználó LDAP-beli jelszó bejegyzésével.
Könnyebben áttekinthető formában a folyamat így áll össze:

1. A felhasználó Windows alól meghívja a jelszóváltoztató programot.
2. A felhasználó rákattint a jelszóváltoztatás elfogadására, és adatot küld a Samba-kiszolgálónak.
3. A Samba megnézi a beállításfájlját, és tudja, hogy az `ldapsync.pl`-t kell az LDAP-jelszavak megváltoztatásához meghívnia.
4. Az `ldapsync.pl` a `-o %u` kapcsolóval hívódik meg, ami azt eredményezi, hogy a program nem kérdez rá a régi jelszóra. Futás közben a felhasználó nevét átadja a parancsfájlnak (ez fontos, ha a rendszergazdai jelszót nem akarjuk anélkül megváltoztatni, hogy tudnánk róla).
5. A Samba az új jelszót átadja az `ldapsync.pl`-nek, anélkül, hogy a régivel bármit is törődne.
6. Az `ldapsync.pl` beszélgetni kezd a Sambával, az új jelszót tartalmazó helyes választ várva.
7. Ha az üzenetváltás sikeres volt, a jelszót az `ldapsync.pl` kódolja.
8. Az `ldapsync.pl` ezután a felhasználó helyes dn-jéhez köti az LDAP-ot, és a felhasználó LDAP-bejegyzésén végrehajt egy ***ldapmodify***-t, lecserélve az LDAP-ban tárolt ***userPassword*** mezőt.
9. Az LDAP és a Samba még utoljára üzenetet vált, meghallgatva az LDAP sikerességének jelentését, ahol is a folyamat a végéhez ér.

A Samba ilyen beállításához a következő ***Smb.conf*** bejegyzésekre lesz szükségünk:

```
passwd program = /etc/samba/ldapsync.pl -o %u
passwd chat = *New*password*
➥%n\n*Retype*new*password* %n\n *modifying*
```

Amikor a felhasználó Windows alatt megváltoztatja a jelszavát, először meg kell adnia a régi jelszót, majd az újat, végül ellenőrzésképpen újfent be kell gépelnie az új jelszót. Mivel az `ldapsync.pl` meghívásakor nem törődtünk a régi jelszóval, csak a két új bejegyzést vizsgáljuk meg. Itt a * (csillag) arra utasítja, hogy bármit megtaláljon, amit a pontos meghatározás követ. Tehát a `*New*password*%n\n` jelentése: minden, amit a `New` szó követ, ami után bármi állhat, ezt a `password` szó követi, azután ismét akármi, végül a felhasználó által begépelt új jelszó (`%n`). A `modifying` azt jelenti, hogyha az LDAP ezt adja vissza, akkor módosította a bejegyzést, azaz a folyamat sikeres volt.
Az `ldapsync.pl`-t is át kell szerkesztenünk, hogy az LDAP kapcsolási adatokat bevihessük:

```
$binddn = "cn=manager,dc=foo,dc=com";
$passwd = "passwd";
```

Végül az `ldapsync.pl` elérését kizárólag a rendszergazdára korlátozzuk (0700).

NFS-megosztások Sambán keresztüli megosztása

Az NFS-megosztásokat is megoszthatjuk a windowsos ügyfelekkel, ha az NFS-gazdagépen Samba kiszolgálót futtatunk. A Samba-kiszolgálónak fel kell csatlakoznia a ***FOO SMB*** tartományra. Az SMB-tartományhoz való csatlakozáshoz a következő parancsot futtassuk rajta:

```
smbbpasswd -j [FOO] -r [PDC]
```

Karbantartás

Gratulálunk! Az LDAP-kiszolgáló immár működőképes az osztott levélkapcsolatokkal, egységes beléptetéssel és osztott fájltárolással rendelkezik, amelyet bármelyik ügyfélről elérhetünk. Esetleg írhatunk néhány karbantartói parancsfájlt, ami megkönnyíti a felhasználók és csoportok azonosítóinak kezelését. E feladatra ismét csak a Perlt ajánljuk.

Köszönetnyilvánítás

Az `ldapsync.pl`-t eredetileg ***Jody Haynes*** készítette a Samba-Tng-hez.

A Kapcsolódó címek a 43. CD Magazin/openLDAP könyvtárában találhatók.

Linux Journal 2002. december, 104. szám

Craig Swanson (craig.swanson@midwest-tool.com)
A Midwest Tool & Die résztulajdonosa, 1993 óta használ Linuxot. Saját kezűleg tervezte a cég hálózatát, és a programfejlesztés, illetve a termelési mérnökség (manufacturing engineering) felett atyáskodik.

Matt Lung (matt.lung@midwest-tool.com)
Hálózati mérnökként dolgozik a Midwest Tool & Die-nál. Májusban szerzett diplomát a Purdue Egyetemen. Ő állította be a cég virtuális belső hálózatát. Szeret robotokat építeni.

Folyamat-nyilvántartás

Eszközök, amelyek segítenek összegyűjteni és értelmezni rendszerünk folyamat-nyilvántartási adatait.

Mostanában, midőn az egyik Fortune 500 társaságnál jártam, fél füllel hallottam, hogy egy műszaki szolgálatos izgatottan azt suttogja projektvezetőnek: „Ne játssz semmilyen játékot a gépeden! A vállalati ellenőrök pontosan meg tudják nézni, hogy milyen programot használtál és mennyi ideig!”

A vezető, miután fennhangon biztosította a műszakist, hogy ő dolgozni szokott, és egyáltalán eszébe sem jutott játékokat játszani, elmosolyodott. Majd sokkal halkabb hangon hozzátette, hogy őt ez egyébként sem érinti, ugyanis a cég döntő részével szemben ő Linuxot, és nem Windowst használ.

Csakhogy, ha a műszakis története igaz, a vezetőt lehet, hogy mégiscsak érinti a dolog. Bár a szóban forgó nyilvántartó alkalmazások ennél a cégnél valószínűleg Windows alá voltak fejlesztve, a Linux-rendszermag beépített folyamat-nyilvántartó képességekkel rendelkezik. Ez pedig a rendszergazdának lehetővé teszi, hogy naplófájlba rögzítse a Linux-rendszeren indított valamennyi program adatait. E képesség kihasználásával mitikus hivatali ellenőrünk valóban képes lenne meghatározni, hogy ki és pontosan mennyi ideig játszott játékokat a linuxos számítógépen.

Bár erősen kétes értékű, hogy a cég megtudhatja-e, mely alkalmazottai játszanak Solitaire-t a céges felszerelésen, azért komoly indítékok is akadnak a folyamat-nyilvántartásra (Process Accounting – PA). Írásomban bemutatok majd néhány esetet, ahol a folyamat-nyilvántartás alkalmazása előnyös lehet; megmutatom, hogyan indíthatjuk be és miként használhatjuk a normál nyilvántartó parancsokat; végül azt is szemléltetem, hogyan használhatjuk a folyamat-nyilvántartás szerkezeteit és rendszerhívásait C-programokból.

Bevezető

Feltételezni fogom, hogy rendszerünkön a folyamatnyilvántartás-támogatást már befordítottuk a rendszermagba. Ezt a feltételezést azért teszem, mert, bár az általam használt valamennyi Linux-rendszer rendszermagja tartalmazta a folyamat-nyilvántartás támogatását, ettől eltérő terjesztés is előfordulhat. Ha a cikk első kódlistáját lefordítva és rendszergazdaként kapcsolók nélkül futtatva hibaüzenetet kapunk, erősen valószínű, hogy a folyamat-nyilvántartás támogatása nincs a rendszermagba építve. Ilyenkor új rendszermagot kell fordítanunk, amelyben a `CONFIG_BSD_PROCESS_ACCOUNTING` értékét ***yes***re kell állítanunk. A rendszermag újrafordítása meghaladja e cikk kereteit, segítséget a Linux Documentation Project (➲ http://www.tldp.org/HOWTO/Kernel-HOWTO.html) lapján találunk.

Erősen használt rendszereken tartsuk szem előtt, hogy a folyamat-nyilvántartás jelentős lemezterületet igényel. Az én Red Hat 7.2-t futtató Pentium III-as rendszeremen minden programvégrehajtás után 64 bájt íródik a folyamat-nyilvántartó naplófájlba.

A cikk írása közben a folyamat-nyilvántartó eszközökkel kísérletezgetve a kis lemezterülettel rendelkező próbagépen egy olyan megfigyelőfolyamatot találtam, ami minden másodpercben végrehajtódott. Azon a gépen hamar be is telt a merevelemez. Néhány kiszolgálódémon minden egyes bejövő kapcsolathoz külön folyamatot nyit. Egy ipari kiszolgálón, amelyen 25 000 folyamat indul óránként, minden hónapban körülbelül 1,1 GB folyamat-nyilvántartási adat készül. Természetesen léteznek olyan eszközök, mint az *1. táblázatban* bemutatott `accttrim` és `handleacct.sh` parancsfájl,

1. táblázat **A folyamat-nyilvántartó parancsok**

Parancs neve	Feladata
accton	Ki- vagy bekapcsolja a folyamat-nyilvántartást
acctentries	Megszámolja a napló bejegyzéseit
accttrim	Csonkolja a megadott nyilvántartási fájlt
dumpacct	Kiírja a napló tartalmát
dump-acct	Hasonló a `dumpacct`-hoz
handleacct.sh	Parancsfájl, ami tömöríti és menti a naplókat, illetve törli a legrégebbit
lastcomm	Kiírja a rendszeren végrehajtott parancsokat, a legfrissebbel kezdve
sa	Összegzi a nyilvántartást

amelyek a naplófájlokat adott időközönként lerövidítik, mentik és tömörítik. Ha elfoglalt rendszeren akarunk folyamat-nyilvántartást vezetni, fontos lehet, hogy megismerjük ezeket a programokat, és megtanuljuk őket kezelni.

Végül tudnunk kell, hogy a folyamat-nyilvántartás indításához vagy lekapcsolásához a Linux-rendszeren rendszergazdai jogosultságokkal kell rendelkeznünk, akár a szabványos parancsokat használjuk, akár saját készletet készítünk.

A folyamat-nyilvántartás előnyei

A folyamat-nyilvántartás egyik legelső felhasználási területe a számítógéptelepeken a felhasználók által lefoglalt processzoridő nyilvántartása volt, amelynek alapján aztán számlázni lehetett. A gépek elterjedtsége és a mai számítási erőforrások viszonylag alacsonyabb árai mellett ez a felhasználási lehetőség jelentéktelenné vált. Ha az osztott számítási modell végül mégis beindul, ez az alkalmazás is ismét előtérbe kerülhet.

A rendszergazdák a folyamat-nyilvántartási eszközök által összegyűjtött adatokat felhasználhatják annak megállapítására, mely programokat alkalmazzák a legtöbbször, és ennek megfelelően a rendszert az ilyen típusú programokra hatékonnyá tehetjük. Például a folyamat-nyilvántartási eszközök által összegyűjtött adatok egy része a program által ki- és beolvasott

1. lista A nyilvántartás engedélyezése és tiltása

```
/* pa.c
 * Linux bemutat  program.
 * A nyilvÆntartÆsi adatokat a parancssorban
 * megadott nevß fÆjlba napl zza.
 * Ha nem adunk meg fÆjlnevet, a folyamat-
 * nyilvÆntartÆs befejezıdik.
 */
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

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

   int rc;

   if (argc == 1)
   /* nincs paramØter - kikapcsolÆs */
   {
      printf("Turning off process
      ➥accounting.\n");
      if ( (rc = acct (NULL)) )
      {
      if (errno == ENOSYS)
      {
        printf
          ("It appears your kernel does not"
           " include accounting support\n");
      }
        perror("Problem turning off
              ➥accounting");
      return rc;
      }
   }


   else /* parancssor paramØter
           - nyilvÆntartÆs bekapcsolÆsa */
   {
      printf
       ("Attempting to log to file %s.\n",
        ➥argv[1]);
      rc =
       creat (argv[1],
            S_IRWXU | S_IRGRP | S_IROTH);

      if (rc == -1)
      {
       perror("Problem creating log file");
       return rc;
      }

      if ( (rc = acct (argv[1])) )
      {
      perror("Problem in acct() call");
      return rc;
      }

   }

   return 0;
}
```

bájtok számát, illetve a processzorhasználatot is rögzíti. Egy nagy számú, gyakori I/O-műveleteket végző alkalmazást működtető rendszert érdemes lehet olyan módon hatékonnyá tenni, ahogyan egy főként magas processzorigényű alkalmazásokat futtató rendszert eszünkbe se jutna.
Ugyanakkor elképzelhető az is, hogy a rendszergazdának két azonos feladatú szolgáltatás közül kell választania. Tegyük fel, hogy a döntés meghozatala előtt a rendszergazda látni szeretné, hogy melyik böngészőt használják ténylegesen az emberek. Mindezt megtudhatja, ha egy hétre bekapcsolja a folyamat-nyilvántartási szolgáltatást, és naplóban rögzíti az összes kiadott parancs nevét. A rendszergazdának ezután nincs más dolga, mint végigkeresni a naplót, és megállapítani, hogy melyik parancsot futtatták többször.
A folyamat-nyilvántartás legáltalánosabb felhasználási területe mégis a rendszerbiztonsági mérések támogatása. Ha betörnek a cég kiszolgálójára, a folyamat-nyilvántartási alrendszer által készített naplófájlok hasznos szolgálatot tehetnek, amikor bizonyítékokat akarunk gyűjteni. A támadó által használt programok gondos vizsgálatával megtudhatjuk, hogy nagyjából mekkora az okozott kár mértéke, milyen módszereket használt a támadó, és mik voltak a lehetséges indítékai. A folyamat-nyilvántartás naplóiból gyűjtött adatok a bíróságon is a segítségünkre lehetnek (már ha a bíróság elfogad egy bármikor hamisítható fájlt bizonyítéknak – a ford.). Ismerek egy bűnügyi esetet, ahol – minthogy a védelem ezzel nem tudott vitába szállni – ez az adat bizonyult döntőnek az ítélet meghozatalában.

Szabványos folyamat-nyilvántartó parancsok

Mégha a folyamat-nyilvántartást be is fordítottuk a rendszermagba, könnyen előfordulhat, hogy a folyamat-nyilvántartást kezelő felhasználói programok nincsenek feltelepítve a rendszerünkre. Ha ez lenne a helyzet, és gyorsan szeretnénk eredményeket elérni, keressük meg a Linux-terjesztésünkhöz mellékelt folyamat-nyilvántartó programokat.
A terjesztésünkhöz adott csomag valószínűleg úgy van elkészítve, hogy a rendszerünk beállításainak megfelelő helyre tegye a naplófájlokat, ami jelentősen megkönnyíti a telepítést. Az én Red Hat 7.2 telepítő CD-lemezeimen, a `ps-acct-6.3.2-9.i386.rpm` csomagot a második lemez ***RedHat/RPMS/*** könyvtárában találtam meg. Amennyiben a `gnorpm` grafikus telepítőeszközt használjuk, a csomag a ***Packages/Applications/System*** ágban fog megjelenni. Debian rendszeren az `acct` csomagot kell telepítenünk.
Ha forrásból telepítünk, az eszközök két változata is rendelkezésünkre áll. Az egyik BSD engedély alatt, a ➲ http://www.ibiblio.org/pub/Linux/system/admin/accounts címen érhető el. A fájlnév az `acct-1.3.73.tar.gz` alakhoz hasonlatos lesz, kisebb különbségekkel az éppen időszerű változatszám függvényében. Ahhoz, hogy ezeket az eszközöket

a rendszeremen lefordíthassam, át kellett szerkesztenem a ***lastcomm.c*** állományt, és megjegyzésbe kellett tennem a `strcpy` függvény mintapéldányát.
Létezik egy másik folyamat-nyilvántartási eszközkészlet, amelyet ***Noel Cragg*** készített, és a GNU GPL engedély alá tartozik. Ezt a ➲ http://www.gnu.org/directory/System_administration/Monitoring/acct.html címen érhetjük el.
A rendszerünkön elérhető tényleges parancsok attól függenek, hogy melyik csomagot telepítettük fel. Az 1. táblázat bemutatja, hogy milyen parancsokkal találkozhatunk, illetve hogy ezek milyen szerepet töltenek be a két csomag esetében.

A GNU nyilvántartó eszközök telepítése

Vegyük gyorsan végig a GNU nyilvántartó eszközök (GNU Accounting Utilities) telepítésének a lépéseit! Használjuk a következő parancsokat:

```
tar zxvf acct_6.3.5.orig.tar.gz
./configure
cd acct-6.3.5
make
su
make install
```

Néhány alapvető folyamat-nyilvántartási program máris elérhető a rendszerünkön. Most már elindíthatjuk a nyilvántartást és a felhasználhatjuk a programokat.

Az eszközök használata

A folyamat-nyilvántartó parancsok használatának e röpke ismertetésében két parancsra: az `accton` és a `lastcomm` utasításokra fogok kitérni. Azért választottam ezt a két parancsot, mert ezek valamennyi folyamatnyilvántartó-változatban azonosak.
Az `accton` parancs a folyamat-nyilvántartást kapcsolja ki-be. Ha a parancssorban egy fájlnevet is megadunk, akkor a folyamat-nyilvántartási adatokat az ilyen nevű naplófájlban fogja tárolni. Ha semmilyen kapcsolót nem adunk meg, a folyamat-nyilvántartás befejeződik.
Rendszerünk folyamat-nyilvántartási szolgáltatásainak indításához először is a `su` paranccsal lépjünk át rendszergazdai módba. A `touch` parancs kiadásával bizonyosodjunk meg róla, hogy a kívánt naplófájl már létezik. Például:

```
touch /var/log/pacct
```

Ezt követően gépeljük be a teljes elérési utat (általában a `/usr/sbin/accton` vagy a `/sbin/accton`) és a fájlnevet. Például:

```
/sbin/accton /var/log/pacct
```

Ezzel el is indítottuk folyamat-nyilvántartási rendszerünket. Figyeljük meg, hogy az egyes folyamatok indulásakor semmilyen adat nem kerül a naplóba; csak akkor íródik ide, amikor a folyamat befejeződik. A korábban említett projektvezető anélkül játszhatna egész nap az xbill játékkal, hogy erről bármilyen adat bekerülne a naplóba, feltéve, hogy soha nem lép ki a programból. Amikor este hazamegy, bekapcsolva hagyhatná az xbillt, esetleg lekicsinyítve az ablakot, vagy egyszerűen úgy is lekapcsolhatná a gépét, hogy kihagyja a helyes rendszerleállítást.
Ha már bekapcsoltuk a nyilvántartást, futtassunk néhány parancsot normál felhasználóként, hogy legyen némi adatunk a következő témánkhoz, a `lastcomm` parancs használatához.

Mikor végeztünk, a `su` paranccsal lépjünk vissza rendszergazdai módba, és futtassuk kapcsolók nélkül a `/usr/sbin/accton` vagy a `/sbin/accton` parancsot, kikapcsolva ezáltal a folyamat-nyilvántartást.
A `lastcomm` parancs a nyilvántartó naplófájlokban tárolt adatokat jeleníti meg, mégpedig oly módon, hogy a legfrissebb bejegyzés kerül az első helyre. Fájlnevet a `-f` parancssori kapcsolóval adhatunk meg. A rendszereken általában úgy állítják be a folyamat-nyilvántartó naplókat, hogy csak a rendszergazda legyen képes olvasni őket. Ezért a parancsot rendszergazdai módban hajtjuk végre:

```
lastcomm -f /var/log/pacct
```

A fenti parancs begépelése után az alábbiakhoz hasonló kimenetet kapunk:

```
id           root   stdin  0.00 secs Mon Jul 22
                                     ➥12:41
xauth     S  root   stdin  0.00 secs Mon Jul 22
                                     ➥12:41
xauth     S  keithg stdin  0.00 secs Mon Jul 22
                                     ➥12:41
xauth     S  keithg stdin  0.01 secs Mon Jul 22
                                     ➥12:41
bubbles   X  keithg ??     0.01 secs Mon Jul 22
                                     ➥12:33
ls           keithg ??     0.01 secs Mon Jul 22
                                     ➥12:26
bash      X  keithg ??     0.03 secs Mon Jul 22
                                     ➥08:25
```

A `lastcomm` megjeleníti minden egyes parancs nevét, a kapcsolókat, a felhasználó nevét, a használt terminált és a parancsok kilépésének idejét. A parancssorban megadhatunk egy bizonyos parancsot, felhasználót vagy terminált is. Ha éldául csak arra vagyunk kíváncsiak, mikor használták a `su` parancsot, a következőt gépeljük be:

```
lastcomm -f /var/log/pacct --command su
```

Most az alábbiakat láthatjuk:

```
su     root    ??      0.01 secs  Mon Jul 22 10:52
su     keithg  stdout  0.05 secs  Mon Jul 22 09:32
su     keithg  stdout  0.00 secs  Mon Jul 22 09:17
su     root    ??      0.00 secs  Mon Jul 22 03:29
su     keithg  tty1    0.00 secs  Sun Jul 21 19:49
```

Figyeljük meg, hogy valamennyi sorban a `su` parancsnév szerepel a bal oldali oszlopban. A programokkal és a táblázatban található egyéb programokkal kapcsolatos további adatokat a megfelelő súgóoldalakon találunk.

Programozással kapcsolatos részletek

A folyamat-nyilvántartáshoz használt `acct`-szerkezet a ***/usr/include/linux/acct.h*** és a ***/usr/include/sys/acct.h*** fejállományokban van meghatározva. A 2. *tábla* az `act`-szerkezet elemeit mutatja be, illetve röviden ismerteti az egyes elemek célját.
Ahogy azt a táblázatból láthatjuk, a 64-bájtos nyilvántartó bejegyzésbe rengeteg adat csomagolható be. Amennyiben valaki úgy véli, hogy a normál folyamat-nyilvántartás által rögzítettnél több tájékoztatóanyagra van szüksége, annak ***Mann–Mitchell***-nek a ***Kapcsolódó címek*** részben felsorolt könyvét tudnám ajánlani.

2. tábla **Az acct-szerkezet elemei**

Elem neve	Típus	Leírás
ac_flag	char	Egyedi, a folyamat viselkedését jelző zászló
ac_uid	u_int16_t	A folyamat felhasználói azonosítója (User ID)
ac_gid	u_int16_t	A folyamat csoportazonosítója (Group ID
ac_tty	u_int16_t	A folyamatot irányító terminál
ac_btime	u_int_32_t	A folyamat indítási ideje
ac_utime	comp_t	Felhasználói idő
ac_stime	comp_t	Rendszeridő
ac_etime	comp_t	Eltelt idő
ac_mem	comp_t	Átlagos memóriahasználat
ac_io	comp_t	Átvitt karakterek
ac_rw	comp_t	Olvasott és írt blokkok száma
ac_minflt	comp_t	Kisebb laphibák
ac_majflt	comp_t	Nagyobb laphibák
ac_swaps	comp_t	Csereterületek száma
ac_exitcode	u_int32_t	Folyamat kilépőkódja
ac_comm	char[]	A parancs nevének első 16 karaktere
ac_pad	char[]	Kitöltő bájtok

Példaprogramok

Az *1. lista* az `acct` rendszerhívás használatát bemutató egyszerű program. Az `acct` hívás egyetlen értéket vár: annak a fájlnak a nevét, amelyhez a folyamat-nyilvántartási adatokat hozzá kell majd fűzni. Ha az érték NULL, a folyamat-nyilvántartás befejeződik. Ezenkívül a rendszerhívás végrehajtásakor a fájlnak már léteznie kell, máskülönben a hívás sikertelen lesz és hibaüzenetet ad vissza.

Ha az egyszerű felhasználó által futtatott program megpróbálja meghívni az `acct` rendszerhívást, a kérés szintén sikertelen lesz és újfent hibaüzenetet kapunk. A folyamat-nyilvántartást ki-bekapcsolni kívánó programoknak a sikeres végrehajtáshoz rendszergazdai előjogokkal kell rendelkezniük.

Az 1. listában található kód igen hasonló az `accton` parancs általános megvalósításához, mindössze két nagyobb különbség található közöttük. Az első, hogy ez a kód a cselekedeteiről üzeneteket küld a szabványos kimenetre. A második, hogy ha az értékként megkapott fájl még nem létezik, akkor előbb létrehozzuk.

A fájl az `<unistd.h>` fejállományra hivatkozik. Az `acct` hívást használni kívánó valamennyi programnak használnia kell ezt a fájlt. A program ellenőrzi, hogy az `argc` egy volt-e, ami azt jelentené, hogy a parancssorból nem kapott kapcsolókat. Ebben az esetben a program megpróbálja kikapcsolni a folyamat-nyilvántartást, és az `acct` függvényt NULL értékkel hívja meg.

Ha a programot kapcsolóval futtatjuk, feltételezni fogja, hogy az első érték a fájlnév. Ha a fájl nem létezik, a program a `creat` rendszerhívás segítségével megpróbálja létrehozni. Ezt követően a program – a fájlnevet használva paraméternek – az `acct`-ot meghívva bekapcsolja a folyamat-nyilvántartást.

Ha a rendszerhívás hibaüzenetet ad vissza, a program egy üzenetet ír ki, majd kilép.

A *2. listában* azt mutatjuk be, hogyan kell a naplóból adatokat az `acct` memóriaszerkezetbe beolvasni, hogy az adatot később kiírathassuk vagy dolgozhassunk vele. A program használja a `<sys/acct.h>` fejállományt. Minden programnak, amely az `acct` szerkezettel szeretne dolgozni, használni kell ezt a fájlt. A `main` függvény helyi változói között találunk egy fájlmutatót, egy változót, amely a fájlból beolvasott bájtok számát tárolja, és egy `acct`-szerkezetet.

A program használójának a fájlnevet a parancssorban kell megadnia. A program a fájlt csak olvasható elérésre próbálja meg megnyitni. Amennyiben a megnyitás sikeres volt, a program a `read()` függvény segítségével közvetlenül a fájlból olvas a helyi `acct`-szerkezetbe. A cikkben helyszűke miatt most feltételezzük, hogy a `read()` egészen a fájl végének eléréséig mindig pont annyi bájtot ad vissza, amennyit kértünk. A program olvas a mezőkből folyamatosan, kiírja a megkapott parancsneveket, míg végül a `read()` hívás nullával nem tér vissza, jelezvén a fájl végét.

A cikkben bemutatott egyszerű listák csak a rendszer nyilvántartó szerkezeteinek áttekintését szolgálták. Egy komolyabb program vermet készített volna, hogy egyszerre több nyilvántartási bejegyzést is be tudjon olvasni, és például ellenőrizné, hogy nem olvastunk-e be a fájlból kevesebb bájtot, mint amennyit szerettünk volna. Ha komolyabb programpéldákat szeretnénk, nézzük meg a feltelepített nyilvántartó eszközök forrását.

Összegzés

Immár elegendő adattal rendelkezünk, hogy üzembe helyezzük a folyamat-nyilvántartást, és a szabványos parancsokat kihasználva adatokat gyűjtsünk a Linux-rendszerünkön futó programokról. Ha kellően eltökéltek vagyunk, azt is elsajátíthatjuk, hogyan készítsünk saját eszközt, ami értelmezi a folyamat-nyilvántartó naplófájlokat.

Amennyiben a folyamat-nyilvántartást biztonsági célokra használjuk fel, ne feledjük, hogy nem tökéletes megoldásról van szó, csak egyetlen apró eszközről. Valójában – ahogy arra Mann és Mitchell rámutat – a folyamat-nyilvántartás naplófájljaiból nyert adatokat fenntartással kell kezelnünk; egy megfelelően képzett támadó módosíthatja a naplót.

A linuxos folyamat-nyilvántartó eszközök használatának alapjainak áttekintése és némi gyakorlás után a saját gépünkön is üzembe helyezhetjük ezeket a szolgáltatásokat. Ha elég szerencsések vagyunk, és rendszergazdai jogosultságunk van azon a gépen, ahol dolgozunk, arra is felkészülhetünk, hogy a nyilvántartó naplófájlokból a Sokoban játék legapróbb nyomait is eltávolítsuk. Ki tudja, talán egy napon a gonosz hivatali ellenőr tényleg felbukkan az irodánkban.

A listák és a kapcsolódó címek a 43. CD Magazin/Folyamat könyvtárában találhatóak.

Linux Journal 2002. december, 104. szám.

Keith Gilbertson (keithg@kellnet.com)
Az ohioi Bowling Green State Egyetemen végzett. Programelemzőként dolgozik a May Company's Data Center vezeték nélküli eszközök és Linux-fejlesztési csapatainál az Erie-tó közelében. A tóbeli halaknak nem kell tartaniuk a pingvinektől.

Linux alapú irodai hálózat megvalósítása (1. rész)

Az irodában Linuxot használni öröm és hasznos, ráadásul ma már egyáltalán nem lehetetlen vállalkozás.

A legtöbb helyen napjainkban is tartja magát az a nézet, hogy a Linux még alkalmatlan betölteni az irodai munkaállomás szerepét. Otthonra jó – mondják sokan –, de amikor a céges felhasználásra terelődik a szó, szinte alapértelmezett feladatként azt róják egy GNU/Linux alapú kiszolgálóra, hogy mindenesként intézze a levél-, fájl- és egyéb kiszolgálást, továbbá szűrjön vírust, főzzön, mosson, takarítson, a munkaállomásokon pedig ott csücsül a rút kiskacsa, a Másik Rendszer, amivel az ember legszívesebben nem is foglalkozna, pedig sajnos mindig muszáj. Sokan nem veszik észre, hogy az évek folyamán a Rút Kispingvinből gyönyörű és nagy tudású Nagypingvin lett, és a fenti összeállítás ésszerűsége korántsem olyan egyértelmű, mint azt akár egy évvel ezelőtt is gondoltuk volna. Az elmúlt hetekben eme állítás alátámasztására tettem egy kísérletet – következzék ennek igaz története.

Munkáltatóm, az AES Cargo (nemzetközi logisztikai cég) mintegy fél éve felállította a cég új IT-fejlesztési stratégiáját, melynek része volt, hogy a cég magyarországi irodájában november 4-re (és hosszú távon a cég összes, körülbelül húsz országban található irodájában is) nemcsak a kiszolgálókon, hanem a munkaállomásokon is GNU/Linux alapú rendszer fusson (a kiszolgálókon immár két éve Debian fut). Az okok egyszerűek:

- A cég tevékenységéből fakadóan a világ minden tájáról naponta fogad leveleket, így a legújabb vírusokat általában rögtön megjelenésük napján kapjuk meg, vagyis akkor, amikor még nem létezik hozzájuk irtóprogram. (És bár ezt a gondot a csatolt állományok levélkiszolgálón való rendkívül szigorú kezelésével sikerült a legkisebbre csökkenteni, mégsem ez az igazi a megoldás.) Az első és legfontosabb ok tehát a biztonság (illetve annak hiánya a Microsoft-termékekben).
- Az eddig használt Windows-munkaállomások nem bírták az egész napos terhelést, és a cég elérte azt a méretet, amikor a Windowst futtató munkaállomások összeomlás utáni újratelepítgetése már nem bizonyult gazdaságos megoldásnak.
- A cég már említett IT-fejlesztési stratégiája tartalmaz egy SQL-re és PHP-re épülő szállítmánykövető rendszert, ami kiváltja az eddigi Windowshoz kötött megoldást, így az áttérés elől (többnyire) elhárult a programoldali akadály. (Ez alól egyelőre a könyvelés sajnos kivétel, ami amúgy is külön kérdés, de a megoldás itt is folyamatban van.)
- A számítógéppark azonos szintre történő fejlesztésével (átlagosan 900 MHz-es processzorral, 256 MB memóriával szerelt vasak) lehetővé vált egy olyan grafikus felület munkába állítása, amelyre az áttérés látható sebességcsökkenés nélkül végrehajtható, és nem is túl „fapados”.
- További lökést adott a BSA nemrégiben bejelentett „tűzszünet” akciója, illetve annak vége (mégha ők nem is ezt a hatást kívánták elérni). Ez, a Microsoft jelenlegi felhasználói szerződés-politikájával együtt, jövőbeni beszerzéseinkből minden Microsoft-terméket kizárt.

Az új rendszerrel szemben alapkövetelmény volt, hogy a felhasználóknak a lehető legkevesebb változással kelljen szembesülniük, vagyis a rendszer minél jobban hasonlítson a Windowsra, és ez – szerintem: sajnos – a kinézetre is vonatkozik, nem csupán a mindennapi használatra. Természetesen ez nem azt jelenti, hogy le kéne mondani a Linux nyújtotta többletszolgáltatásokról, sőt. Az én szempontomból nézve a legfontosabb igény az volt, hogy a munkát minél kisebb időráfordítással meg tudjam oldani, vagyis lehetőleg egyetlen rendszerlenyomat legyen, ami az összes munkaállomáson működik. Ennek megfelelően a rendszernek a következő felhasználási területeket kellett lefednie:

- Levélügyfél – az eddigi Outlook Express kiváltására.
- Irodai csomag – eddig a Microsoft Office '97-et (a finnyásabbak a 2000-t) használták.
- Internetböngésző – az eddigi Internet Explorer kiváltására.
- Címjegyzék és kapcsolat adatbázis-kezelő (CRM) – ez egy különleges terület, eddig az ACT! nevű programot alkalmazták, ami sokféle szolgáltatással bír, és ezt sose használtunk ki kellőképpen, cserébe viszont olyan adatbázis-formátummal dolgozhattunk, ami rengeteg helyet foglal, és sérülékeny.
- Hálózati erőforrások elérése – eddig természetesen Samba alapú fájl- és nyomtatómegosztást használtunk.

Néhány kevésbé fontos felhasználási terület:

- Naptár – ezt szintén az ACT! szolgáltatta.
- Szótár – a tevékenységi körből fakadóan rengeteg nyelv (az angolon, francián, oroszon kívül a japán és egyes szláv nyelvek) megértése szükséges. Ezt az igényt Windows alatt nem is voltunk képesek kielégíteni.
- „Egzotikus” billentyűzetkiosztások (számomra a franciánál egzotikusabb nemigen létezik).
- Lapolvasó használata és grafikai feldolgozás.
- A CD-jogtár használata.

Ez volt tehát az igénylista. Az első nagy kérdés az volt, hogy milyen terjesztéssel oldjam meg a feladatot. A projekt első hónapja ennek jegyében telt, a Red Hat 7.2, 7.3, Mandrake 8.1 és SuSE 7.3, 8.0 változatok kipróbálása után végül visszatértem az „igaz útra”, és a Debian mellett döntöttem. Ennek több oka is van. Linuxos életem egy kezdeti SuSE- és Red Hat-időszak után néhány éve már a Debian bűvöletében telik, ehhez értek (értek?), ezt ismerem (hadd tegyek ide még egy kérdőjelet: ?). A többi általam kipróbált terjesztéssel nem éreztem elegendőnek az ellenőrzést, amit a rendszer felett gyakorolhatok: nagyon könnyen „felmászott”, de mindig volt valami, amit utólag meg kellett volna változtatni, és a Debianon kívül másnál egyszerűen képtelen voltam rá (ez nyilván főként tudatlanságomból ered – vállalom). Egy-egy gondot okozó hálózati vagy grafikus kártya, LDAP-kínok, nyomtatás – a Debian volt az egyetlen, amelyben az összes feladatot gond nélkül és gyorsan sikerült megoldanom. Mindamellett általában is igaz, hogy a Debian-

hoz könnyebb támogatást találni (annak ellenére, hogy elvileg a SuSE „fizetős” terméktámogatása is a rendelkezésemre állt). Mivel időközben az a szerencsés helyzet állt elő, hogy éppen időszerű laptopom kilehelte nemes lelkét, én magam voltam a kipróbálás alanya, amíg beszereztem egy újat. A végső döntést a Debian Woody alapokon nyugvó próbagép ösztönözte, ám jó szokásomhoz híven (gondolván, hogy a legújabb a legjobb) a munkaállomásokra Debian SID-et terveztem telepíteni. (Ezt később egy munkanapom bánta.)
Miután döntésemet a feletteseimmel is sikerült elfogadtatni, megkezdődhetett a programok kiválasztása.

Ablakkezelő

Hosszas próbálgatás után a KDE 3 mellett döntöttem. Ez inkább vallási, mint szakmai döntés, a Gnome (illetve Gnome és valami – a továbbiakban Gnome) és a KDE közül lehet választani (mint Windows szintű ablakkezelő), nekem ez tetszett jobban. Ami azt illeti, a KDE 3 szerintem igen szép és jól használható ablakkezelő; ha nem IceWM futna mindig a saját gépemen, mindenképpen a KDE 3 lenne a kiválasztott.
A Debian Woody alá elég egyszerűen telepíthető, ha a */etc/apt/sources.list* fájlba beírjuk a következő `apt`-forrást:

```
deb http://people.debian.org/~schoepf/kde3/
➥woody ./
```

Egy `apt-get update` után csak arra kellett vigyázni, hogy ne próbáljak meg semmi olyat feltenni, ami egyszerre KDE és 2.x, mert ekkor függőségi ütközés lép fel.
A felhasználók kényelme érdekében az átállás gondnélküliséget és a kifejezett főnöki kívánalmat szem előtt tartva egy Windows XP-hez igen hasonlító téma lett az alapértelmezett (ide most írhatnék valami olyasmit, hogy fegyverrel kényszerítettek rá, de az azért túlzás lenne). Aki jobban megismerkedik a rendszer viselkedésével, úgyis átállítja magának (mint ahogy többen már meg is tették).
A KDE-ben alapértelmezett nyomtatóprotokollként a CUPS-ot választottam, mivel a nyomtatókiszolgálón is ezt használom, és sokkal többet tud, mint a jó öreg lprng–magicfilter páros, amit körülbelül három hónappal ezelőttig futtattam. A nyomtatás előtti párbeszédet (milyen nyomtató, milyen papír stb.) a QTCups program kezeli, viszont a programok alapértelmezésként általában az `lpr` paranccsal próbálnak nyomtatni, ezért lpr néven létrehoztam egy symlinket a qtcups-ra. Ez a megoldás nagymértékben megkönnyíti a többi program beállítását, nem kell külön a nyomtatóparancs szerkesztésével bajlódni.
Még egy kérdés merül fel a KDE-vel kapcsolatban: a nyelvek kezelése, a billentyűzetkiosztások, illetve a nyelvi beállítások. Ebben a KDE magasan veri a Windowst, a felhasználók nagy örömére: egyszerűen feltelepítettem a megfelelő i18n-csomagokat, innentől minden felhasználó a saját nyelvén, a saját billentyűzetkiosztásával használhatja a rendszert. Egy olyan rendszerben, ahol nemcsak magyar, de amerikai, szerb, orosz stb. felhasználók is vannak, ez nagy adomány, sokat emel a Linux elismertségén.

Levélügyfél

A cég eladdig Outlook Expresst használt, tisztán IMAP-környezetben. Az új ügyféllel szemben támasztott főbb követelmények tehát az alábbiak voltak:

- IMAP-képesség,
- több fiók egyidejű kezelése,
- HTML alapú levelek megjelenítése,
- lehetőség a csatolt állományok beágyazott megtekintésére,
- LDAP-címjegyzék használata,
- továbbá a szokásos kívánalom: ne kelljen teljesen új környezetet megszokni (sajnos a Mutt itt ki is esett).

Ezek alapján végül az Evolution levelezőprogram (a Woodyban található 1.0.5-ös változat) – a Ximian válasza az Outlookra – lett a kiválasztott (megméretett még a Mozilla, a Netscape, a KMail és a Sylpheed is). Mellette szólt az LDAP-címtárak rendkívül jó támogatása: igaz, a beállítófájlok némi kézi állítgatásával, de LDAP-ból megoldható az önműködő kiegészítés (a legújabb, 1.2-es változat már grafikusan is elérhetővé teszi ezt), továbbá ezzel gyakorlatilag egy egyszerű kapcsolat adatbáziskezelő (CRM), a határidőnapló és a naptár kérédését is letudtuk, mivel ezek mind megtalálhatók az Evolutionban.

Ezenkívül az IMAP-protokollt is ez kezelte a legjobban, bár bevallom, hogy ezen a téren azért akadnak gondok: mindenféle megszakadt kapcsolatok, valamint „broken pipe” üzenetek képében. Ezeket én a rendszer túlterheltségének tudom be (a cégnél egy-egy levélmappa mérete a 200–300 MB-ot is elérheti, a szerkezet megváltoztatása lassan elkerülhetetlenné válik), de ez nem változtat a tényen, hogy az Outlook Express jobban tudta kezelni az ilyen hibákat vagy legalábbis a felhasználó felé „barátságosabb maradt”.
Nagyon tetszik a munkatársaknak az Evolutionba épített időjárás-jelentés, a „Word of the day” és a hírek. Ezek valóban aranyosak, bár vagy a tűzfal szabályainak a megváltoztatását, vagy (mivel KDE alatt vagyunk) egy újabb sor kézi állítgatást (a proxy használatához) igényelnek.

Irodai csomag

Linux alatt irodai csomagot választani látszólag nem nehéz. Természetes választásnak tűnik az OpenOffice.org, ami szolgáltatásaiban a Microsoft Office-csomag vetélytársa. Rendkívüli lassúsága miatt mégis némi próbálkozás és keresgélés előzte meg a döntést (végül mégiscsak az OpenOffice.org mellett maradtunk). A KDE-be nagyon jól beillesztett KOffice sajnos nem megfelelő lehetőség, ugyanis nekünk (és a legtöbb felhasználónak) létfontosságú a Microsoft Office által előállított formátumok olvasásának képessége. Próbálkoztam még az Abiword–GNUmeric párossal, de ezek sem kezelik elég jól az említett formátumokat, ezért kénytelen voltam más megoldás után nézni, bármennyire is szeretem az Abiwordöt a saját gépemen. Ezután rövid ideig a Crossover Office látszott esé-

lyesnek. A Crossover nem más, mint egy módosított WINE, amit a Microsoft Office futtatásához igazítottak, és ami azt illeti, próbáink szerint a Word ezzel a megoldással gyorsabban futott, mint az egyik Windows 98-as gépen. Természetesen némi beruházás szükséges hozzá: meg kell venni magát a Crossovert, továbbá Microsoft Office felhasználói szerződésekkel is rendelkezni kell, a Microsoft szája íze szerint. Épp ez volt a legfőbb oka, hogy végül nem ezt választottam, bár némi (valószínűleg megoldható) nyomtatásbeli hiba is közrejátszott.
A végső választás végül mégiscsak az OpenOffice.org-ra esett, amihez mára szerencsére Linux alá is létezik egy gyorsítóprogram, ami a programot a memóriában tartja, így nem kell azt a röpke 30–60 másodpercet kivárni, amíg betöltődik. A megoldás neve OpenOffice.org Quick Starter, KDE és Gnome alá írt változata is létezik, a következő címekről tölthető le: ➲ http://segfaultskde.berlios.de/index.php?content=oooqs (KDE-változat) és a ➲ http://ooqstart.sourceforge.net (Gnome-változat).
Én a KDE-változatot telepítettem, aminek a cikk írása idején 0.9.5.1-es a száma, tehát még nem üzembiztos, de a gyorstöltő szolgáltatása jól működik. (Jellemző módon a Gnome-változatból ***.deb*** csomag is létezik.)
Még egy apróságot elkövettem: X alá feltelepítettem a Windows 98-ban található TrueType fontokat (amelyek még mindig szebbek, mint a többi Linux alatt elérhető font), hogy az OpenOffice.org is használni tudja őket.
A nyomtatásrésszel a már korábban létrehozott lpr -> qtcups hivatkozás révén nem volt semmi gond, talán csak a PDF-átalakítót hoztam létre itt, de még ebben sem vagyok biztos.

Internetböngésző

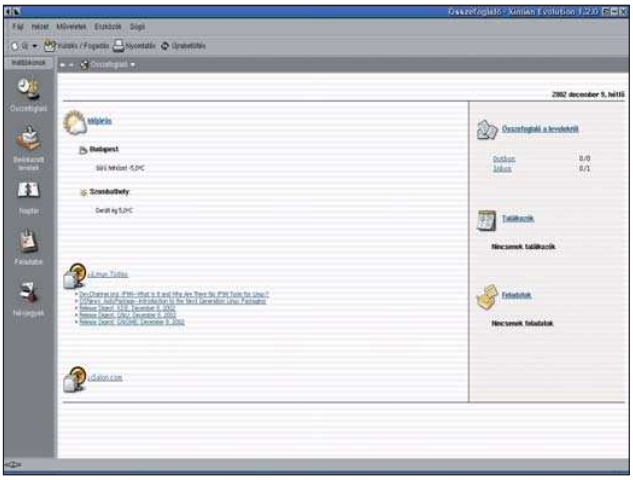
Linux alatt internetböngészőt választani meglehetősen nehéz: rengeteg lehetőség van, igazán jó viszont csak kevés. Ebben a témában végül is nem tudtam egyértelmű döntést hozni: felkerült (a KDE 3 részeként) a Konqueror 3, továbbá a Mozilla és az Opera is. Az eddigi tapasztalatok szerint nekünk az Opera a legmegfelelőbb, ezért valószínűleg érdemes lesz rá beruházni (a szerződésenkénti 29 dollár a windowsos díjakhoz képest vicc), így még a reklámokat se kell néznünk. A böngészőkkel kapcsolatos igények miatt feltelepítettem a Blackdown Java csomagot, amelyet Magyarországon egy SOTE-kiszolgáló tükröz, és APT-csomagok is léteznek belőle. A ***sources.list***-be illesztendő deb-forrás:

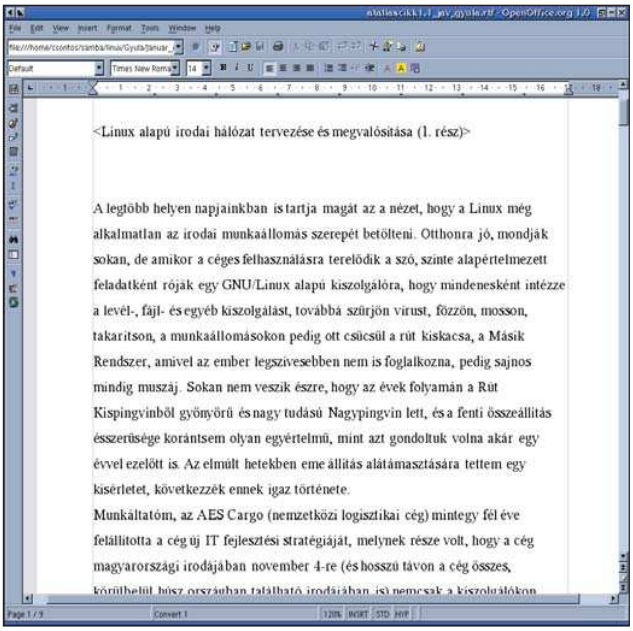
```
deb ftp://xenia.sote.hu/pub/mirrors/
➥java.blackdown.org/debian woody non-free
```

Ez a csomag felteszi a JRE1.3-at, továbbá a böngészőbővítményekhez szükséges közvetett hivatkozásokat is létrehozza. Szétnéztem még a ➲ http://plugindoc.mozilla.org/linux.html oldalon is, az itt található bővítmények közül a Flash Plugint, az Acrobat Reader bővítményt és a Pluggert tettem fel. Hiszen mint tudjuk, a kemény és eredményes irodai munka alapfeltétele a Flash.

Szótár

Erre a célra Linux alatt remek ügyfelek találhatók, és bár a téma megfejtése folyamatban van, kezdetnek feltettem a KDict nevű programot (a KDE miatt), és a ***dict.dnp.fmph.uniba.sk*** kiszolgálót állítottam be neki.
Itt mindenféle magyar és szláv szótárat meg lehet találni (sok egyéb, számunkra kevéssé használható mellett, mint például az ördöggel kapcsolatos fogalmak adatbázisa), tehát ha nem is tel-





jesen, de a Windowsnál mindenképpen jobban (és ingyenesebben) lefedtem a területet. Szóba jött még a JDictionary is (magyar fejlesztések előnyben), de tekintve, hogy egyelőre csak a magyar, az angol és a német nyelvet ismeri, várok vele.

Kapcsolók kezelése és grafikai programok

Az irodában SCSI lapolvasó (egy régebbi Umax) található. Ez nagy szerencse, így biztosan használható, és nem kell támogatás után kutatni. Kezelésére a SANE és Kooka csomagot tettem fel, a grafikus állományok kezelésére pedig a Gimpet (a KView és ImageMagick mellé). Ezzel tulajdonképpen a raszteres grafika magas színvonalon le is van fedve, a vektoros programok terén viszont a Linux sajnos el van maradva. Tekintve, hogy vektorgrafikákat nálunk főleg nyomdai használatra készítenek, és tudva, hogy a nyomdák nagy része csakis Corel-formátumokat hajlandó feldolgozni, két megoldás marad: a Windows vagy az ilyen jellegű munkák külső cégnek való kiadása.

CD-jogtár

A CD-jogtár korongján rajta van a linuxos telepítőprogram, így nem jelenthet gondot a használata.

```
mkfs /dev/hda2;mkswap /dev/hda1;tune2fs -j /dev/hda2;mkdir /mnt/sys;mkdir
/mnt/hda2;/etc/init.d/portmap start;/etc/init.d/nfs-common start;mount /dev/hda2
/mnt/hda2;mount 192.168.1.8:/mnt/sys /mnt/sys;cp /mnt/sys/system-20021104.tar /mnt/hda2;cd
/mnt/hda2;tar xvf system-20021104.tar;chroot /mnt/hda2;dmesg |less
```


A rendszer építése

A telepítés első lépése a géppark felépítésének a vizsgálata volt, mivel a cégnek nem azonos felépítésű számítógépei vannak. Különböző lapkakészletek (Intel, VIA, SiS) fordulnak elő ennek minden vonzatával, vagyis különböző típusú hang-, háló- és videokártyákkal. A gépek vásárlásánál szerencsére már nagyon régóta alapelv, hogy lehetőleg viszonylag megbízható, és Linux alatt valahogy működésre bírható eszközöket vásároljunk. Ennek fényében végül két minta rendszermagot fordítottam, processzortípusok szerint: az egyiket Pentium II/Celeron, a másikat Pentium III/Celeron II (Coppermine) alapú gépekhez. Mindkét rendszermagba belekerült (a gépek változatossága miatt) az Intel810, SiS, VIA lapkakészletek támogatása (IDE, hangvezérlő, videovezérlő, hálózati csatlakozó vonatkozásokban), a rendszermagok a processzortípuson kívül azonosak.

Az előző részben említett felhasználói programok telepítésén kívül még várt rám néhány feladat a mintarendszer elkészültté nyilvánításáig:

- Mivel LDAP távoli azonosítást használok (a beállításában nyújtott segítségért ezúton is köszönet ***Bärnkopf Bencének*** és ***Koncz Ádámnak***), be kellett állítani az LDAP-ügyfelet, illetve a PAM megfelelő (`pam_ldap`) modulját. Itt tennék említést arról, hogy a Debian SID-ben a benne levő hiba miatt az LDAP távoli azonosítás nem használható (illetve nem volt használható akkor, amikor próbáltam: azóta frissítették benne a pam-csomagokat – ki tudja, most mi a helyzet). Mire erre rájöttem, nagyjából egy teljes munkanapom ráment az ügyre. Gondolom, mindannyian ismerjük azt az egyre feszélyezőbb érzést, amikor már századszor győződünk meg a beállítások hibátlanságáról, de a dolog még mindig nem működik, ráadásul mindez hétvégén. Ezt természetesen a Woody-telepítés utáni „pöccre indulás" esete váltja ki, amikor is az ember (itt: a rendszergazda) lélekben megnemesülve tekint a fényes jövőbe…
- A felhasználók saját könyvtárai (home directory) az egyik fájlkiszolgálón vannak tárolva, és NFS-sel (azonosítás után) érhetők el a munkaállomásokról. (Ezeket a hálózati könyvtárakat természetesen indításkor be kell fűzni (`mount`).) Mivel a rendszermag és néhány a rendszernek fontos beállítóállomány kivételével a rendszer felépítése minden munkaállomáson teljesen azonos, a KDE és az egyéb programok beállítása nem okoz gondot.
- A rendszerfrissítést egy egyszerű indító parancsállomány segítségével oldottam meg: ez minden induláskor (illetve a `cron` által éjjeleként) lefut, és elindít egy a hálózaton található másik parancsfájlt, amiben természetesen már én mondom meg, hogy mi legyen. Így leegyszerűsödik az új programok telepítése (a ***.deb*** csomagokat hálózatról telepítem), az új beállítások munkába állítása: csak átszerkesztem a kiszolgálón a kis parancsállományt (általában egy mintagépen való kipróbálás után), és az ügyfelek önmagukat frissítik.

A rendszermagok és a mintarendszer elkészítése után a legnagyobb (és leghosszadalmasabb) feladat a gépek klónozásra való felkészítése, illetve maga a klónozás volt. Miután a beállítóállományokba minden lehetőséget beírtam (így a telepített gépeken csak megjegyzésbe kellett tenni őket), az egész rendszert tartalmazó lemezrészt egy ***.tar*** állományba mentettem, és NFS-sel elérhetővé tettem a hálózaton. Ezután a munkaállomások telepítése viszonylag gyorsan ment, és ebben nagy segítségemre volt a Knoppix nevű CD-ről indítható Linux-változat (ami valójában egy rendszerindításra alkalmas Debian-telepítés). Csak elindítottam a gépet a Knoppixról, és miután elindult, újra felosztottam a merevlemezeket. Ezt sajnos a merevlemezek különbözősége miatt mindig „kézzel" kellett csinálnom, mivel első lépésben fent kellett hagyni a Windowst mint választási lehetőséget (ez azzal is járt, hogy telepítés előtt a megmaradásra ítélt Windowsokat töredezettségmentesíteni kellett). Az egységesség kedvéért mindenhol ugyanaz lett a lemezrészek kiosztása: `hda1` a `csereter let`, `hda2` a rendszert tartalmazó lemezrész, `hda3` a Windows – már ahol megmaradt. Lemezfelosztás után a *listán* látható parancsokat adtam ki. Sorrendben tehát: létrehoztam a csereterületet és az `ext3` típusú lemezrészt, majd befűztem a hálózaton található könyvtárat, ahol a rendszerlenyomat található (mindig a pillanatnyilag legfrissebb, ami a legutóbbi csomagokat, esetleg az általam telepített újabb programokat, beállításokat tartalmazza), átmásoltam, kicsomagoltam, majd gyökérkönyvtárként átállítottam be az újonnan létrehozott fájlrendszert. Végül lefuttattam a `dmesg` parancsot, hogy lássam, milyen processzort és lapkakészletet tartalmaz az a gép, amit éppen telepítek. Ennek fényében átírtam a `lilo` beállítását (Pentium II vagy III rendszermag), lefuttattam a `lilo` parancsot, megjegyzésbe tettem a megfelelő lapkakészletet az X-kiszolgáló beállításában, majd átírtam a */etc/hostname*, */etc/hosts*, */etc/dhclient.conf* fájlokat, hogy az ügyfél nevét tükrözzék. Erre a távoli felügyelet érdekében van szükség (veszélyes kimenni a kiszolgálószobából, ezt mindenki tudja): az IP-címet DHCP-vel osztom a hálózaton a hálókártya MAC-címe alapján, tehát végső soron az ügyfelek egy IP-címmel vannak összerendelve.

Ha így leírva hosszadalmasnak tűnik is, maga a klónozás gépenként megközelítőleg öt percet vett igénybe, nem számítva a hálózaton való adatmozgatást és a ***.tar*** fájl kicsomagolását. Mivel az egyetlen szűk keresztmetszet azonban a hálózat, egyszerre két-három munkaállomást lehetett telepíteni. Így meglehetősen hamar megvoltam vele: azzal együtt, hogy néhány munkaállomásban CD-ROM-meghajtót kellett cserélni (a bennük lévők hibája miatt), egy hétvége – azaz nagyjából húsz munkaóra – alatt végeztem a telepítésekkel.

Idő hiányában most nem volt rá lehetőségem, de a közeljövőben egy saját telepítőkészlet összeállítását tervezem.

Lippai Gergő (lipilee@bekafarm.hu)
Munkaidőben rendszergazda, munkaidőn kívül Joseph Hellertől idéz mindenki füle hallatára, Monty Python-maratonokat szervez, countrydalokat énekel vagy Budapest utcáit rója.

Kísérletezgetés a ptrace-szel (1. rész)

A ptrace lehetővé teszi felhasználói szintű rendszerhívás-elfogó és -módosító rendszer kiépítését.

Kíváncsiskodtál már valaha, hogyan is lehet elfogni a rendszerhívásokat? Próbáltad már megbolondítani a rendszermagot a rendszerhívás értékeinek megváltoztatásával? Tettél-e már kísérletet annak megfejtésére, hogyan állítanak meg egy folyamatot a programfejlesztő eszközök, és miképp adják át a folyamat irányítását?
Ha a feladatok elvégzésére az összetett rendszermag-programozás jut eszedbe, akkor gondolkozz csak tovább. A Linux valamennyi feladat elvégzéséhez egy elegáns eszközt bocsát rendelkezésünkre, a ptrace-t (Process Trace System Call).
A ptrace olyan módszert használ, amelynek révén a szülőfolyamat egy másik folyamatot figyelhet meg és irányíthat. Ez az eszköz képes arra, hogy ellenőrizze és megváltoztassa a másik folyamat magképét és regisztereit, és elsődlegesen a töréspontok végrehajtásának ellenőrzésére, valamint a rendszerhívások nyomon követésére használják.
E cikkből azt fogjuk megtudni, hogyan kell egy rendszerhívást elfogni és a kapcsolóit megváltoztatni. A cikk második részében a haladó módszerekkel fogunk megismerkedni: az ellenőrzési pontok kijelölésével és a kódok futó programba történő beillesztésével. Bekukkantunk majd a gyermekfolyamatok regisztereibe és adatszakaszaiba, és módosítani fogjuk a tartalmukat. Meg fogjuk mutatni azt is, hogyan kell a programkódot úgy beültetni, hogy a folyamatot le lehessen állítani, és tetszőleges utasítást lehessen végrehajtani.

Alapismeretek

Az operációs rendszerek szolgáltatásaikat szabványos módszereken, úgynevezett rendszerhívásokon keresztül ajánlják fel. Szabványos alkalmazói program felületet (API) kínálnak a készülékek kezelésére, valamint alacsony szintű szolgáltatásokat, ilyenek például az állományrendszerek. Amikor egy folyamat rendszerhívást szeretne használni, a rendszerhíváshoz szükséges kapcsolókat a regiszterekben helyezi el, és elindítja a 0x80-as lágy megszakítást. A lágy megszakítás olyan, mint valamiféle kapu, ami a rendszermag-üzemmódra nyílik, ahol a mag a kapcsolók ellenőrzése után végrehajtja a rendszerhívást. Az i386-os típusú gépeken – e cikkünkben minden programkód i386-os típusra készült – a rendszerhívás száma az `%eax`-regiszterbe kerül. Az ehhez a rendszerhíváshoz tartozó értékek pedig rendre az `%ebx`, `%ecx`, `%edx` regiszterekbe kerülnek, ebben a sorrendben. Például a `write(2, "Szia", 5)` rendszerhívás a következő utasításokká alakul át:

```
movl    $4, %eax
movl    $2, %ebx
movl    $szia,%ecx
movl    $5, %edx
int     $0x80
```

ahol a `$szia` jelsorozat a `"Szia"` karakteres állandót jelöli, vagyis szakkifejezéssel élve az egy karakteres literálra mutat. De hogyan kerül a képbe a ptrace? A rendszerhívás végrehajtása előtt a rendszermag ellenőrzi, hogy a folyamat nyomkövetés alatt áll-e. Ha igen, a rendszermag leállítja a folyamatot, és a vezérlést a nyomkövető folyamatnak adja át, hogy képes legyen ellenőrizni és módosítani az ellenőrzött folyamat regisztereit. Hogy megértsük a folyamat működését, világítsuk meg egy példával:

```
#include <sys/ptrace.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <linux/user.h>   /* A karakteres
Ælland k, pØldÆul ORIG_EAX stb. szÆmÆra */

int main()
{   pid_t child;
    long orig_eax;

    child = fork();
    if(child == 0) {
        ptrace(PTRACE_TRACEME, 0, NULL, NULL);
        execl("/bin/ls", "ls", NULL);
    }
    else {
        wait(NULL);
        orig_eax = ptrace(PTRACE_PEEKUSER,
                  child, 4 * ORIG_EAX, NULL);
        printf("A gyermekfolyamat Æltal
               ➥elind tott rendszerh vÆs:
               ➥%ld\n", orig_eax);
        ptrace(PTRACE_CONT, child, NULL, NULL);
    }
    return 0;
}
```

A program futásakor az alábbi üzenetet írja ki:
A gyermekfolyamat által elindított rendszerhívás: 11
az `ls` parancs kimenetével együtt. A 11-es rendszerhívás az `execve`, ez a gyermekfolyamat által végrehajtott első rendszerhívás. Tájékozódás végett: a rendszerhívások szám szerint megtalálhatók a ***/usr/include/asm/unistd.h*** állományban. A bemutatott példában látható, hogy a folyamat gyermekfolyamatot (child) hoz létre, s ez a gyermekfolyamat hajtja majd végre az általunk nyomon követni kívánt folyamatot. Az `exec` futtatása előtt a gyermekfolyamat az első értékkel, ami a `PTRACE_TRACEME`, meghívja a ptrace-t. Ez megmondja a rendszermagnak, hogy a folyamat nyomonkövetés alatt áll, és amikor a gyermekfolyamat végrehajtja az `execve` rendszerhívást, a vezérlést a szülőfolyamatnak (parent) adja át. A szülő értesítést vár a rendszermagtól, egy `wait()` függvényhívással. Ezt követően a szülő ellenőrzi a rendszerhívás értékeit vagy más tevékenységet folytat, mondjuk megnézi a regiszterek tartalmát.

Amikor a rendszerhívás megtörtént, a rendszermag menti az eax regiszter eredeti tartalmát, amely tartalmazza a rendszerhívás számát. Ezt az értéket a gyermekfolyamat USER (felhasználói) adatterületéről olvashatjuk ki, a ptrace első értékeként használt PTRACE_PEEKUSER-rel, ahogyan az fentebbi példánkban is szerepelt. Ha végeztünk a rendszerhívás vizsgálatával, a gyermekfolyamat folytathatja a ptrace meghívását az első értékként beállított PTRACE_CONT értékkel, ami lehetővé teszi, hogy a rendszerhívás tovább folytatódjon.

ptrace-értékek

A ptrace programot négy értékkel hívják meg:

```
long ptrace(enum __ptrace_request request,
            pid_t pid,
            void *addr,
            void *data);
```

Az első érték meghatározza a ptrace viselkedését és előírja, hogyan kell a többi értéket használni.
A hívás értéke az alábbiak egyike kell legyen:

PTRACE_TRACEME, PTRACE_PEEKTEXT, PTRACE_PEEKDATA, PTRACE_PEEKUSER, PTRACE_POKETEXT, PTRACE_POKEDATA, PTRACE_POKEUSER, PTRACE_GETREGS, PTRACE_GETFPREGS, PTRACE_SETREGS, PTRACE_SETFPREGS, PTRACE_CONT, PTRACE_SYSCALL, PTRACE_SINGLESTEP, PTRACE_DETACH.

Minden egyes hívás értelmezését a cikk további részében fogjuk megadni.

A rendszerhívás értékeinek kiolvasása

Ha a PTRACE_PEEKUSER-t használjuk a ptrace első értékeként, akkor megvizsgálhatjuk a USER terület tartalmát, ahol a regiszterek tartalma és egyéb adatok tárolása történik.
Lássunk erre is egy példát:

```
#include <sys/ptrace.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <linux/user.h>
#include <sys/syscall.h>   /* A SYS_write-hoz,
                              meg miegymÆshoz */

int main()
{   pid_t child;
    long orig_eax, eax;
    long params[3];
    int status;
    int insyscall = 0;

    child = fork();
    if(child == 0) {
        ptrace(PTRACE_TRACEME, 0, NULL, NULL);
        execl("/bin/ls", "ls", NULL);
    }
    else {
       while(1) {
          wait(&status);
          if(WIFEXITED(status))
              break;
          orig_eax = ptrace(PTRACE_PEEKUSER,
                  child, 4 * ORIG_EAX, NULL);
          if(orig_eax == SYS_write) {
             if(insyscall == 0) {
                /* Rendszerh vÆs belØpØsi
                   pontja */
                insyscall = 1;
                params[0] =
                ptrace(PTRACE_PEEKUSER,
                child, 4 * EBX, NULL);
                params[1] =
                ptrace(PTRACE_PEEKUSER,
                child, 4 * ECX, NULL);
                params[2] =
                ptrace(PTRACE_PEEKUSER,
                  child, 4 * EDX, NULL);

                printf("Ki ratÆs megh vÆsa a
                      ➥k vetkezı tartalommal:"
                        "%ld, %ld, %ld\n",
                        params[0], params[1],
                        params[2]);
                }
          else { /* Rendszerh vÆs vØge */
                eax = ptrace(PTRACE_PEEKUSER,
                        child, 4 * EAX, NULL);
                printf("Ki ratÆsi mßvelet
                      ➥visszatØrØsi k dja: "
                      "with %ld\n", eax);
                    insyscall = 0;
                }
             }
             ptrace(PTRACE_SYSCALL, child,
                   NULL, NULL);
          }
     }
     return 0;
}
```

Ez a program az alábbi kimenethez hasonlót fog mutatni:

```
ppadala@linux:~/ptrace > ls
a.out          dummy.s       ptrace.txt
libgpm.html    registers.c   syscallparams.c
dummy          ptrace.html   simple.c

ppadala@linux:~/ptrace > ./a.out
Ki ratÆs megh vÆsa a k vetkezı tartalommal:
  ➥1, 1075154944, 48
a.out          dummy.s       ptrace.txt
Ki ratÆsi mßvelet visszatØrØsi k dja: 48
Ki ratÆs megh vÆsa a k vetkezı tartalommal:
  ➥1, 1075154944, 59
libgpm.html    registers.c   syscallparams.c
Ki ratÆsi mßvelet visszatØrØsi k dja: 59
Ki ratÆs megh vÆsa a k vetkezı tartalommal:
  ➥1, 1075154944, 30
dummy          ptrace.html   simple.c
Ki ratÆsi mßvelet visszatØrØsi k dja: 30
```

Itt az írási művelethez (write) kapcsolódó rendszerhívásokat követjük nyomon, az ls pedig három rendszerhívást fog elindítani. A ptrace indításánál első értékként használt

PTRACE_SYSCALL hatására a rendszermag, minden esetben megállítja a gyermekfolyamatot, ha rendszerhívás-belépési vagy -visszatérési pont keletkezik. Ez egyenértékű a PTRACE_CONT végrehajtásával, és a következő rendszerhívási belépési vagy visszatérési pontnál való megállással. Az előző példában a PTRACE_PEEKUSER-t használtuk, hogy betekintést nyerjünk a rendszerhívás értékeibe. Amint a rendszerhívás véget ér, a visszatérési érték a %eax regiszterbe kerül, és onnan kiolvasható, amint ezt az adott példában is láthattuk.
A várakoztatási rendszerhívásban (wait call) az állapotváltozót arra használhatjuk, hogy ellenőrizzük, vajon véget ért-e már a gyermekfolyamat. Ez az ellenőrzés jellegzetes módja, mivel így megvizsgálható, hogy a ptrace állította-e le a folyamatot, vagy az maga képes volt-e sikeresen befejeződni. Az olyan makrók, mint a WIFEXITED, részletes leírása elolvasható a wait(2) man oldaláról.

A regiszterek értékeinek kiolvasása

Ha a regiszterek értékeit rendszerhívás idején vagy annak visszatérésekor szeretnénk kiolvasni, a fent bemutatott eljárás kényelmetlen lehet. Ha a ptrace első értékeként a PTRACE_GETREGS-et használjuk, akkor ez a művelet valamennyi regisztert egyetlen rendszerhívásba fogja elhelyezni. A regiszterek értékét kiolvasó programot valahogy így kell elképzelni:

```
#include <sys/ptrace.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <linux/user.h>
#include <sys/syscall.h>

int main()
{   pid_t child;
    long orig_eax, eax;
    long params[3];
    int status;
    int insyscall = 0;
    struct user_regs_struct regs;

    child = fork();
    if(child == 0) {
        ptrace(PTRACE_TRACEME, 0, NULL, NULL);
        execl("/bin/ls", "ls", NULL);
    }
    else {
       while(1) {
          wait(&status);
          if(WIFEXITED(status))
              break;
          orig_eax = ptrace(PTRACE_PEEKUSER,
                      child, 4 * ORIG_EAX,
                      NULL);
          if(orig_eax == SYS_write) {
              if(insyscall == 0) {
                 /* Rendszerh vÆs belØpØsi
                  * pontja */
                 insyscall = 1;
                 ptrace PTRACE_GETREGS,
                       child, NULL, &regs);
                 printf("Write called with "
                       "%ld, %ld, %ld\n",
                       regs.ebx, regs.ecx,
                                 regs.edx);
              }
              else { /* Rendszerh vÆs
                      * visszatØrØse */
                  eax = ptrace(PTRACE_PEEKUSER,
                         child, 4 * EAX,
                         NULL);
                  printf("Write returned "
                         "with %ld\n", eax);
                  insyscall = 0;
              }
           }
           ptrace(PTRACE_SYSCALL, child,
                  NULL, NULL);
       }
    }
    return 0;
}
```

Ez utóbbi példánk hasonlít az előzőhöz, a ptrace-nek a PTRACE_GETREGS értékkel együtt történő meghívását kivéve. Itt a <linux/user.h>-ban meghatározott user_regs_struct szerkezetet használtuk a regiszterek tartalmának kiolvasására.

Néhány érdekesség

Elérkezett az ideje annak, hogy kipróbáljunk egy kis vicces furcsaságot. A következő példában meg fogjuk fordítani a rendszerhívásnak átadott karakterláncot:

```
#include <sys/ptrace.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <linux/user.h>
#include <sys/syscall.h>

const int long_size = sizeof(long);

void reverse(char *str)
{   int i, j;
    char temp;

    for(i = 0, j = strlen(str) - 2;
        i <= j; ++i, --j) {
        temp = str[i];
        str[i] = str[j];
        str[j] = temp;
    }
}

void getdata(pid_t child, long addr,
             char *str, int len)
{   char *laddr;
    int i, j;
    union u {
            long val;
            char chars[long_size];
    }data;

    i = 0;
    j = len / long_size;
    laddr = str;
```

```
    while(i < j) {
        data.val = ptrace(PTRACE_PEEKDATA,
                          child, addr + i * 4,
                          NULL);
        memcpy(laddr, data.chars, long_size);
        ++i;
        laddr += long_size;
    }

    j = len % long_size;
    if(j != 0) {
        data.val = ptrace(PTRACE_PEEKDATA,
                          child, addr + i * 4,
                          NULL);
        memcpy(laddr, data.chars, j);
    }
    str[len] = '\0';
}

void putdata(pid_t child, long addr,
             char *str, int len)
{   char *laddr;
    int i, j;
    union u {
            long val;
            char chars[long_size];
    }data;

    i = 0;
    j = len / long_size;
    laddr = str;
    while(i < j) {
        memcpy(data.chars, laddr, long_size);
        ptrace(PTRACE_POKEDATA, child,
               addr + i * 4, data.val);
        ++i;
        laddr += long_size;
    }

    j = len % long_size;
    if(j != 0) {
        memcpy(data.chars, laddr, j);
        ptrace(PTRACE_POKEDATA, child,
               addr + i * 4, data.val);
    }
}

int main()
{
   pid_t child;

   child = fork();
   if(child == 0) {
      ptrace(PTRACE_TRACEME, 0, NULL, NULL);
      execl("/bin/ls", "ls", NULL);
   }
   else {
      long orig_eax;
      long params[3];
      int status;
      char *str, *laddr;
      int toggle = 0;

      while(1) {
         wait(&status);
         if(WIFEXITED(status))
             break;
         orig_eax = ptrace(PTRACE_PEEKUSER,
                           child, 4 *
                           ORIG_EAX, NULL);

         if(orig_eax == SYS_write) {
            if(toggle == 0) {
               toggle = 1;

               params[0] =
               ptrace(PTRACE_PEEKUSER,
                child, 4 * EBX, NULL);
               params[1] =
               ptrace(PTRACE_PEEKUSER,
                child, 4 * ECX, NULL);
               params[2] =
               ptrace(PTRACE_PEEKUSER,
                child, 4 * EDX, NULL);

               str = (char *)calloc((params[2]+1)
                      * sizeof(char));
               getdata(child, params[1], str,
                      params[2]);
               reverse(str);
               putdata(child, params[1], str,
                      params[2]);
            }
            else {
               toggle = 0;
            }
         }
         ptrace(PTRACE_SYSCALL, child, NULL, NULL);
      }
   }
   return 0;
}
```

A program az alábbi kimenetet fogja mutatni:

```
ppadala@linux:~/ptrace > ls
a.out            dummy.s          ptrace.txt
libgpm.html      registers.c      syscallparams.c
dummy            ptrace.html      simple.c
ppadala@linux:~/ptrace > ./a.out
txt.ecartp       s.ymmud          tuo.a
c.sretsiger      lmth.mpgbil      c.llacys_egnahc
c.elpmis         lmth.ecartp      ymmud
```

Ez a példaprogram a fentebb bemutatott elgondolásokat hasznosítja, sőt rajtuk kívül még néhány másikat is. Használjuk benne az adatok megváltoztatását végző PTRACE_POKEDATA értékkel kiegészített ptrace-hívásokat. Ez a program a PTRACE_PEEKDATA-val teljesen azonos módon működik, ez azonban olvassa és írja is azokat az adatokat, amelyeket a gyermekfolyamat az értékekben a rendszerhívásnak átad, miközben a PEEKDATA csak az adatok olvasására vállalkozik.

Lépésenkénti végrehajtás

A ptrace szolgáltatást biztosít a gyermekfolyamat kódjának lépésenkénti végrehajtásához.
A ptrace PTRACE_SINGLESTEP értékkel együtt történő meg-

hívása jelzi a rendszermagnak, hogy minden egyes utasításnál állítsa meg a gyermekfolyamatot, és adja át a vezérlést a szülőfolyamatnak. Az alábbi példában az éppen végrehajtás alatt álló utasítás kódjának olvasását láthatjuk, miközben rendszerhívás végrehajtása zajlik. Létrehoztam egy kisméretű, semmi különöset sem tevő példaprogramot (dummy), hogy bemutassam, mi is történik valójában, ahelyett, hogy a libc-hívásokkal kellene bajlódnunk. A következőkben a ***dummy1.s*** program listáját olvashatjuk. A program gépi nyelven íródott, fordítása a `gcc -o dummy1 dummy1.s` paranccsal történt.

```
.data
szia:
     .string "Szia vilÆg! \n"

.globl  main
main:
     movl     $4, %eax
     movl     $2, %ebx
     movl     $hello, %ecx
     movl     $12, %edx
     int      $0x80
     movl     $1, %eax
     xorl     %ebx, %ebx
     int      $0x80
     ret
```

A példaprogram, ami a fenti program lépésenkénti végrehajtását végzi, a következő:

```
#include <sys/ptrace.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <linux/user.h>
#include <sys/syscall.h>

int main()
{    pid_t child;
     const int long_size = sizeof(long);

     child = fork();
     if(child == 0) {
          ptrace(PTRACE_TRACEME, 0, NULL, NULL);
          execl("./dummy1", "dummy1", NULL);
     }
     else {
          int status;
          union u {
               long val;
               char chars[long_size];
          }data;
          struct user_regs_struct regs;
          int start = 0;
          long ins;

          while(1) {
               wait(&status);
               if(WIFEXITED(status))
                    break;
               ptrace(PTRACE_GETREGS,
                      ➥child, NULL, &regs);
               if(start == 1) {
                    ins = ptrace(PTRACE_PEEKTEXT,
                           child, regs.eip,
                           NULL);
                    printf("EIP: %lx A
                          vØgrehajtott "
                          "utas tÆs: %lx\n",
       regs.eip, ins);
               }

               if(regs.orig_eax == SYS_write) {
                    start = 1;
                    ptrace(PTRACE_SINGLESTEP,
                          child, NULL, NULL);
               }
               else
                    ptrace(PTRACE_SYSCALL, child,
                           NULL, NULL);
          }
     }
     return 0;
}
```

A program kimeneteként az alábbiakat kapjuk:

```
Szia vilÆg!
EIP: 8049478 A vØgrehajtott utas tÆs: 80cddb31
EIP: 804947c A vØgrehajtott utas tÆs: c3
```

Az utasításbájtok értelmezéséhez szükség lehet az Intel kézikönyveire. Az olyan összetettebb folyamatok lépésenkénti végrehajtása, mint amilyen a töréspontok kijelölése, gondos tervezést és összetettebb forráskódot igényel.
Sorozatunk második részében meg fogjuk vizsgálni, hogyan lehet töréspontokat kijelölni és egy már működő programba programkódokat beültetni.

A cikk mostani és második részéhez tartozó forráskódok a 43. CD Magazin/Ptrace könyvtárában találhatók.

Linux Journal 2002. november, 103. szám

Pradeep Padala (p_padala@yahoo.com)
Jelenleg a Floridai Egyetemen diplomája megszerzésén munkálkodik. Érdeklődési területei között a rácsos kiépítésű és osztott rendszerek szerepelnek. Honlapját a ➲ http://www.cise.ufl.edu/~ppadala címen lehet elérni.

KAPCSOLÓDÓ CÍMEK

Az operációs rendszer kiterjesztése felhasználói szinten: az UFO globális állományrendszer
➲ http://www.cs.ucsb.edu/projects/ufo/97-usenix-ufo.ps
A ptrace súgóoldalán„A biztonságos, felhasználói szintű, korlátozott erőforrás-igényű linuxos gép (Sandboxing)”
➲ http://csdocs.cs.nyu.edu/Dienst/Repository/2.0/Body/ncstrl.nyu_cs%2fTR1999-795/pdf

A soros illesztőprogram-réteg

Greg ismerteti az új soros illesztőprogram-réteg felületét, illetve az új soros illesztőprogramok és az egyedi soros kapuk bejegyzésének módját.

Előző két cikkemben (Linuxvilág 2002. szeptember és november) a tty-rétegről volt szó, illetve egy alapvető szolgáltatásokat nyújtó illesztőprogram készítését is ismertettem. Szó esett néhány `ioctl` függvényről, illetve a `termios` adatszerkezet értelmezéséről. Ezek a cikkek nagyszerű kiindulópontot adnak, ha beágyazott rendszeredhez új tty jellegű eszközt kell létrehoznod – például egy soros kaput. Minden új rendszer megalkotásakor az eszköz tervezője szereti valamilyen más címre helyezni a soros kaput, illetve hajlamos más UART használata mellett dönteni – sokszor pedig egyszerűen elmarad a soros kapu, és USB váltja fel. A legtöbb fejlesztő tehát teljesen új tty illesztőprogramot kénytelen az új eszközhöz készíteni, ha Linuxszal is rendesen akarja használni. Szerencsére a tty-réteg felett további rétegek helyezkednek el, amelyek segítenek elfedni bonyolultságát, illetve olyan lehetőségeket biztosítanak a fejlesztő számára, amelyek egyrészt szükségesek a soros illesztőprogramhoz, másrész jobban illeszkednek az UART vagy az USB modellhez. Ezek a rétegek a soros és az USB-soros illesztőrétegek. Ebben a cikkben a soros illesztőrétegről lesz szó, míg az USB-soros réteg egy későbbi írás témáját szolgáltatja majd.

Soros őrület

Ha megnézed az általános PC soros illesztőprogram kódját a 2.2-es vagy a 2.4-es rendszermagban (a ***drivers/char/serial.c*** fájl), egy meglehetősen összetett kódot találsz, tele `#ifdef` sorokkal, amelyek az eszköz típusától függően jutnak szerephez. Az állományt számtalanszor másolták, így kapott támogatást jó néhány olyan eszköz, amelyik nem illeszkedik az általános PC UART-eszközök sorába (ilyen például a ***serial_amba.c*** és a ***serial_21285.c*** illesztőprogram). Szerencsére néhány fejlesztő – az ARM Linux karbantartója, ***Russell King*** vezetésével – átdolgozta a soros illesztőprogramot, ami immár egy általános soros magból, és kisebb, az adott eszközökhöz igazodó illesztőprogramokból áll. Kódjuk a fő rendszermagfában valahol a 2.5.28-as változat környékén jelent meg. A cikkben szereplő példák a 2.5.35-ös változatból származnak, így nem árt ellenőrizni, hogy az általad használt rendszermagban mely dolgok változtak meg.

Soros illesztőprogram bejegyzése

A soros réteg két dolgot vár el az illesztőprogramtól: jegyezze be önmagát a soros magnál, majd jegyezze be a rendszerben a PCI-számbavétel (PCI enumeration) vagy egyéb eszközfelsorolási módszer révén megtalálható soros kapukat. Az illesztőprogram bejegyzéséhez az `uart_register_driver()` függvényt kell meghívni egy az `uart_driver` nevű adatszerkezetre irányított mutatóval. A függvény kiveszi az `uart_driver` adatszerkezetben található adatokat, és ennek alapján elvégzi a tty-réteg beállítását.

A soros illesztőprogram által az `uart_driver` adatszerkezetben keresett adatmezők a következők (*lásd a következő oldalon*):

```
struct uart_port {
  spinlock_t        lock;       /* a kapu zÆrolÆsÆra */
  unsigned int      iobase;     /* be/ki[bwl] */
  char              *membase;   /* olvasÆs/ rÆs[bwl] */
  unsigned int      irq;        /* irq szÆma */
  unsigned int      uartclk;    /* alap uart  rajel */
  unsigned char     fifosize;   /* k ldØs fifo mØrete */
  unsigned char     x_char;     /* xon/xoff karakter */
  unsigned char     regshift;   /* regiszthereltolÆs */
  unsigned char     iotype;     /* be- Øs kiviteli hozzÆfØrØs
                                   jellege */
  unsigned int      read_status_mask;
                                /* illesztıprogramt l f ggı */
  unsigned int      ignore_status_mask;
                                /* illesztıprogramt l f ggı */
  struct uart_info  *info;
                               /* mutat  a sz lı adataira */
  struct uart_icount icount;   /* statisztikÆk */
  struct console    *cons;
                    /* konzol-adatszerkezet, ha sz ksØges */
#ifdef CONFIG_SERIAL_CORE_CONSOLE
  unsigned long sysrq;          /* sysrq idıhatÆr */
#endif
  unsigned int      flags;
  unsigned int      mctrl;
                    /* pillanatnyi modemvezØrlı-beÆll tÆsok */
  unsigned int      timeout;
                    /* karakteralapœ idıhatÆr */
  unsigned int      type;       /* kapu t pusa */
  struct uart_ops   *ops;
  unsigned int      line;       /* kapuindex */
  unsigned long     mapbase;    /* be- Øs kiviteli œjra-
                                   hozzÆrendelØs cØljÆb l */
  unsigned char     hub6;
                    /* a 8250 illesztıprogramban kell
                       szerepelnie */
  unsigned char unused[3];
    };
```

```
struct uart_ops {
   unsigned int     (*tx_empty)(struct
                                uart_port *);
   void    (*set_mctrl)(struct uart_port *,
                        unsigned int mctrl);
   unsigned int     (*get_mctrl)(struct
                                uart_port *);
   void    (*stop_tx)(struct uart_port *,
                      unsigned int tty_stop);
   void    (*start_tx)(struct uart_port *,
                       unsigned int tty_start);
   void    (*send_xchar)(struct uart_port *,
                         char ch);
   void    (*stop_rx)(struct uart_port *);
   void    (*enable_ms)(struct uart_port *);
   void    (*break_ctl)(struct uart_port *,
                        int ctl);
   int     (*startup)(struct uart_port *);
   void    (*shutdown)(struct uart_port *);
   void    (*change_speed)(struct uart_port *,
                           unsigned int cflag,
                           unsigned int iflag,
                           unsigned int quot);
   void         (*pm)(struct uart_port *,
                      unsigned int state,
                      unsigned int oldstate);
   int         (*set_wake)(struct uart_port *,
                           unsigned int state);

/* A kapu t pusÆt visszaad  karakterlÆnc.*/
   const char *(*type)(struct uart_port *);

   /*A kapu Æltal hasznÆlt be- Øs kiviteli
    *mem ria-erıforrÆsok elengedØse.
    *Sz ksØg szerint a be- Øs kiviteli c mek
    *felszabad tÆsÆt is magÆban foglalja.*/
   void    (*release_port)(struct uart_port *);

   /*A kapu Æltal hasznÆlt be- Øs kiviteli
    *mem ria-erıforrÆsok lefoglalÆsa. RØsze
    *a be- Øs kiviteli c mek lefoglalÆsa is,
    *ha sz ksØges.*/
   int     (*request_port)(struct uart_port *);
   void    (*config_port)(struct uart_port *,
                          int);
   int     (*verify_port)(struct uart_port *,
                          struct serial_struct *);
   int     (*ioctl)(struct uart_port *, unsigned
                    int, unsigned long);
};
```

```
struct module   *owner;
const char      *driver_name;
const char      *dev_name;
int              major;
int              minor;
int              nr;
struct console  *cons;
```

Az ***owner*** (tulajdonos) mező egy mutató a soros illesztőprogramot birtokló modulra. Általában a `THIS_MODULE` makróra mutat.

A ***driver_name*** mező az illesztőprogram leírását tartalmazó karakterláncra mutat, ennek tartalma általában egyezik a ***dev_name*** mezőével. A ***dev_name*** mező megadásakor természetesen a `devfs` is figyelembe veendő, és a `devfs` választása esetén `%d` karaktereket kell hozzáadni a mező végéhez. Ennek oka a `devfs` eszközcsomópont létrehozási módszerében keresendő. Például az ***amba.c*** illesztőprogram az alábbi ***driver_name*** és ***dev_name*** mezőket használja:

```
   .driver_name          = "ttyAM",
#ifdef CONFIG_DEVFS_FS
   .dev_name             = "ttyAM%d",
#else
   .dev_name             = "ttyAM",
#endif
```

A major (fő) és a minor (mellék) mezők az illesztőprogram fő- és melléksorszámát tartalmazzák.

Az ***nr*** mező adja meg, hogy az illesztőprogram legfeljebb hány soros kaput támogat.

A ***cons*** mező egy mutató, a `console` adatszerkezetre mutat, és csak akkor jut szerephez, ha az illesztőprogram támogatja a soros konzol használatát. Ha az illesztőprogram nem támogat a soros konzolt, a mezőnek `NULL` értéket kell adni.

Soros kapu bejegyzése

Most, hogy a soros illesztőprogramot a soros illesztőprogram-rétegnél sikeresen bejegyezted, a soros kapukat is egyenként be kell jegyezni egy-egy `uart_add_one_port()` hívással. A függvény lefoglal egy mutatót az eredeti `uart_driver` adatszerkezetre, amelyet az `uart_register_driver` függvénynek adtál át, illetve egy másikat az `uart_port` adatszerkezetre.

Az `uart_port` adatszerkezetet az *1. lista* tartalmazza. Ezeknek az adatmezőknek a nagy részét működés közben az egyedi illesztőprogramok használják annak meghatározására, hogy az adott kapu hogyan csatlakozik a processzorhoz (`hub6`, `iobase`, `membase`, `mapbase` és `iotype` változók).

Az adatszerkezet egyik legérdekesebb változója az `uart_ops` adatszerkezet-mutató, amely a soros mag által a kapuhoz egyedileg készített illesztőprogram meghívására használt függvények listáját adja meg. Az adatszerkezet felépítését a *2. listában* láthatjuk.

Meglehetősen nagyméretű adatszerkezetről van szó, számos függvénymutatóval – legalább olyan ronda, mint a `tty_driver` adatszerkezet.

A `startup` (indítás) függvény lefuttatása minden egyes `open(2)` (megnyitás) hívásnál megtörténik. Futtatására csak az után kerül sor, hogy a soros mag elvégezte a számos erőforrás-ellenőrző művelet mindegyikét, és úgy találta, hogy a kaput valóban meg kell nyitni. A soros illesztőprogram általában megad bizonyos eszközfüggő beállításokat, hogy a kaput a függvényben használni lehessen.

A `shutdown` (leállás) függvény a `startup` ellentettje. Meghívására akkor kerül sor, ha a kapu lezárult, és több adat nem halad át rajta. Akkor történik ilyen, ha az eszközt leállítottuk, és a `startup` függvény lefutása során lefoglalt erőforrásokat fel kell szabadítani.

A `request_port` és a `release_port` függvények a soros kapu használatához szükséges memória és egyéb erőforrások lefoglalására szolgálnak. A `config_port` függvény nagyon hasonlít a `request_port` függvényre, ám akkor hívódik meg, amikor a gép végigpróbálja (autoprobe) a csatlakozó soros kapuk mindegyikét; illetve ugyanazokat az erőforrás-foglalásokat végzi el, mint a `request_port`.

A `change_speed` függvény használható a kapu vonali beállításainak módosítására. A függvénynek átadott értékek már

letisztultak az eredeti, a tty-rétegen keresztül átadott termios adatszerkezethez képest, így a soros illesztőprogram működése egyszerűbb lehet.
A vonali és kapuállapot lekérdezésére és módosítására számos függvény használható:

- set_mctrl: új értéket ad az MCR UART regiszternek.
- get_mctrl: lekérdezi az MCR UART regiszter pillanatnyi értékét.
- stop_tx: leállítja az adatküldést a kapun.
- start_tx: elindítja az adatküldést a kapun.
- tx_empty: megadja, hogy a kapu küldő része üres-e vagy sem.
- send_xchar: ***XOFF*** karakter küldésére utasítja a kaput a gép felé.
- stop_rx: leállítja az adatok fogadását.
- break_ctl: a BREAK értéket küldi át a kapun.
- enable_ms: engedélyezi a modemállapot-megszakításokat.

Két, a soros kapu energiakezelésével kapcsolatos függvény létezik: a pm és a set_wake. Ha a számítógép támogatja az energiagazdálkodási lehetőségeket, ezekkel a függvényekkel lehet kezelni az eszköz indítását és leállítását.
A verify_port használható a neki átadott értékek ellenőrzésére a tekintetben, hogy azok használhatók-e a megadott kapuhoz. Meghívása akkor történik, amikor a felhasználó TIOCSSERIAL ioctl(2) hívást végez a kapun. (A TIOCSSERIAL szerepével kapcsolatban lásd: „A tty-réteg 2. rész”, Linuxvilág 2002. november.)
A soros illesztőréteg számos általános soros ioctl kezelésére képes, úgymint TIOCMGET, TIOCMBIS, TIOCMBIC, TIOCMSET, TIOCGSERIAL, TIOCSSERIAL, TIOCSERCONFIG, TIOCSERGETLSR, TIOCMIWAIT, TIOCGICOUNT, TIOCSERGWILD és TIOCSERSWILD. Bármely más ioctl-nek a soros kapura történő meghívása esetén az uart_ops adatszerkezet ioctl hívása segítségével átadás történik az adott kapura.
Az utolsó függvény a type, ami a soros kapu leírását megadó karakterlánccal tér vissza. Ezt a */proc/tty/driver/* könyvtárban elérhető *proc* állomány használja, illetve a rendszerindító üzenetek között jelenik meg a kapu térképezés.

Hol vannak az adatok?

Bizonyára feltűnt, hogy az uart_ops adatszerkezet semmilyen függvényt nem tartalmaz adatok fogadására vagy küldésére. A felhasználó által write(2) hívással a tty-rétegnek átadott adatokat a soros réteg egy körkörös átmeneti tárba helyezi, ezt követően az egyedi uart illesztőprogram feladata, hogy kivegye és átküldje őket a kapun. Az illesztőprogram általában minden UART megszakításnál ellenőrzi a körkörös átmeneti tár tartalmát, hátha talál benne továbbításra váró adatokat. Az alábbi példafüggvény erre kínál egy lehetséges megoldást:

```
static void tiny_tx_chars(struct uart_port
*port)
{
     struct circ_buf *xmit =
                    &port->info->xmit;
     int count;

     if (port->x_char) {
          /* itt t rtØnik a port->x_char
             k ldØse a kapura */
          UART_SEND_DATA(port->x_char);
          port->icount.tx++;
          port->x_char = 0;
          return;
     }
     if (uart_circ_empty(xmit) ||
        uart_tx_stopped(port)) {
          tiny_stop_tx(port, 0);
          return;
     }
     count = port->fifosize >> 1;
     do {
          /* send xmit->buf[xmit->tail]
             kik ldØs a kapura */
          UART_SEND_DATA
           (xmit->buf[xmit->tail]);
          xmit->tail = (xmit->tail + 1) &
                         (UART_XMIT_SIZE - 1);
          port->icount.tx++;
          if (uart_circ_empty(xmit))
               break;
     } while (--count > 0);

     if (uart_circ_chars_pending(xmit)
                    <WAKEUP_CHARS)
          uart_event(port, EVT_WRITE_WAKEUP);

     if (uart_circ_empty(xmit))
          tiny_stop_tx(port, 0);
}
```

A függvény azzal indul, hogy ellenőrzi, az x_char* éppen kiküldésre vár-e. Ha igen, akkor elküldi, majd megnöveli a kapun átküldött karakterek számlálójának az értékét. Ha nem, a körkörös átmeneti tár tartalmát ellenőrzi, hogy talál-e benne adatot, illetve azt, hogy éppen nem állította-e le valami a kaput. Ha a feltételek teljesülnek, megkezdjük a karakterek kivételét a körkörös átmeneti tárból, és átküldjük őket az UART-nak. A példában legfeljebb a FIFO méretének a felét kitevő adatmennyiséget küldünk el, ami nagy átlagban megfelelő mennyiségnek mondható. A karakterek elküldése után láthatjuk, hogy elegendő karaktert vettünk-e ki a körkörös átmeneti tárból, és kérhetünk-e újabbakat. Ha igen, az EVT_WRITE_WAKEUP átadott értékkel meghívjuk az uart_event() függvényt, ezzel felkérjük a soros magot arra, hogy értesítse a felhasználói réteget a további adatok elküldésének lehetőségéről.
Minden, a soros illesztőprogram által fogadott adat a tty-réteghez kerül, pontosan úgy, mintha egy általános tty illesztőprogram tenné ezt: a tty_insert_flip_char() függvény segítségével. A művelet általában az UART megszakítási függvényen belül játszódik le.

Rövid soros példa

Az 1. kódrészlet azt szemlélteti, hogyan lehet soros illesztőprogramot bejegyezni a soros illesztőrétegnél, illetve hogyan lehet egy soros kaput bejegyezni. Ez a soros kapu viselkedésében nagyon hasonlít a korábbi soros tty példa-illesztőprogramra, a kapu nyitott állapotának fennmaradásáig két másodpercenként érkezik egy karakter. A neki küldött karaktereket egy write(2) hívással a rendszermag hibakereső naplójába írja.

Linux Journal 2002. december, 104. szám

Greg Kroah-Hartman (greg@kroah.com)
Jelenleg a Linux-rendszerek USB és gyors csatlakoztatású (PCI Hot Plug) egységeinek rendszermagba épített meghajtóprogramjainak fejlesztője. Az IBM-nél dolgozik.

A PostgreSQL adatbázis-kezelő használata

Egy programozó szemszögéből vizsgálva…

A PostgreSQL (továbbiakban PG) korszerű objektum-relációs adatbázis-kezelő rendszer, amit a Berkeley Egyetem Számítástudományi tanszékének vezetésével fejlesztenek. A program támogatói között olyan híres szervezetek is részt vállalnak, mint az amerikai DARPA, ARO és NSF. A PG elődjének – az Ingresnek – a fejlesztése 1988 körül kezdődött. A jelenlegi PG 7.2-es változat támogatja az SQL92/SQL99 szabványokat, valamint lehetővé teszi az öröklődést, SQL-adattípusok, tárolt eljárások, kioldók (trigger), megszorítások, számlálók használatát, valamint a tranzakciókezelést (BEGIN WORK, COMMIT, ROLLBACK). A cikk célja, hogy az általános SQL-és programozási ismeretekre építve belépőt nyújtson a PG használatának megkezdéséhez.

Indulás a PG világába…

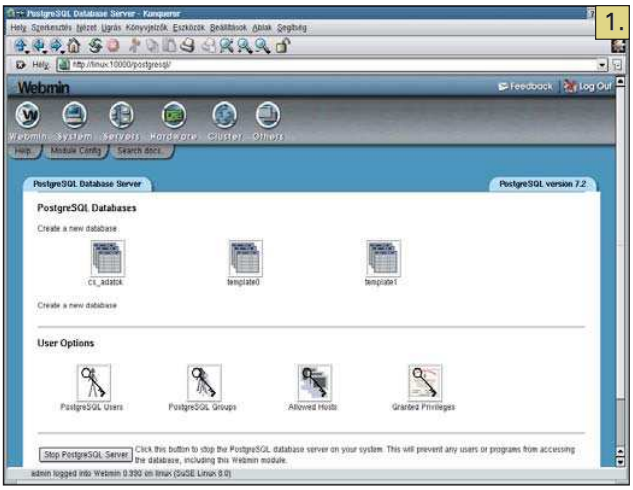
Az összes jelentős Linux-terjesztés tartalmazza a PG-t. Tekintettel arra, hogy ez egy nagyméretű alkalmazás, több csomagra szét van osztva, hiszen nem biztos, hogy minden összetevőre szükségünk lesz. A cikk mintapéldái SusSE Linux 8.0 alatt készültek, ahol az `rpm -qa | grep postgres` parancsot kiadva listázhatjuk ki a feltelepített PG-csomagokat:

```
postgresql-libs-7.2-86
postgresql-7.2-86
postgresql-contrib-7.2-86
postgresql-devel-7.2-86
postgresql-server-7.2-86
postgresql-tcl-7.2-86
postgresql-test-7.2-86
postgresql-tk-7.2-86
postgresql-jdbc-7.2-86
postgresql-odbc-7.2-86
```

Maga a PG-kiszolgáló és annak teljes leírása a `postgresql-7.2-86` csomagban található. A csomagok telepítése után javaslom megnézni, hogy engedélyezve van-e a TCP/IP-n keresztüli hozzáférés, illetve hogy az adatbázis-elérésre mely gépek jogosultak. A legegyszerűbb módszer erre a `su postgres` paranccsal átmenetileg Postgres-felhasználóvá válni, majd egy kapcsolók nélküli `cd` paranccsal belépni a Postgres ***saját*** könyvtárába. Itt találhatóak meg a ***pg_hba.conf*** és ***postmaster.opts*** fájlok (Linux-változattól függően ezek eltérőek lehetnek!).
Először nézzük meg a második fájl tartalmát, ami az alábbi egyetlen sorból áll:

```
/usr/bin/postmaster '-i' '-D/var/lib/pgsql/data'
```

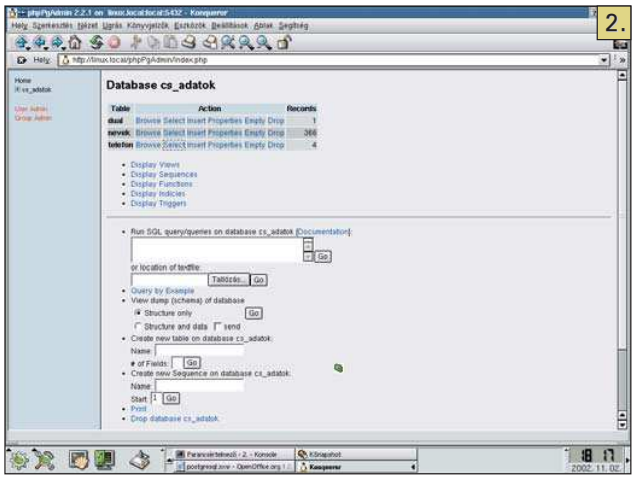
Ebből megtudhatjuk, hogy adatbázisaink a `-D` által jelölt könyvtárban helyezkednek el (az új adatbázisok is ide jönnek majd létre). A kis `"-i"` jelzi, hogy a postmaster kiszolgálófolyamat a TCP/IP-n keresztüli (azaz internet-) kapcsolódási kérelmeket is elfogadja, így ha ez a kapcsoló hiányzik, szúrjuk be a `-i`-t, majd a `/etc/init.d/postgresql restart` paranccsal indítsuk újra a PG-kiszolgálót.
A ***pg_hba.conf*** fájl tartalma határozza meg azokat a gépeket, amelyek adatbázisainkra bejelentkezni jogosultak. A fájlban megjegyzésbe tett mintasorok szerepelnek, így egy-egy új beállítósor beszúrása nem okozhat gondot.
Azok számára, akik a napi felügyeleti feladatokhoz jobban szeretik a grafikus felületet, a Webmin alkalmazást ajánlom, ami PG-adatbázis-kezelőnk Weben keresztüli felügyeletét is lehetővé teszi. A Webmin telepítése után böngészőnkben a következő címet adjuk meg: ➲ http://gépnév:10000. A ***felhasználónév és a jelszó*** megadása után bejelentkezik a Webmin indulóképe, ahol a ***Servers*** fülre, majd a PostgreSQL ikonra kattintva betöltődik a PG-t kezelő alkalmazás – ezt mutatja az *1. kép*.

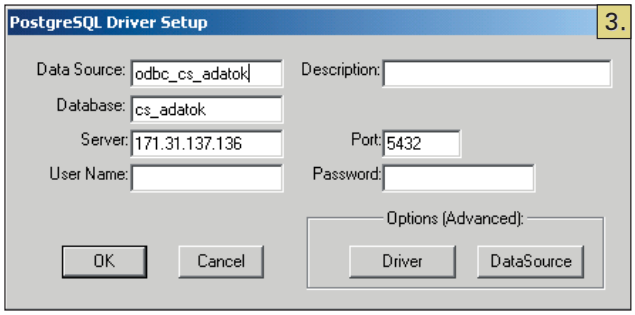


Ez a cikk nem a PG-kiszolgáló felügyeletéről, hanem annak programozásáról szól, ezért ezen a ponton abba is hagyjuk a telepítéssel és üzemeltetéssel kapcsolatos kérdések megválaszolását, és a következő lépésként inkább egy új adatbázist hozunk létre.

Egy új adatbázis létrehozása

A PG több segédprogramot (helyük általában a ***/usr/bin*** könyvtár) tartalmaz, amelyekkel minden feladatot el lehet végezni. Az alapértelmezett adatbázisgazda a `postgres`.
Új adatbázist a `createdb` paranccsal tudunk létrehozni, ennek neve legyen most `cs_adatok`. A szövegek alapértelmezésben SQL_ASCII kódban tárolódnak, de az adatbázis létrehozása során az ISO-8859-2 (Latin 2) karakterkódok használatát javaslom. Ez a beállítás elsősorban a Java-programok számára fontos, mert ezzel válik lehetővé az Unicode-ról való hibátlan oda-vissza történő karakterátalakítás. Itt az a gond, hogy az ő, ű karakterek nem képeződnek le az ISO-8859-1 kódlapra, hiszen ott nincs ilyen. Ez fordítva is igaz, azaz a kalapos ékezeteknek is lesz Unicode-beli párjuk, így azok nem a mi ő, ű karakte-reinkhez fognak rendelődni. A Latin 2 használatának további előnye, hogy ilyenkor nem kell kódátalakítást végezni az





ISO-8859-2-es kódlapon dolgozó ügyfél és kiszolgáló között. Nézzük meg adatbázis-létrehozó parancsunkat:

```
createdb -U postgres cs_adatok -E latin2
```

Érdemes még megjegyezni, hogy a `latin2` helyett az `unicode` is használható.
Ez a parancs a `-U` kapcsolóban tartalmazza azt a felhasználót, akinek jelenleg joga van PG-adatbázis-műveleteket végezni (természetesen további felhasználókat is felvehetünk, de ez már üzemeltetési kérdés). A jelszót azért nem kell megadnunk, mert éppen rendszergazdaként dolgozunk. Amennyiben a `su postgres` paranccsal egy új héjat kérünk, a `-U` kapcsoló minden parancsból elmaradhat, hiszen ezzel az operációs rendszer ekkor már azonosította a `postgres` felhasználót. Általában is igaz, hogy a PG elfogadja az operációs rendszer által már elvégzett felhasználói azonosításokat.
A PG több programozási nyelven (SQL, C/C++, plpgsql, pltcl, pltclu, plperl, plperlu, plpython) is biztosítja a tárolt eljárások és kioldók írását. Az adatbázis alapértelmezetten csak az SQL nyelvet ismeri.
Az Oracle-ben megszeretett PL/SQL nyelv alapján született meg a plpgsql nyelv (mindkettő az ADA leegyszerűsített változata), így ezt a lehetőséget mindenképpen érdemes kihasználni. Egy új tárolt eljárás készítését lehetővé tévő nyelvi modul telepítése a `createlang` paranccsal lehetséges. Tegyük elérhetővé a `cs_adatok` adatbázis számára a plpgsql nyelvet a következő parancs kiadásával:

```
createlang -U postgres -d cs_adatok plpgsql
```

Ezt követően az Oracle-programozók is azonnal ismerős programozási környezetben találják magukat.

A PSQL terminál felfedezése

A PG SQL konzolt a `psql` program valósítja meg, ami hasonló célokat szolgál, mint az Oracle *Plus. Kapcsolódjunk a most létrehozott `cs_adatok` adatbázishoz:

```
psql -U postgres cs_adatok
```

A sikeres kapcsolódást a következő képernyőtartalom igazolja vissza:

```
Welcome to psql, the PostgreSQL interactive
terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to
          execute query
       \q to quit

cs_adatok=#
```

A `psql` terminálon kétféle parancsot használhatunk. Az egyik egy bármilyen, közvetlenül kiadott SQL-parancs, amit a `psql` parancsjele után lehet beírni, és pontosvesszővel (;) kell lezárni. A másik parancscsoportot „slash” parancsoknak hívják, mert mindig visszaperjel (\) karakterrel kezdődnek, és olyan utasításokat fogalmaznak meg, amelyek magának a `psql` programnak szólnak.
A végrehajtható SQL-parancsokról a `\h`, míg a „slash” parancsokról a `\?` „slash” utasítások adnak részletes tájékoztatást.
Gyakorlásképpen hozzunk létre egy ***dual*** nevű táblát, hogy az Oracle-programozók még otthonosabban érezzék magukat a PG adatbázis-kezelőben:

```
cs_adatok=# create table dual (
cs_adatok(#    o1 varchar(10),
cs_adatok(#    o2 varchar(10)
cs_adatok(# );
CREATE
```

Látható, hogy az SQL-parancsokat több soron keresztül is gépelhetjük. Szúrjunk be egy új rekordot a ***dual*** táblába:

```
cs_adatok=# insert into dual values
            ('Valami1', 'Valami2');
INSERT 17045 1
```

Az egyes SQL-parancsokkal a továbbiakban nem foglalkozunk, hiszen az SQL-nek hatalmas irodalma van, és írásunknak nem az SQL ismertetése a célja. Nézzük meg inkább, hogy milyen további meglepetéseket tartogatnak a \ parancsok!
Sokszor szükség lehet rá, hogy az adatbázis-objektumok leírását megtekinthessük. Erre a szolgál a `\d` *`obj_nØv`* parancs (d=describe). Nézzük meg, mit mond ez az utasítás a most létrehozott ***dual*** táblára!

```
cs_adatok=# \d dual
                Table "dual"
 Column  |          Type          | Modifiers
---------+------------------------+-----------
 o1      | character varying(10)  |
 o2      | character varying(10)  |
```

1. lista

```
-----------------------------------------
-- Visszaadja az  sszesen fizetett alapd j
--  sszegØt,amelynek telefononkØnti d ja a
-- f ggvØny ØrtØke forintban
-----------------------------------------
create or replace function MyFunc(integer)
returns integer as '
declare
 s integer := 0;
begin
 // LekØrdezi Øs s-ben tÆrolja a telefonok
 // szÆmÆt
 select count(*) into s from telefon;
 // KiszÆm tja az  sszesen beszedheti
 // alapd jat
 s := ($1) * s;
 return s;
end;
' language 'plpgsql';


-----------------------------
-- BeszØr egy Øj telefonszÆmot
-----------------------------
drop function MyFunc2(varchar);
create or replace function
MyFunc2(varchar,varchar) returns integer as
'
declare
 s varchar;
 p varchar;
begin
  s := $1; p := $2;
  insert into telefon values ( s,  p);
  return 1;
end;
' language 'plpgsql';
```

Alapvető igény, hogy egy előre elkészített SQL-parancsfájlt végrehajthassunk. Legyen egy ***nevek_db.sql*** parancsfájlunk, aminek az első pár sora így néz ki:

```
create table nevek (
        honap    int2,
        nap      int2,
        nevnap   varchar(50)
);

insert into nevek values ('1','1','Fruzsina');
insert into nevek values ('1','2',''bel');
insert into nevek values ('1','3','GenovØva,
BenjÆmin');
...
```

Ezt a fájlt a `\i /opt/imre/sql/`*`nevek_db`*`.sql` paranccsal futtathatjuk le, ennek hatására létrejön a ***nevek*** tábla és töltődik fel adattal.
A psql-ben a pillanatnyi könyvtár fogalma is létezik, amit a `\cd` *`k nyvtÆr`* alakban adhatunk meg, elkerülve, hogy min-

2. lista

```
CREATE FUNCTION uj_telefon(varchar, varchar)
returns integer  AS '
INSERT INTO telefon VALUES ($1, $2);
select 1;
' LANGUAGE 'SQL';
```

3. lista

```
#include "postgres.h"
//  rtØk szerint
int duplazo(int i)
{
  return 2 * a;
}

// C m szerint
float8 *fduplazo(float8 *f)
{
  return 2.0 * a;
}
```

den parancsfájlt a teljes neve megadásával kelljen futtatni.
A psql programból a \q slash paranccsal léphetünk ki. Tegyük meg! Miért? Azért, hogy kipróbálhassuk a psql parancs -c kapcsolójának használatát, ami egy bármilyen SQL- vagy slash parancsot végrehajt, majd a psql program futása befejeződik. Ezen a ponton elérkeztünk oda, hogy héjprogramjainkba PG SQL-részleteket is betehetünk. Kezd izgalmassá válni, ugye? Nézzünk egy egyszerű héjparancsot, ami kilistázza a ***dual*** táblát:

```
linux:~ # psql -U postgres cs_adatok -c
"select * from dual"
   o1    |    o2
---------+---------
 Valami1 | Valami2
(1 row)
linux:~ #
```

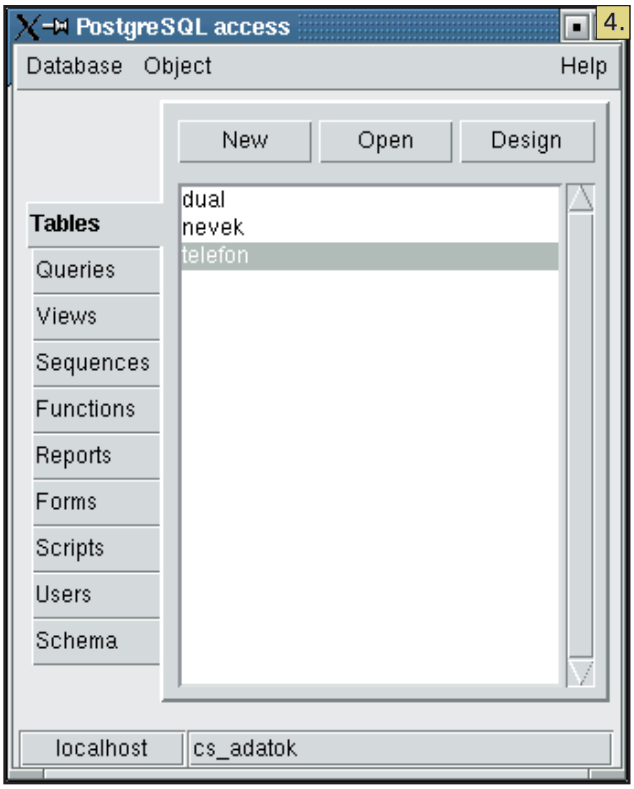
Természetesen ezzel a módszerrel egy teljes SQL-parancsfájl is lefuttatható. Az előbbi select parancsunkat írjuk be egy ***listdual.sql*** nevű fájlba, majd adjuk ki a következő parancsot:

```
psql -U postgres cs_adatok -c
➥"\i /opt/imre/sql/listdual.sql"
```

Ez az előző parancséval teljesen megegyező kimenetet fog eredményezni.
A psql tartalmaz néhány On/Off jellegű kapcsolóparancsot is. A \t parancs segítségével ki- és bekapcsolhatjuk, hogy egy select parancs sorai tartalmazzák-e a fejlécet. A \H-val a html-, illetve nem html-mód között kapcsolhatunk. Nézzünk erre egy példát!

```
linux:/opt/imre/sql# psql -U postgres
cs_adatok -H -c "select * from telefon"

<table border=1>
  <tr>
```




```
    <th align=center>nev</th>
    <th align=center>szam</th>
  </tr>
  <tr valign=top>
    <td align=left>Nyiri Imre ıı   </td>
    <td align=left>123456789</td>
  </tr>
  <tr valign=top>
    <td align=left>Kiss JÆnos</td>
    <td align=left>123456789</td>
  </tr>
  <tr valign=top>
    <td align=left>PØntek TamÆs</td>
    <td align=left>123456789</td>
  </tr>
</table>
(3 rows)<br>
```

Adjuk ki parancssorból a következő utasítást, ami a fejléckiírás-letiltást is tartalmazza:

```
linux:/opt/imre/sql# psql -U postgres
cs_adatok  -t  -c "select * from telefon"
 Nyiri Imre ıı    | 123456789
 Kiss JÆnos       | 123456789
 PØntek TamÆs     | 123456789
```

E parancs eredménye már olyan formátumú, amit bármelyik Unix-parancsfájlban kiválóan alkalmazni tudunk. Beállítható az is, hogy az oszlopelválasztó karakter ne a csőkarakter (|), hanem valamely más karakter legyen; például vessző (,) esetén olyan kimenetet kapunk, ami az Excel vagy OpenOffice által ismert formátum (csv). A psql további lehetőségeinek izgalmas felfedezését olvasóinkra hagyjuk.

Készítsünk tárolt eljárásokat!

Ebben a pontban megnézzük, hogyan lehet SQL, PLPGSQL, illetve C/C++ nyelven megfogalmazott tárolt eljárásokat készíteni. A kódot az ***eljarasok.sql*** fájlba gépeljük be (lásd az *1. listát*). Ezután adjuk ki a következő parancsot, ami a `cs_adatok` adatbázisban létrehozza a `MyFunc()` és `MyFunc2()` tárolt eljárásokat:

```
psql -U postgres cs_adatok -c
     ➥"\i /opt/imre/sql/eljarasok.sql"
```

Láthatjuk, hogy az eljárások utolsó sorában a tárolt eljárás nyelvét a `language plpgsql` sor határozta meg, ami az ORACLE PL/SQL-t ismerők számára nem nyújt túl sok újdonságot.
A függvényeknek csak a típusát kell megadni, és az átvett értékekre a `$1, $2...` helyhatározó névvel lehet hivatkozni.
Most nézzünk egy rövid példát egy SQL tárolt eljárásra, amit az ***sqleljaras.sql*** fájl tartalmaz (lásd *2. listánkon*). Psql-ben kiadva a `"select uj_telefon('Janisovszky KÆroly', '11111');"` parancsot a telefontáblába egy új sor szúródik be.
A tárolt eljárások terén végzett kalandozásunk befejezéseként nézzük meg, hogyan lehet C/C++ nyelvű tárolt eljárást készíteni. Először is kell egy C/C++ nyelvű program, amit a ***duplazo.c*** forrásfájlba (*3. lista*) gépeltem be.
Következő lépésként a listán szereplő két függvényt egyetlen ****.so*** (shared object file = megosztott objektfájl) fájlba fordítjuk:

```
gcc -fpic -c duplazo.c
gcc -shared -o libduplazo.so duplazo.o
```

Az `fpic` a position-independent code rövidítése. Ezzel egy so-fájlt kaptunk, a dinamikus összeszerkesztő (linker) számára azonban még meg kell adnunk, hogy hivatkozáskor hol keresse ezt a könyvtárat (azaz a ***libduplazo.so*** fájlt). Ez a `JAVA CLASSPATH` szerkezethez hasonló, de a környezeti változót itt `LD_LIBRARY_PATH`-nak hívják. Másoljuk be a ***libduplazo.so*** fájlt egy ***/opt/pglibs/pg_so*** könyvtárba, majd gondoskodjunk arról, hogy az `LD_LIBRARY_PATH`-ban ez a könyvtár is benne legyen. Ez a módszer nem az egyetlen, hogy a ***libduplazo.so*** keresési helyét a dinamikus összeszerkesztő tudtára adjuk, de most nem ez a fő témánk.
Utolsó lépésként még az adatbázisba is be kell jegyezni egy hivatkozást a `psql` konzol segítségével:

```
create function duplazo( integer )
       returns integer as
       '/opt/pglibs/pg_so/libduplazo.so'
       language 'C'
```

Itt a C-forrásban lévő két függvény közül csak az első használatát tettük elérhetővé. A tárolt eljárás megvalósítása az a C nyelvű függvény lesz, aminek a ***libduplazo.so*** fájlban ***_duplazo*** a neve. Ezek után kiadhatjuk a `select duplazo(3);` parancsot, ami eredményként hatot fog kiírni.
Folytatjuk...

Nyíri Imre (inyiri@mol.hu)
Jelenleg a MOL Rt.-nél dolgozik. Informatikai vállalkozásában az Internet, a Linux, valamint a Java-programozás gyakorlati hasznosításával foglalkozik. Örök szerelme a C++ maradt.

A biztonság növelése FTP proxy használatával

Mick azt mutatja be, hogyan lehet biztonsági réteget helyezni a rossz szándékú felhasználók és nyilvános hozzáférésű FTP-kiszolgálónk közé.

Egy nyilvános hozzáférésű FTP-hely üzemeltetése nehéz feladatnak bizonyulhat, de egy tetszőleges FTP-kiszolgálóalkalmazás által nyújtott összes biztonsági szolgáltatás lehetőség szerinti legjobb kihasználása is kemény munkának ígérkezik. Mindennek ellenére a sebezhetőségi pontok előbb vagy utóbb nyilvánvalóvá válnak, és semmissé teszik az addig elvégzett munkát. Vagy tehetünk valami mást is? Fontos módszer, hogy az FTP-proxy alkalmazást a tűzfalunkon futtatjuk. Miközben a szabványos rendszermagba épített Netfilter-kód csak a csomagok vizsgálatára képes, az FTP-proxy a tűzfal számára lehetővé teszi, hogy az minden FTP-műveletben közvetítőként működjön. Ez fokozza a biztonságot az ideiglenes tárterületek (pufferek) túlcsordulásával és sok másfajta FTP-támadással szemben. Ezenkívül megengedi, hogy az FTP-ügyfelek számára megszabjuk, milyen parancsokat hajthatnak végre.
E hónapban azt mondom el, hogy a SuSE ingyenes – és a nem csak SuSE alatt futó – Proxy-Suite (készlet) FTP-proxykiszolgálóját hogyan kell Linux-tűzfalon üzemeltetni, átlátható, de erős védelmet nyújtva minden egyes FTP-művelet számára.

A proxy-suite beszerzése és telepítése

Ha SuSE Linux-változatot üzemeltetsz, telepítheted a proxy-suite csomagot, amely az ftp-proxy bináris példányát telepíti annak beállító állományával és héjprogramjával együtt. Amennyiben az ftp-proxyt áttetsző proxyként kívánod használni, vagy elvárod tőle, hogy LDAP-hitelesítést is végezzen, akkor a legfrissebb változatot kell beszerezned – ez az írás elkészültekor az 1.9-es volt.
A legfrissebb programváltozat működtetéséhez, illetve annak SuSE Linux-változatoktól eltérő rendszerekben való használatához a legjobb megoldás, amelyet választhatunk, ha magunk fordítjuk a forráskódot, amelyet az alábbi címről szerezhetünk be: ➲ http://ftp.suse.com/pub/projects/proxy-suite/src.

Programépítés forráskódból

Az ftp-proxy összeépítését és telepítését érintő összes utasítás megtalálható az INSTALL nevet viselő állományban. Az előzetes beállításoknak megfelelően a beállítási héjprogram ellenőrzi, hogy a libwrap, libldap csomagok telepítve vannak-e, valamint azt, hogy a rendszer támogatja-e szabványos kifejezések (regular expressions) használatát. Red Hat 7.3-as rendszeremen a libwrap és csomag telepítve volt ugyan, de fordításiidő-hibát eredményezett (compile time error), emiatt a libwrapet az alábbi módon kikapcsoltam:

```
# ./configure --without-libwrap
```

és így az ftp-proxy fordítása már megfelelően lezajlott. Mindazonáltal ez a lépés szükségtelen volt, amikor az ftp-proxyt a SuSE 7.1-es rendszeremen fordítottam: magától értetődő, hogy a SuSE- és a Red Hat-változatok libwrap csomagjai eltérőek.
Az ftp-proxy összeépítését és a dokumentáció felmásolását felmásolását is magában foglaló telepítését követően új proxy-kiszolgálónkhoz szükségünk támadhat egy indító héjprogramra. Szerepel az ftp-proxy forráskódjában – az ***ftp-proxy***/ könyvtárban – egy minta héjprogram, az ***rc.script***, amelynek tartalmát a kísérő ***rc.script.txt*** állomány ismerteti.
SuSE változatokban az ***rc.script***-et egyszerűen be kell másolni a ***/etc/init.d*** könyvtárba, majd az egyéni igényeknek megfelelően egy erre mutató közvetett hivatkozást (symbolic link) kell készíteni a ***/usr/sbin*** könyvtárból. Nevezzük át a szóban forgó héjprogramot a ***/etc/init.d/ftp-proxy*** névre, és a közvetett hivatkozást ***/usr/sbin/rcftp-proxy***-ra. Ha valamilyen SuSE 7.x rendszert üzemeltetsz, a ***/etc/rc.config*** állományt a fentieken kívül még a következő sorral is ki kell egészítened:

```
START_FTP_PROXY="yes"
```

A SuSE-től eltérő Linux-változatoknál az ***rc.script*** állomány jelentős mértékben módosításra szorul, mivel ez az állomány nagyon is jellemző a SuSE Linux-változatra. Példák végett vedd szemügyre a Linux-változatod ***init.d*** könyvtárában található egyéb héjprogramokat.
Nos, ha sikerült valami okosat kisütni, javaslom, hogy az átalakított héjprogramot küldd el ***Marius Tomaschewski***-nek (mt@suse.de), aki az ftp-proxy egyik fő fejlesztője, hogy az ő révén mások is hasznosíthassák ragyogó ötleteidet.

Az ftp-proxy beállítása

Amennyiben már akár forráskódból, akár SuSE-csomagból elvégezted az ftp-proxy telepítését, elérkezett az ideje, hogy beállítsuk. A legtöbb beállítható paraméter a ***/etc/proxy-suite/ftp-proxy.conf*** állományban található, illetve a ***/usr/local/etc/proxy-suite/ftp-proxy.conf*** állományban, amennyiben forráskód alapján végeztük a telepítést.
Mielőtt azonban fejest ugranánk az ***ftp-proxy.conf*** állományba, akad néhány apróság, amire figyelmet kell fordítanunk. Először is proxydémonodnak szüksége lesz egy különleges jogok nélküli felhasználói azonosítóra, amelyet használhat. A saját rendszeremen létrehoztam egy ilyen azonosítót:

```
bash-# useradd -u 65500 -g nogroup
➥-d /var/ftp-proxy/rundir -s /bin/false ftpproxy
```

Senki sem jelentkezhet be ezzel a felhasználói névvel. Hogy ezt biztosra vehessük, a ***/etc/shadow*** állományban a proxyfelhasználói bejegyzéshez tartozó sorba tegyünk egy csillagot:

```
ftpproxy:*:12345:0:99999:7:0::
```

Aztán készítened kell egy chroot-dobozt (jail), amelyben a ftp-proxy gyermekfolyamatai majd működni tudnak. A SuSE Linux változat felhasználói számára ez egyszerű feladat – az ftp-proxy indító héjprogramja ezt elvégzi helyettük, ha indítása

1. lista ftp-proxy.conf

```
  ServerType          standalone
  User                ftpproxy
  Group               nogroup
  LogDestination      daemon
# LogLevel            INF
                      # DBG csak hibakeresØs
                      # idejØn hasznÆland !
  PidFile             /var/run/ftp-proxy.pid
# ServerRoot          /var/ftp-proxy/rundir
  AllowMagicUser      no
  AllowTransProxy     yes
  DestinationAddress  192.168.1.2
  ValidCommands       USER, PASS, PWD, CWD,
                      CDUP, PORT, PASV,
                      RETR, TYPE, REST,
                      ABOR, LIST, NLST,
                      TAT, QUIT
```

Tűzfallal kapcsolatos alapismeretek

Ha a tűzfalhasználat számodra még újdonságszámba megy, ez a cikk nagyon keveset fog mondani a számodra. A *Kapcsolódó címekből* kövesd végig *Farrow* és *Power* „A tűzfal-technológiák története" című cikkének gondolatmenetét és a „Building Secure Servers with Linux" című könyvemét is. Mindkettő angol nyelven következő sítlusban tárgyalja a különféle tűzfalelgondolásokat és megvalósításokat.

a `chroot` paranccsal együtt történik:

```
bash-# /etc/init.d/ftp-proxy chroot
```

De még akkor is, ha nem éppen SuSE Linuxot üzemeltetsz, a példaprogramot – a korábban említett ***rc.script*** héjprogramot – meglehetősen egyszerű visszafejteni, hogy kitaláljuk, hogyan is lehetne ezt megoldani. Tulajdonképpen a szükséges ftp-proxy doboz a */var/ftp-proxy/rundir*, ennek tartalmaznia kell azokat a könyvtárakat és állományokat, amelyeket az ftp-proxy használ, valamint a saját */dev/log* különleges naplóállományt, amelyre a helyi `syslog` démon figyelni képes. A `syslog` démonnak a chroot-olt naplózóprogramra való irányításához az indító héjprogramot csupán `-a` kapcsolóval kell kiegészíteni, hogy az elindítsa a rendszernaplózást:

```
syslog -a /var/ftp-proxy/rundir/dev/log
```

A SuSE-rendszerekben ennek a */etc/rc.config* állományban a `SYSLOGD_PARAMS` változón keresztül történő beállítás a megszokott módja. De akár többszörös `-a` utasításokat is kiadhatunk, például akkor, ha a chroot-olt `named`-től fogadunk naplókat.

Az ftp-proxy.conf állomány

Végül eljött az ideje, hogy beállítsuk a proxydémont. Mint azt már korábban említettem, ez az *ftp-proxy.conf* állományban történik meg, amely a */etc/proxy-suite* vagy a */usr/local/etc/proxy-suite* könyvtárak egyikében található. Zavaró vagy bosszantó lehet, hogy a SuSE szóhasználatában a suite elnevezés mindössze egy egyetlen alkalmazásból álló „készletre" utal. Remélhetőleg hamarosan további proxyk fognak elkészülni, és ha annyira hasznosak lesznek, mint az ftp-proxy, én a magam részéről meg fogom bocsátani nekik ezt az aprócska csalást. Ennek az állománynak a leggyorsabb ismertetése a példák felsorolása és részekre szedése (lásd *1. listánkon*). Az első értéket a `ServerType` határozza meg, hogy az ftp-proxyt önálló démonként vagy az `inetd`-ből kell-e futtatni. Jóllehet a programot az imént démonnak neveztem, valójában mindkét üzemmódra képes. Én a nyilvános hozzáférésű kiszolgálógépeken szeretem elkerülni az `inetd` vagy akár `xinetd` futtatását, mivel így az előzetes beállításnak megfelelően elinduló programokat nem kell állandóan kikapcsolgatnom, de a démonként futó programok működéséből származó előnyök miatt is szeretem. Abban az esetben, ha igényeid ettől eltérőek, a `ServerType`-ot `inetd`-re állíthatod, ami szintén működőképes megoldás, ha az `inetd` helyett az `xinetd`-t futtatod.
A felhasználói és a csoportnév magától értetődően elegendő a felhasználói azonosító (UID) és a csoportazonosító (GID) meghatározásához, hogy a kezdőértékek feltöltését követően melyik ftp-proxykiszolgáló alatt fusson. Jó ötlet ezeket különleges jogok nélküli UID-hoz és GID-hez rendelni, hogy csökkentsük a kockázati következményeit annak, hátha egy betörőnek valahogy mégiscsak sikerül eltérítenie egy ftp-proxyfolyamatot. A `LogDestination` kijelöli, hogy az ftp-proxy hová küldje a naplóüzeneteket. Ez lehet egy démon – azaz a helyi rendszernaplózást végző syslog-szolgáltatás – vagy egy állomány, sőt egy csővezeték is. A `LogLevel` a naplózandó adat mennyiségét jelöli: a legtöbb felhasználó számára az `INF` a legjobb választás, azonban a `DBG` – vagyis a lehető legtöbb adat begyűjtése – a hibakeresés során bizonyulhat hasznosnak. A `PidFile` az ftp-proxy számára kijelöli, hol legyen tárolva a mesterfolyamat folyamatazonosítója. Ezt az indító héjprogram használja, amit rendszerleálláskor a `stop` parancs hív meg. Ha viszont az ftp-proxy `inetd` üzemmódban működik, akkor a `PidFile` nincs használatban.
A `ServerRoot` meghatározza a ftp-proxy chroot-dobozának elérési útvonalát. Hagyjuk meg megjegyzéssornak, ha nem szeretnénk, hogy az ftp-proxy a futás során chrootolásra kerüljön (lásd „Chroot kérése az 1.9-es változat alatt" címet viselő keretes írásunkat).

Áttetsző proxyhasználat

Az 1. listában szereplő következő három parancs különösen fontos, ezek határozzák meg, hogy a proxy vajon áttetsző lesz-e. A legtöbb helyzetben az áttetsző proxy a kívánatosabb – a végfelhasználóknak nem kell FTP-ügyfélprogramjuk beállítását elvégezni ahhoz, hogy proxyhasználatukat egyértelműen támogassák. Ehhez csupán az szükséges, hogy az ftp-proxy a rendszermagba fordított Netfilter-kóddal összhangban dolgozzék, amely az FTP-csomagokat a proxydémonunkhoz irányítja, ahelyett, hogy elküldené őket annak a gépnek, amelyiknek tulajdonképpen címezve lettek. Amikor az ftp-proxy ilyen módon átirányított FTP-ügyfélcsomagokat kap, a cél IP-címet annak az új FTP-kapcsolatnak a céljaként használja fel, amelyet a kívánt FTP-kiszolgálóval létesít. A `DestinationAddress` érték az alapértelmezés szerint használandó célt jelöli meg.
Ha felhasználóink számára lehetővé szeretnénk tenni, hogy proxykiszolgálónkat ne áttetsző módon használják, más szóval, hogy FTP-folyamataik kapcsolataikat közvetlenül a proxyki-

Az RFC 959-es szabvány által meghatározott FTP-parancsok

Az RFC 959-es szabvány parancsai	*és azok leírása*
USER	A felhasználó nevének megadására szolgál.
PASS	A jelszó megadására szolgál.
PWD	Az éppen használt munkakönyvtárat írja ki.
CWD	A munkakönyvtár váltására szolgál.
CDUP	A munkakönyvtár szülőkönyvtárára váltó parancs.
TYPE	Adattípus: IMAGE (bináris), ASCII, EBCDIC vagy L (lokális bájtokban kifejezett méret).
MODE	Az adatfolyam típusát állítja be (stream, block, compressed); állítja; a compressed adatfolyam mód visszaélésekre teremt lehetőségeket, ezért nem szabad megengedni, hogy a felhasználók a MODE-ot megváltoztassák.
PASV	Ez a parancs arra utasítja kiszolgálógépet, hogy készüljön fel az adatcsatorna passzív üzemmódú használatára.
PORT	A kiszolgálógépet arra utasítja, hogy az adatcsatornát aktív módban indítsa el (active mode).
RETR	Állomány visszakeresése, tulajdonképpen olvasása.
STOR	Állomány küldése, vagyis felírása a kiszolgálóra.
APPE	Állomány küldése – ha már létezik, akkor az állomány hozzáfűzése a létező állomány végéhez.
REST	Gyors előrecsévélés egy megadott helyzetig. Az utasítást a RETR vagy STOR utasítások egyikének kell követnie.
RNFR	Átnevezés valamiről, amely parancsot ugyanebben a sorban az RNTO parancsnak kell követnie.
RNTO	Átnevezés valamire: ezt a parancsot ugyanebben a sorban az RNFR parancsnak kell megelőznie.
STOU	Állomány felírása, az állomány nevének megváltoztatása, ha az állomány már létezik.
ABOR	Az előzőleg kiadott parancs és annak bármilyen járulékos műveletének törlése (állományátvitel) stb.
DELE	Állomány törlése.
RMD	Könyvtár eltávolítása.
MKD	Könyvtár létrehozása.
LIST	Az állományok nevének és tulajdonságainak felsorolása.
NLST	Csak az állományok nevét mutató rövid lista.
SYST	A kiszolgálógéptől lekérdezi az rajta futó operációs rendszer típusát.
STAT	Egy megadott állomány tulajdonságainak vagy a kiszolgáló gép/kapcsolat állapotának lekérdezése.
QUIT	A kapcsolat leállítása.

Chroot kérése az 1.9-es változat alatt

Talán neked is gondjaid támadnak, amikor az ftp-proxy-t chrootolva futtattad – nekem legalábbis akadtak. Mindjárt az elején ftp-proxy indító héjprogramjában a chrootdoboz használata során furcsa eredményekhez jutottam az ftp-proxy chroot támogatása révén. Az 1.9-es változattal folytatott próbaüzem során az ftp-proxy nem volt hajlandó elindulni, ekkor a *Can't determine Group-ID to use* (Nincs meghatározva, hogy milyen csoportazonosítót kell használni) üzenettel ért véget. Az 1.7-es változattal végzett próbafuttatások során a fentebb említett gond nem jelentkezett, de áttetsző üzemmódban az ftp-proxy nem volt képes megfelelően megoldani a címfeloldást. Lehet, hogy olvasóim jobb eredményekre jutnak, mint jómagam.
Ha kétségek merülnének fel, segítséget kaphatunk egy a SuSE Linux által erre a célra fenntartott levelezőlistáról. A feliratkozáshoz küldj egy elektronikus levelet a proxy-suite-subscribe@suse.com címre.

szolgálóval kezdeményezhessék, akkor az `AllowMagicUser` értékét ***yes***-re állítani. Ezt azonban nem javaslom, ha a proxykiszolgálót külső felhasználók is használni fogják, minthogy pontosan ez az eset áll fenn a nyilvános hozzáférésű FTP-kiszolgálókkal. Az `AllowMagicUser` változó hatására proxykiszolgálónk olyan nyitott proxyként fog működni, amelyet a külső felhasználók alkalomadtán arra is felhasználhatnak, hogy általa más, külső kiszolgálókhoz kapcsolódjanak, esetleg támadási célzattal.
Ha azonban a Netfiltert úgy állítottuk be, hogy a proxy felhasználói kapcsolatokat kizárólag a belső hálózati, „tiszta” oldalról fogadjon el, és az `AllowMagicUser` változót ***yes***-re állítottuk, a felhasználóknak lehetőségük lesz arra, hogy a felhasználói azonosítót és a kukacot (@) követően megadják az elérni kívánt FTP-kiszolgálót, például mick@ftp.wiremonkeys.org. Az `AllowMagicUser` használható, tekintet nélkül arra, hogy az `AllowTransProxy` változó értéke ***yes***-re vagy ***no***-ra lett-e állítva. Fontos megjegyezni, hogyha ez utóbbi változó értéke ***no***-ra lett állítva, és az `AllowMagicUser` változóé szintén, akkor minden FTP-kapcsolat a `DestinationAddress` változót fogja használni. Az egyéb paraméterek között szerepel a `MaxClientString` és a `DestinationTransferMode`. A teljes lista végett olvasd el a ftp-proxy.conf(8) súgóoldalát, ugyanitt a fentebb ismertetett változókról bővebb ismeretek is beszerezhetők.

A Netfilter beállítása transzparens proxyhasználathoz

A működőképes transzparens proxyhasználathoz az IP Tables telepítésére van szükség, ami képes az FTP-csomagokat a helyi proxykiszolgálóhoz irányítani, más szóval a proxykiszolgáló gépen a Netfiltert kell futtatni – a cikkben ezt tételezzük fel. Természetesen ezenkívül szükség lesz az FTP-kapcsolatokat leíró szabályokra is, vagyis arra, hogy a proxykiszolgálóhoz milyen kimenő és bejövő kapcsolatok megengedettek. A továbbítási láncban (FORWARD chain) viszont semmilyen szabályra nem lesz szükség.
Először is az áttetsző proxyhasználat támogatásához a Linux 2.4-es tűzfalra több modult be kell tölteni: az `ipt_conntrack_ftp` és az `ip_nat_ftp` modul szükséges az FTP-kapcsolat nyomon követéséhez; az `ipt_REDIRECT` modul pedig a szabály céljának átirányításához nélkülözhetet-

2. lista Az IP Tables program – a transzparens proxyhasználat parancsai

```
iptables -t nat -A PREROUTING -p tcp -i eth2 \
  --dport 21 -j REDIRECT
iptables -t nat -A PREROUTING -p tcp -i eth0 \
  --dport 21 -j REDIRECT

    # kihagyÆs ...

iptables -A INPUT -p tcp -d $PUBLIC_FTP \
  --dport 21 -m state --state NEW,RELATED \
    -j ACCEPT
iptables -A INPUT -p tcp -s $INTERNAL_HOSTS \
  --dport 21 -m state --state NEW,RELATED \
    -j ACCEPT

    # kihagyÆs ...

iptables -A OUTPUT -p tcp -d $PUBLIC_FTP \
  --dport 21 -m state --state NEW,RELATED \
    -j ACCEPT
iptables -A OUTPUT -p tcp -o eth2 --dport 21 \
  -m state --state NEW,RELATED -j ACCEPT
```

len. A legtöbb Linux-változat valamilyen 2.4-es rendszermagot állít csatasorba, amelyek ezeket a modulokat már tartalmazzák. Amint a modulok betöltése megtörtént, Netfilter indító héjprogramod számára tűzfalszabályokat adhatsz meg, amint az a *2. ábrán* látható. A *2. lista* első két utasítása a tűzfalat arra utasítja, hogy az összes olyan külső és belső csatolófelületén (`eth2` és `eth0`) kapott csomagot irányítsa át, amiknek a TCP 21-es kapuja a célkapuja, vagyis a kiszolgálógépen az FTP számára fenntartott kapu. Fontos megjegyezni, hogy ezek a csomagok semmilyen módon nem lesznek újraírva (mangled), egyszerűen csak át lesznek irányítva a helyi FTP proxydémonhoz.
A 2. lista harmadik és negyedik parancsa arra utasítja a tűzfalat, hogy fogadja el a nyilvános hozzáférésű FTP-kiszolgáló 21-es kapujára küldött összes csomagot – ahol a `PUBLIC_FTP` változó tartalmazza annak IP-címét –, és a belső felhasználók által küldött összes FTP-csomagot, ahol az `INTERNAL_HOSTS` változó tartalmazza a címtartományt a CIDR-jelölésnek megfelelően, vagyis például `192.168.99.0/24`. Az első két sor miatt, ha a harmadik és negyedik sorban lefektetett szabályoknak megfelelő csomag érkezik, az a helyi proxyhoz lesz irányítva.
A 2. lista ötödik és hatodik sora a helyi ftp-proxy démon számára lehetővé teszik, hogy a külső csatolófelületről kapcsolatot létesítsen a megadott nyilvános hozzáférésű FTP-kiszolgálókkal és gépekkel, ez ebben a példában az `eth2` volt.
A 2. listában szereplő sorok nem alkotnak önálló Netfilter-szabálygyűjteményt, csupán azt mutatják, hogyan egészítheted ki a címfordítást (NAT) és egyebeket. A héjprogram a `PUBLIC_FTP` és `INTERNAL_HOSTS` változókat is be kell állítsa. Jó szokás ilyen beszédes neveket használni, ez ugyanis érdekesebbé teszi a kódot.

Az FTP-használatot korlátozó parancsok

Most pedig térjünk vissza az ***ftp-proxy.conf***-hoz (1. lista), pontosabban az ftp-proxy egyik legfontosabb szolgáltatásához: a `ValidCommands`-hoz. Ez nem más, mint a proxy által megengedett, vesszőkkel elválasztott parancsok felsorolása. A sorokat a fordított perjel írásjellel (\) zárjuk le, a lista akár több sorra is kiterjedhet. Az 1. lista alján a `ValidCommands` utasításban az ftp-proxy úgy lett beállítva, hogy az FTP-könyvtárak közötti tájékozódást segítő parancsok megengedettek: `PWD`, `CWD`, `CDUP`; az FTP-olvasó parancsai: `LIST`, `NLST`, `RETR`; ezenkívül néhány felügyeleti parancs: `MODE`, `PORT`, `PASV`.
Itt helyhiány miatt nem áll módomban részletes magyarázatot adni, csupán annyit mondhatok, hogy ezek nem végfelhasználói ügyfélparancsok, hanem olyan, az FTP-protokollnak szóló parancsok, amelyek meghatározását megtalálhatjuk az RFC959-ben (➲ http://ftp.isi.edu/in-notes/rfc959.txt). E parancsokat az FTP ügyfél- és kiszolgálóalkalmazások egymás között használják. A parancsok összesítését *táblázatunk* tartalmazza.
Az ftp-proxy egyik korlátozása az, hogy a külső felhasználók számára nem lehet a belső felhasználókétól eltérő korlátozó parancsokat kialakítani. Emiatt aztán a `ValidCommands`-sal légy óvatos – ha az FTP-kiszolgálóknak a belső felhasználóknak kell állományokat küldenie, akkor nincs lehetőség a `STOR` vagy `STOU` parancsok korlátozására, vagyis ez azt jelenti, hogy ezeket a parancsokat fel kell venni a `ValidCommands` parancsai közé. De ez egyúttal azt a feladatot is magában hordozza, hogy meg kell győződnünk róla, hogy a csak olvasható, nyilvános hozzáférésű FTP-kiszolgáló úgy van-e beállítva, hogy ezeket a parancsokat figyelmen kívül hagyja őket.

Összegzés

Az FTP-proxy fontos biztonsági réteget képez a tisztességtelen emberek és a nyilvános hozzáférésű FTP-kiszolgálógép között. A cikkben ismertettem az áttetsző proxy-használatot a SuSE Linux proxy-suite készletével, ami még sok megismerésre érdemes szolgáltatással dicsekedhet, de ezeket most nem tudtam bemutatni. Bővebb ismeretek végett böngészd át a ***Kapcsolódó címek*** részt. Szerencse fel!

Linux Journal 2002. december, 104. szám

Mick Bauer (mick@visi.com)
Hálózati biztonsági tanácsadó az Upstream Solutions Inc.-nél Minneapolisban (Minnesota). Mick a szerzője a hamarosan megjelenő új O'Reilly könyvnek, amelynek címe „Building Secure With Linux".

KAPCSOLÓDÓ CÍMEK

A hivatalos SuSE Linux Proxy-Suite honlapja
➲ http://www.suse.de/en/whitepapers/proxy_suite
Az FTP – File Transfer Protocol, illetve az RFC, amely meghatározza, hogyan is működik az FTP
➲ http://ftp.isi.edu/in-notes/rfc959.txt
Rik Farrow és *Richard Power* History of Firewall Technologies (A tűzfal-technológiák története)
➲ http://www.spirit.com/Network/net0597.txt
A *Mick Bauer* nevével fémjelzett Building Secure Servers with Linux (Biztonságos kiszolgálógépek építése Linuxszal) című könyv hivatalos honlapja
➲ http://www.oreilly.com/catalog/bssrvrlnx

Linux alatt Windows vagy fordítva?

E havi cikkünkben megnézzük, hogyan is futtathatunk x86-os processzorra készült teljes operációs rendszereket a VMWare segítségével.

A VMWare teljes mértékben megfelel egy szabványos PC-nek, egészen a Phoenix BIOS szintig, ami egyben azt jelenti, hogyha a virtuális gépet elindítjuk, mindenféle BIOS-beállításokat is módosíthatunk (természetesen csak a virtuális gép BIOS-ában). Ezt a környezetet a legjobban talán így tudnám szemléltetni: egy x86-os gép fut egy x86-os gépen. Akárhány virtuális gépünk lehet, azonban előre be kell állítanunk őket, lássunk is neki a munkának!

A program beszerzése

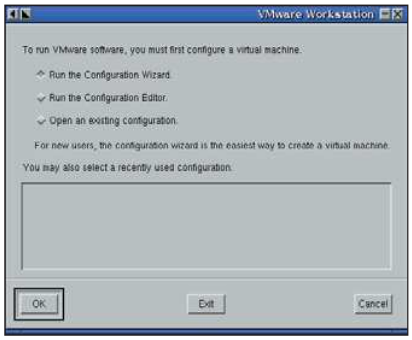
Ennek a programnak a 30 napos próbaváltozatát mindenki letöltheti a ➲ http://www.vmvare.com weboldalról. Ahhoz, hogy használni is tudjuk, be kell jegyeztetni magunkat. Ekkor egy ideiglenes számot kapunk, amivel 30 napig próbálgathatjuk, mit is tud a program, és eldönthetjük, hogy szükségünk van-e rá egyáltalán.

A telepítés

A program telepítése nagyon egyszerűen a letöltött fájl kicsomagolásával kezdődik, majd a létrejött ***vmware-distrib*** könyvtárba lépve rendszergazdai jogosultsággal futtassuk a `vmware-install.pl`-t. A telepítő nagyon jól kidolgozott, néhány kérdésre válaszolva újdonsült barátunkat már neki is ugraszthatjuk a vasunknak, hadd melegedjenek az alkatrészek!
A telepítésnél figyeljük arra, hogyha nem a támogatott Linux-változatok valamelyikét használjuk, akkor az éppen futó rendszermagunk forrása a ***/usr/src/linux*** könyvtárban legyen, továbbá a rendszermagfordításhoz szükséges eszközökkel is rendelkeznünk kell, mivel a telepítő a megfelelő rendszermagmodulokat megpróbálja elkészíteni.

Használatbavétel

Felhasználóként belépve a grafikus felületen adjuk ki a `vmware` parancsot. Itt meg kell adnunk az elektronikus levélben kapott sorozatszámot, ugyanis nélküle nem léphetünk tovább.
Miután ezekkel megvagyunk, egy kisebb



ablak ugrik elő (miként az első képünkön is látszik), ennek tartalma alapján három lépés közül választhatunk:

- Futtatjuk a varázslót, és a segítségével létrehozunk egy új virtuális gépet.
- Futtatjuk a beállítómenüt, amelyben elvégezhetjük a rendszerbeállításokat (a virtuális gépekhez tartozó memória stb.).
- Megnyithatunk egy már létező beállítást.

Mindenkinek javaslom a varázsló futtatását, mert a segítségével csak néhány kérdésre válaszolva (szinte csak kattingatva) állíthatjuk be az új gép paramétereit.

A varázsló

Az üdvözlőszövegen átlépve válasszuk a ***Create standard virtual machine*** menüpontot, és haladjunk tovább. E ponton egy szép listát kapunk arról, hogy milyen operációs rendszereket futtathatunk. Mint a 2. *képen* is látható, az MS teljes termékkínálatát használhatjuk, illetve a Linux, a FreeBSD és a NetWare 6.0 is számításba jöhet. Említésre méltó még az ***Other*** menüpont, ha ezt választjuk, elvileg bármilyen x86-os gépre írt operációs rendszert futtathatunk – néhány ötlet: BeOS, AtheOS, QNX, NetBSD és Sun Solaris, sajnos ezekhez nem áll rendelkezésre a VMWare-tools. Mivel a legtöbbet választott menüpont

az MS-termékek valamelyike, nézzük meg, miből is áll egy ilyen rendszer telepítése. Válasszuk ki a Windows 98 menüpontot (vagy amelyiket telepíteni szeretnénk), és lépjünk tovább! Adjuk meg a beállításunk nevét és azt a könyvtárat, ahová a fájlokat rakni szeretnénk. A következőkben állítsuk be a merevlemezt – ez lehet egy új, esetleg egy már régebben létrehozott virtuális vagy fizikai merevlemezrész is. Hozzunk létre egy új lemezrészt (***Create a new virtual disk***)! Úgy válasszuk meg, hogy minden olyan program kényelmesen elférjen rajta, amire szükségünk lesz. Engedélyezhetjük a CD-ROM-és a hajlékonylemezes meghajtó használatát, beállíthatjuk a hálózati kapcsolódás tulajdonságait, és ha mindezzel készen vagyunk, kattintsunk a ***Done*** gombra.
Ezzel elkészítettük első virtuális gépünket. A következő hónapban a rendszerek telepítésével ismerkedünk meg.

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

KAPCSOLÓDÓ CÍMEK

➲ http://www.vmware.com/
➲ http://www.vmware.com/download/

Séta a lemezes egységek körül (2. rész)

A számítógép adatokkal dolgozik, és ezeket az adatokat az úgynevezett beviteli-kiviteli eszközökön keresztül juttathatjuk a memóriába, illetve ezek segítségével ismerhetjük meg az adott művelet végeredményét. Azt is megállapítottuk, hogy a beviteli-kiviteli eszközöket nagyjából két csoportra oszthatjuk: karakteresekre és blokkosokra. A karakteres eszközök tárgyalásakor a terminálokkal foglalkoztunk, amelyek a felhasználó és a számítógép közötti kapcsolat megvalósításáért felelősek. A blokkeszközök közül a legjellemzőbbek a lemezek, amelyek adataink háttértárolására szolgálnak. Legutóbb a blokkeszközöket csak általában véve néztük meg, továbbá bemutattuk, hogy milyen elvet alkalmaz a Linux a blokkeszközök meghajtására. Itt is az eszközfüggetlenség elve érvényesült, tehát az volt a cél, hogy az operációs rendszer kódjának csak a lehető legkisebb része tudjon bármit is az eszköz kezelésének módjáról. A felsőbb rétegek az eszközöket csupán absztrakt utasítások formájában érhetik el. Ilyen utasítás például az „Olvasd be a 12. blokkot!", vagy a felhasználói szinten lévő programok esetében az „Írd ki ezt valamelyik állományba!".
Most továbbra is a blokkeszközökkel, pontosabban a lemezekkel foglalkozunk. Sorozatunk előző részében az eszközfüggetlen részt már alaposan áttanulmányoztuk, ám magáról a lemezkezelőről (ez tartja a kapcsolatot ténylegesen a lemezegységgel) kevés gyakorlati mozzanatot árultunk el. Ezért az elkövetkezendőkben tovább boncolgatjuk a lemezek felépítését és az operációs rendszer lemezkezelőjének a működését.
Sokakban az a kétely támadhat, hogy e cikk írója a terminálokat hátrányosan kezeli, ugyanis csak röpke két oldal erejéig foglalkozott velük, míg a lemezekkel két részen keresztül. Valójában nincs szó ilyesmiről. A fentiek oka egyrészt abban rejlik, hogy a karakteres eszközökkel az operációs rendszer könnyebben „elbír", mint a blokkosokkal. Másrészt ne feledkezzünk meg arról, hogy a terminál a felhasználóval áll közvetlen kapcsolatban. A felhasználó pedig nem eleve meghatározott jelenség, a lehető legelképzelhetetlenebb dolgokra képes. A terminálkezelő kódja és működése eléggé bonyolult, de erre azért is szükség van, hogy hatásosan meg tudja védeni a rendszert a felhasználó ténykedéseitől. Tehát bármit tegyen emberünk (akár csépelheti is a gombokat, vagy telemorzsázhatja a billentyűzetet), az ne legyen hatással a rendszer működésére (például ne tudja lefagyasztani magát a terminálkezelőt, ami rendkívül sok kellemetlenséggel járhat).
A lemezkezelőnek talán annyival könnyebb a dolga a terminálkezelővel szemben, hogy biztos lehet benne: az eszköztől az adatokat mindig ugyanúgy kapja vissza. Ennek köszönhetően meglehetősen kevés külső tényező befolyásolhatja a feladat végrehajtását, ezért nem is kell annyi óvintézkedést tennünk, mivel gépekkel és nem emberekkel van dolgunk. De ne higgyük, hogy egy lemezkezelő tervezése gyerekjáték. A nehézségek egész seregével kell megküzdenünk, és nem könnyű igazán jó megoldást találni.

A merevlemez

Az előző részben beszéltünk már ugyan a lemezek felépítéséről, de ahhoz, hogy még részletesebben bemutathassuk a merevlemez-kezelő működését, ki kell egészítenünk az ott leírtakat.
A merevlemezekről már tudjuk, hogy szektorokra, pályavonalakra (vagy sávokra), illetve cilinderekre tagolódnak. A pályavonal nem más, mint egy teljes körülfordulás alatt felírt bitsorozat, amely általában 512 bájt hasznos adatot tartalmazó szektorokból áll. A merevlemez több lemezt tartalmaz, amely egymás felett helyezkedik el, és mindegyikhez tartozik egy olvasófej (illetve kettő, mert a lemez mindkét oldalán lehet tárolni adatokat). Az olvasófejek mozgatókarjai rögzítve vannak egymáshoz, tehát egyik fej sem mozgatható külön-külön. Így mindig egymás alatt, minden lemeznek ugyanazon a pályavonalán tartózkodnak. Az egymás alatti pályavonalak együttesen egyetlen cilindert alkotnak. (Könnyű belátni, hogy a cilinderek száma megegyezik a lemezek sávjainak számával). Ezt a felosztást a merevlemez geometriai felépítésének nevezzük, amely nem minden esetben egyezik meg a valódi (fizikai) felépítéssel (lásd később).
Hogy a geometriai felépítést könnyebben megértsük, nézzük meg, miképp határozhatunk meg egy szektort a merevlemezen. Először is meg kell adnunk a cilinder számát, ami azt mondja meg, hogy az olvasófejeket hova kell mozgatnunk (és egyben azt is, hogy majd melyik pályavonalról szeretnénk olvasni). Ezután ki kell választanunk egy fejet (amellyel meghatározzuk, hogy egyrészt melyik lemezről, illetve annak melyik oldaláról szeretnénk olvasni). Legvégül át kell adnunk a szektor sorszámát. E három adat segítségével egyértelműen meghatározhatjuk a merevlemez bármelyik szektorát.
A mai merevlemezeknél a külső sávok több szektort tartalmaznak, mint a belsők. Ez lényegesen megnöveli a lemez tárolókapacitását, de bonyolultabb vezérlést követel. Szerencsére az operációs rendszernek ezzel nem kell foglalkoznia, ugyanis minden lemezegységhez tartozik egy vezérlő, ami tulajdonképpen egy kis számítógép saját processzorral és memóriával. Az operációs rendszer merevlemez-kezelője a vezérlővel tartja a kapcsolatot. A mai vezérlők programjai már nagyon bonyolultak és sok mindenre képesek, például saját gyorstárral rendelkeznek, illetve ha egy szektor meghibásodik, azt helyettesítik egy másikkal. (Minden pályavonalhoz tartozik pár tartalékszektor, amelyet kifejezetten ilyen célokra tartanak fenn. Ha egy szektor „beadja a kulcsot", a vezérlő a hibás szektor címét átirányítja az egyik tartalék szektorra, ennek köszönhetően a hibás szektorok elérhetetlenné válnak. Ez mind az eszköz szintjén történik, ezekről a dolgokról azonban az operációs rendszer az égvilágon semmit sem tud.)
A PC-s világban leggyakrabban használt lemezcsatolófelület az IDE (Integrated Drive Electronics, azaz beépített eszközelektronika), illetve az azt szinte már teljesen kiváltó EIDE (Extended IDE). Az IDE-lemezek a hajdani XT-hez készült Seagate 10 MB-os lemezéből fejlődtek ki, amelyeket az operációs rendszer BIOS-hívások segítségével tudott kezelni. Egy szektor eléréséhez a processzor regisztereibe be kellett töltenie a megfelelő fej-, cilinder- és szektorszámokat. Az IDE fejlesztői az együttműködés érdekében megtartották a BIOS-on keresztül zajló címzési módot.
A nehézség abban gyökeredzett, hogy egy szektor címzésében a fej megadásához 4, a cilinderhez 10, a szektorhoz pedig 6 bit állt rendelkezésre, azaz a BIOS segítségével csak olyan

merevlemezeket kezelhettünk, amelyben legtöbb 16 fej, 1024 cilinder és 63 szektor van (összesen körülbelül egymillió szektor). Ezzel nem is volt semmi gond, egészen addig, amíg meg nem jelentek az 528 MB-nál nagyobb tárterületű lemezek. Ezek 1024 cilindernél sokkal többet tartalmaztak, tehát már nem lehet őket a „hagyományos" módon címezni. Ezért a vezérlőket úgy módosították, hogy képesek legyenek a címek közötti bűvészkedésre: a BIOS-határon belüli értékeket fogadtak el, de azokat átszámolták maguknak a valódi fizikai értékre. Magyarán a felhasználó szemébe hazudtak, de ez legalább működő módszer volt, ha nem is túlzottan hatékony.

Nem sokkal ezt követően megjelent az EIDE, ami az IDE-vel szemben számtalan előnnyel rendelkezett. Lehetővé tette egy addig teljesen más címzési mód használatát is, ezt LBA-nak (Logical Block Addressing) hívták, ami nemcsak az 528 MB-os határ átlépését, de a szektorok egyszerűbb elérhetőségét is lehetővé tette. Az elv rendkívül egyszerű: a szektorokat beszámozzuk 0-tól 2^{24}-ig, a vezérlő pedig gondoskodik ennek az értéknek fej-, cilinder- és szektorszámmá való átalakításáról. Ez igazán kényelmes megoldás az operációs rendszer lemezkezelője számára. Az EIDE vezérlőhöz összesen négy meghajtót lehet csatlakoztatni, és a CD-ROM-meghajtókat is kezelni tudja. Manapság az EIDE-meghajtók olcsó áruk miatt nagyon elterjedtek, ezért szinte minden mai, Intel-kiépítésen is futó operációs rendszer támogatja (habár az EIDE ma már más számítógépeken is használatos). Mindezek ellenére az EIDE-lemezekkel számos gond akadt, ilyen például a viszonylag kis adatátviteli sebesség (ez például a videodigitalizáláskor jelenthet gondot), illetve az, hogy egyszerre csak egy eszköz működhet. A lemezek másik nagy családja, a SCSI azonban a gyorsabb átvitelt és több eszköz egyszerre való működését is lehetővé teszi.

A SCSI-ról csupán annyit említenénk meg, hogy ez több mint egy egyszerű vezérlő, valójában egy olyan adatsín, amelyre hét vagy tizenöt eszköz csatlakoztatható (például merevlemez, CD-író, lapolvasó vagy bármilyen más SCSI-s eszköz). A legtöbb ilyen egység külön kimenettel és bemenettel rendelkezik, ahol a kimenetet a másik bemenetére kell kapcsolni. A SCSI leginkább a hálózati kiszolgálóknál vagy a videodigitalizálásra használt gépekben elterjedt. Mi a továbbiakban az EIDE merevlemezkezelővel foglalkozunk, de a többi is ugyanezeket az alapelveket követi.

Folytassuk egy kis kitérővel. A PC-k esetében különösképpen fel kell készülnünk egy olyan nehézségre, aminek a gyökere magának a PC-nek az egyik jelentős tulajdonságából fakad. A PC ugyanis nem egy kézzelfogható számítógépfajta (mint például a Macintosh, ahol kapunk egy dobozt, amelybe minden elképzelhető dolog be van építve), hanem a különböző típusú egységekből felépülő számítógépek nagy családja, amelyek különböző processzorokat, sínrendszereket, memóriákat használhatnak. Gond abból fakad, hogy ezeknek az eszközöknek a programozása olykor gyökeresen eltérő lehet. Példaképpen megemlíthetjük a ma már múzeumba való XT-ket, amelyek 8-bites sínrendszerrel rendelkeztek. A 286-os, 386-os, 486-os és a Pentium kategóriás processzorokhoz azonban már az úgynevezett AT-s sín tartozik, ami 16 bites kapcsolódási felülettel bír. Ma pedig már a 32-bites PCI-sínek a legelterjedtebbek. A legtöbb számítógépben található egy úgynevezett gyári program (firmware), amely az alaptesten (PC-sen szólva az alaplapon) található, csak olvasható memóriában helyezkedik el. Ez tulajdonképpen nem más, mint a BIOS.

A BIOS egyik feladata az, hogy olyan eljárásokat nyújtson az operációs rendszer számára, amelyeket használva az figyelmen kívül hagyhatja az eszköz ehhez hasonló sajátosságait. Ez nagyon örvendetes, ugyanis – ha például a merevlemez-kezelést vesszük alapul – az operációs rendszernek, a BIOS merevlemezkezelő eljárásait használva, elvben nem kell olyannal foglalkoznia, hogy az adott vezérlő most XT-s vagy AT-s, vagy akármilyen más típusú, valószínűleg minden működni fog (el kell ismernünk, ez a példa egy kicsit sántít, ugyanis aki ma XT-n szeretne Linuxot futtatni, az „kemény arc"; akinek még sikerül elindítania is, az vagy hazudik, vagy valamiféle nagymester, mivel a Linux nem hajlandó védett módot nem támogató processzorokon futni; de azért vannak olyan operációs rendszerek – például a MINIX –, amelyek, ha nagyon muszáj, valós módban is elfutnak).

A BIOS-szal viszont alapvető gondok vannak, például az, hogy az MS-DOS-hoz fejlesztették ki, amely köztudottan 16-bites és nem is többfeladatos. Mi most azonban védett módban futó, többfeladatos operációs rendszerekben gondolkozunk. Ha az ilyen rendszerek a BIOS segítségével szeretnék a beviteli-kiviteli kapukat elérni, szembe kell nézniük azzal, hogy a processzort minden BIOS-hívásnál át kell kapcsolniuk valós módra, majd ismét védett módba. Az ilyen ide-odakapcsolgatás pedig iszonyatosan lassú művelet. Ezért ha valaki még azt is szeretné, hogy a rendszere valamennyire gyors legyen, úgy, ahogy van, el kell felejtenie a BIOS-t, és a beviteli-kiviteli eszközök kezelését saját magának kell elvégeznie.

Azért a BIOS-nak óriási szerepe van a gép elindításánál: sok operációs rendszer indítóprogramja is a BIOS-t hívja segítségül, hogy a rendszermagot a merevlemezről a memóriába töltse. Utána viszont már semmi szükség sincs rá; ma már legfeljebb az elavult DOS-os alkalmazások használják a BIOS eljárásait (de még azok sem mindig, mert iszonyatosan lassú).

El is érkeztünk igazi kérdésünkhöz: hogyan fér meg a rendszermagban például két merevlemez-kezelő? Értelemszerűen csak az egyikre lesz szükség, de valamilyen úton-módon el kell dönteni, hogy melyikre. Összesen három lehetőségünk van. Az első: a rendszermagot úgy fordítjuk le, hogy csak azokat a kezelőket tartalmazza, amelyekre valóban szükség lesz. A második és a harmadik lehetőségnél az összes létező támogatást befordítjuk, de míg az egyiknél a döntést a felhasználóra bízzuk, a másiknál maga a rendszermag próbálja kipuhatolni, hogy melyik kezelő is illik az adott eszközhöz.

Mind a három módszernek egyaránt vannak előnyei és hátrányai. Az elsőnél például semmi ilyen jellegű gond nincs, de az adott rendszermag kizárólag a mi beállításainkkal megegyező gépeken lesz hajlandó futni. Az sem feltétlenül jó, ha a választást a felhasználóra bízzuk, mivel nem biztos, hogy szakértője a gépben található eszközök pontos típusainak. Elengedhetetlen tehát, hogy az operációs rendszer önmaga is képes legyen felismerni a gépünkben található beviteli-kiviteli eszközöket, és több kezelő esetén ki tudja választani a megfelelőt. (Ám a PC-k esetében ez az „eszközfelismerősdi" sem olyan egyszerű, ugyanis a piacon rengeteg, a szabványnak csak részben megfelelő termékkel találkozhatunk. Amikor egy eszközt próbálunk felismerni, első lépésként a beviteli-kiviteli kapukat kezdjük el letapogatni. De ez nagyon veszélyes játék! Például véletlenül működésbe hozhatunk egy olyan eszközt, amelynek helytelen beállítása következtében az egész rendszer kiakadhat, és már csak a RESET segíthet rajtunk. Ezt elkerülendő a Linux, a Windows és a többi rendszer is bizonyos biztonsági eljárásokat alkalmaz. Ha például az eszköztől sokáig nem érkezik értékelhető válasz, az adott folyamat azonnal megsemmisül.)

De térjünk vissza a lemezkezelőre! Az előző részben említettük, hogy az eszközfüggő lemezkezelő a felsőbb rétegektől hatféle

utasítást kaphat. Ezek a következők voltak: DEV_OPEN, DEV_CLOSE, DEV_IOCTL, DEV_READ, DEV_WRITE és SCATTERED_IO. Most megnézzük, mit is jelentenek ezek az utasítások a merevlemez esetében!
A DEV_OPEN az eszköz első elérésekor hívódik meg. Feladata többnyire csak annyi, hogy beállítson bizonyos alapértékeket, amelyekre szükség lesz az eszközről történő olvasáshoz, illetve az arra való íráshoz – ilyenek például a merevlemezen található lemezrészek. A DEV_CLOSE például a CD-ROM-oknál lehet hasznos, segítségével az eszközt rábírhatjuk, hogy dobja ki a CD-t. A DEV_IOCTL a merevlemezek esetében a lemezfelosztási tábla módosításával foglalkozik, de például a CD-ROM esetében beállít egy értéket, ami megmondja a DEV_CLOSE-nak, hogy ki kell-e adnia a korongot a meghajtóból. A DEV_READ-hez és a DEV_WRITE-hoz csupán annyit fűznénk hozzá, hogy a beérkező kérések végrehajtásakor a kezelő nem alkalmazza az előző részben bemutatott liftes algoritmust. Helyette egy rendezési eljárást alkalmaz, ami összeszedi az egymás utáni szektorokra vonatkozó kéréseket. Ez azért jó, mert a merevlemezről az esetek többségében a fájlrendszer olvasni szeretne, és a művelet az esetek többségében több egymást követő szektor beolvasásával jár. Erre a fájlrendszer bemutatásánál még részletesebben visszatérünk.
A DEV_READ és a DEV_WRITE műveleteknél a feladat elvégzése után mindig hívódik egy befejező eljárás. De mivel gyakran van szükség egymás utáni szektorok beolvasására, illetve írására, a SCATTERED_IO nélkülözhetetlen a lemezműveletek felgyorsításához. Segítségével ugyanis egyszerre több, egymást követő szektort is beolvashatunk, de az a bizonyos befejező eljárás csak egyszer hívódik meg, miután már mindent beolvastunk, illetve kiírtunk, amit csak kellett.

Hajlékonylemezek

A hajlékonylemez-meghajtók jóval egyszerűbb felépítésűek, mint a merevlemezek, kezelésük azonban nagyobb kihívást jelent az operációs rendszer számára. Ennek oka több dologra is visszavezethető.
Először is a viszonylag egyszerűbb felépítésű eszközhöz egyszerűbb vezérlő is társul. Míg a merevlemezek esetében az operációs rendszer a bonyolult és összetett műveleteket a vezérlőre bízta, sok mindenről saját magának kell gondoskodnia. Vegyük például az olvasófejnek egyik pályavonalról a másikra történő átmozgatását! A merevlemezek esetében az olvasófej pozicionálásával nem kell foglalkoznunk, mivel a vezérlő azt elintézi helyettünk. Sőt egyáltalán nem biztos, hogy a cilinder-szektor-fej felosztása megegyezik a lemez fizikai felépítésével, például könnyen elképzelhető, hogy a külső cilindereken valójában több szektor található, mint a belsőkön. Vagy gondoljunk csak az LBA-ra, amikor is a vezérlő számolja ki a megadott logikai címből, hogy pontosan hová is kell mozgatnia az olvasófejet. Hajlékonylemezek esetében azonban nem is álmodhatunk az LBA-hoz hasonló segítségről, és a fej pozicionálásához sem kapunk sok segítséget a vezérlőtől. Itt az operációs rendszernek közvetlenül meg kell hívnia a SEEK nevű műveletet, amely a fej pályavonalak közti mozgatására szolgál. A SEEK művelet értékeként azt kell megmondanunk, hogy a fejet melyik irányban hány pályavonallal arrébb szeretnénk tuszkolni. Ha a SEEK valamilyen oknál fogva nem járna sikerrel (például nem arra a pályavonalra kerül, ahová kell), a RECALIBRATE nevű utasítás végrehajtására is szükség van, ami a fejet kiviszi a 0. pályavonalra, majd onnan ismét elkezdhetjük a megfelelő helyzetbe való állítást.
Ezenkívül van még néhány tényező, ami nagyban bonyolítja a hajlékonylemez kezelését, de erről nem az eszköz csekély tudású vezérlője tehet. Ilyen tényező, hogy a hajlékonylemez cserélhető eszköz. Ez önmagában nem is lenne baj, csak az a gond, hogy nem tudjuk érzékelni, ha valaki két művelet között orvul kicseréli a meghajtóban lévő lemezt. Magyarán nem biztosíthatjuk, hogy a kiírandó adat arra a lemezre kerül, amelyre azt szántuk. (Felmerülhet a kérdés, hogy mi a helyzet a CD-ROM esetében, mivel az is cserélhető eszköz. Ez igaz, de ott nem léphet fel ilyen nehézség, mivel az egység vezérlője érzékeli, ha kinyitják a meghajtó ajtaját. Erre a hajlékonylemez-egység vezérlője sajnos nem képes, ha tudná, nem lenne semmi gond). Mivel azt sem tudjuk megállapítani, hogy a meghajtó tartalmaz-e éppen lemezt vagy sem, további fejfájást okozó helyzetekkel is számolnunk kell. Mi történik például, ha a felhasználó kísérletet tesz a hajlékonylemez elérésére, ám ezt megelőzően elfelejtette a meghajtóba helyezni? Valószínűleg sokan tapasztalták már az ebből adódó kellemetlenségeket: az egész rendszer addig felfüggesztett állapotba kerül, amíg a meghajtó bőszen a bent nem lévő lemez elérésével próbálkozik. Ezért van szükség különböző óvintézkedésekre, ha például a kiadott művelet bizonyos idő alatt nem fejeződik be, az egészet gyorsan el kell felejtetni a lemezkezelővel.
Ha már a meghajtóba helyezhető lemezekről volt szó, érdemes megemlítenünk, hogy két típusú hajlékonylemezt ismerünk: az 5,25 és a 3,5 hüvelykeset (inch). Ezek közül csak az 5,25-öst tudjuk hajlítani, mivel a másikat egy merev külső tok védi. A „hajlékonyabbat" ma már szinte sehol sem használják, de minden operációs rendszernek tudnia kell kezelni. Hogy mégse legyen ennyire egyszerű az életünk, mind a kétféle lemezből van kisebb (LD – Low-Density) és nagyobb (HD – High-Density) sűrűségű is, amelyek más-más kapacitásúak (a 3,5 hüvelykes LD lemezek például 720 KB-osak, míg a HD-sek 1,44 MB-osak). Ez a sokféle lemezformátum annyiban kellemetlen a lemezkezelő számára, hogy meg kell tudnia állapítani, hogy a felhasználó éppen milyen lemezt helyezett be. Ennek megállapítására több módszer is kínálkozik, például a magasabb számozású szektorok és pályavonalak útján. Akárhogy is, a formátum megállapítása időigényes feladat, sőt ha a lemez hibás szektorokkal van teletűzdelve, az sem biztos, hogy felismerhető.
A merevlemezekkel ellentétben itt az olvasófej hozzáér a lemez felületéhez, és a súrlódás miatt hamar elkophat. Ezért a lemezt forgató motort csak akkor szabad bekapcsolni, amíg lemezművelet zajlik, egyébként érdemes leállítani, és a fejet visszahúzni. Csakhogy a lemeznek a megfelelő fordulatszámra való felpörgetése és a fej előrehúzása körülbelül fél másodperces művelet. Ezért nem jó megoldás, ha a motort minden lemezművelet után kikapcsoljuk. A lemezkezelők éppen ezért úgy működnek, hogy a művelet után egy kis ideig még bekapcsolva hagyják a motort, és csak akkor kapcsolják ki, ha adott időn belül nem érkezett további kérés.
Most már kétség sem férhet hozzá, egy hajlékonylemez-kezelő tervezése nem éppen az élet örömteli feladatai közé tartozik. Még egy jelentős beviteli-kiviteli eszközről nem ejtettünk szót, mégpedig az óráról. Elsőre kicsit meglepő, hogy az óra is B-K eszköz. Pedig az, mi több, nélkülözhetetlen eszköz az operációs rendszer számára. Hogy miért, erről a következő hónapban írunk. Az óra bemutatása után befejezzük a B-K eszközöket, és a memóriakezeléssel foglalkozunk.

Garzó András (garzoand@interware.hu)
Körülbelül három éve foglalkozik Linux- és más Unix-rendszerekkel. Legjobban az operációs rendszerek lelkivilága érdekli, de nyitott egyéniség. Kedvenc étele a palacsinta, és van egy Richard nevű macskája. Minden észrevételt, megjegyzést, levelet szívesen fogad.

Linux és játék

Ma már senki számára sem kétséges, hogy a Linux igenis teljes értékű operációs rendszer, ami hatékonyan munkára fogható, de az élet nem csupán ebből áll.

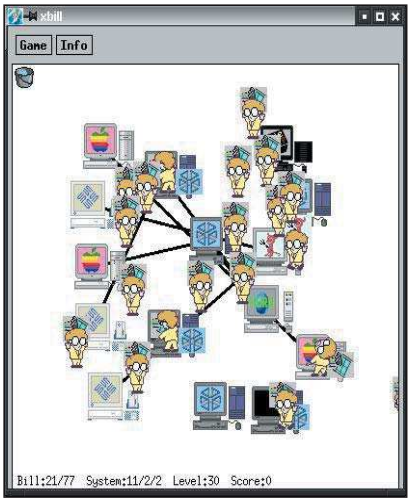
Elég csak végiglapozni magazinunk bármelyik számát, hogy lássuk, hány különböző területen alkalmazzák sikerrel. Néhányan ezt a tényt úgy értékelik, hogy a Linux csak munkára való, és például játékprogramok futtatására alkalmatlan.
Ez azonban téves elképzelés. Sőt ami azt illeti, Linux alá már régóta elérhetők azok az eszközök (például OpenGL), amelyek a játékprogramok fejlesztéséhez elengedhetetlenek. Az sem mondható tehát, hogy Linux alá nehezebb a játékkészítés, mint más rendszerek esetében.
A gond inkább abból fakadt, hogy a Linux sokáig kiesett a játékfejlesztők érdeklődési köréből. Így a mindenre elszánt „gamerek" valóban arra kényszerültek, hogy még valamilyen ablakos operációs rendszernek is helyet szorítsanak gépükön (vagy valamelyik Windows-emulátort üzemeltessék). Szerencsére ma már nem egészen ez a helyzet. Egyre több cég ismeri fel a Linuxban rejlő lehetőségeket, így a „játékgyárosoknak" is megéri erre a felületre fejleszteni. Mostanra rengeteg népszerű játéknak készült el a linuxos változata (lásd később), és talán arra sem kell olyan sokat várnunk, hogy minden játékot úgy dobjanak piacra, hogy az Windowson és Linuxon egyaránt futtatható legyen.

Az OpenGL

Mivel nagyon sok játék (illetve más 3D-s alakzatokkal dolgozó alkalmazások) az OpenGL-re támaszkodnak, nem árt, ha kicsit körbejárjuk, mi is ez valójában.
Az OpenGL-t a Silicon Graphics (SGI) cég kezdte el fejleszteni valamikor a 90-es évek legelején, igaz, akkor még IrisGL néven. A cél az volt, hogy az általuk gyártott grafikus munkaállomásokra (amelyek már akkor híresek voltak arról, hogy rendkívül gyorsan tudtak grafikai számításokat végezni, például mátrixátalakításokat) megkönnyítse az alkalmazások fejlesztését. Később felmerült az igény, hogy az IrisGL segítségével készített programokat más rendszerekre is át lehessen vinni. Ennek köszönhetően elindult az OpenGL nevű projekt, ami az IrisGL-re épült, de teljesen rendszerfüggetlen volt. Ma már az operációs rendszerek széles köre alá elérhető, beleértve a Linuxot, a Windowst és a MacOS-t is.
Az OpenGL tulajdonképpen egy függvénygyűjtemény, mely segítséget nyújt a két-, illetve háromdimenziós alakzatok létrehozásában, és az azokkal történő műveletek elvégzésében. Aki például egy 3D-s lövöldözős játék írására adja fejét, jelentős munkától kímélheti meg magát, ha az OpenGL eljárásait és tagfüggvényeit használja, és nem saját maga készíti el őket.
A legfontosabb dolog, amit az OpenGL fejlesztői szem előtt tartanak, az a felületfüggetlenség. Ez abból a szempontból nagyon jó, hogy ha valaki megír mondjuk egy játékprogramot Windowsra, az viszonylag kevés munkával átírható egy teljesen más operációs rendszerre, esetleg Linuxra. A felületfüggetlenségnek viszont ára van! Az OpenGL például nem foglalkozhat a beviteli eszközök kezelésével, ez tehát teljes egészében a programozóra hárul. Továbbá az ablakozás megvalósítása sem tartozik a feladatkörébe, mivel ez is operációsrendszer-függő. (A Linuxban az ablakok kezelése az X Window feladata, míg a Windowsban ezért maga az operációs rendszer felel).
Az OpenGL az X Windowhoz hasonló kiszolgálóalapú felépítést követi, amely szintén a rendszerfüggetlenséget segíti elő. Ennek köszönhetően az is megvalósítható, hogy a végeredményt egy másik gép számolja ki, nem pedig az, amelyik a grafikus alkalmazást futtatja.
Az OpenGL-t minden Linux-terjesztés tartalmazza, és általában együtt települ az X-szel.
Ha azt szeretnénk, hogy a játékok igazán gyorsak legyenek, egy „OpenGL gyorsításra" is szükség lesz. Ez tulajdonképpen nem jelent mást, minthogy lehetővé tesszük az OpenGL eljárásai számára a grafikus eszköz közvetlen elérését. Ennek egyik módja a közvetlen leképező rendszer (Direct Rendering Infrastructure – DRI) használata, amelyet a 4-es X már alapból tartalmaz. Ha nVidia, 3DFX, Matrox vagy ATI kártyával rendelkezünk, érdemes a saját meghajtóikat (ezek a gyártó honlapjáról letölthetők) erre a célra használnunk. Ez általában két részből áll: egy rendszermagmodulból (ami a rendszermag szintjén történő támogatást biztosítja) és egy X-kiegészítésből. A telepítés módjáról természetesen részletes leírást kapunk. Ezek után nézzünk meg néhány népszerű linuxos játékot!

Támad a Bill: minden ellenállás hasztalan

A terjesztések rengeteg kisebb játékot tartalmaznak az X-hez, a KDE-hez, illetve a Gnome-hoz. Ezek általában kártya-, logikai vagy ügyességi játékok. Közülük



egyet emelnénk ki: az xbillt. Ezzel a játékkal vigyázni kell, mert ha nem vagyunk elég körültekintőek, komoly lelki és fizikai függőséget válthat ki!
Feladatunk az, hogy megvédjük a ránk bízott számítógépeket egy csapat Billtől, akiket nem kisebb cél hajt, minthogy az összes masinánk operációs rendszerét Windowsra váltsák át. Ezért ne kíméljük őket, egerünk segítségével üssük is szét őket mind egy szálig! Ha valamelyik gépre mégis felkerülne a Windows, még nincs veszve semmi, visszatehetjük rá a régi rendszert, de figyeljünk arra, hogy mely gépek vannak hálózatba kötve, ugyanis a Windows a hálózaton keresztül is képes terjedni, és a többi masinára is átkerülhet!
A nehézségi fokozat emelkedésével nő a gépek és a támadó Billek száma is. A 30-as fokozattól kezdve már egy egész Billhadsereggel kell felvennünk a versenyt, ami jócskán megterheli „kattintóujjunk” izmait és egerünk állóképességét.

Chromium

Aki egy kis ész nélküli lövöldözésre vágyik, annak mindenképp ajánljuk, hogy vessen egy pillantást a Chromium nevű alkotásra. Ez is egy ingyenes játék, számos terjesztésnél (például a Red Hatben) „alapfelszerelésnek” számít.
Se a grafikára, se a hangra nem lehet panaszunk, az irányítás elsajátítása sem okozhat gondot. Amire azonban nem árt figyelni, hogy egy ellenséges repülő micsodát se engedjünk a hátunk mögé, mert akkor rögvest elvesztünk egy életet. Az időnként felbukkanó pingvinemblémákat és halálfejeket érdemes felszednünk, mivel az előbbi életerőt (illetve pajzsot), az utóbbi pedig fegyvert rejt számunkra. Összesen kilenc pályán kell átverekednünk magunkat, és mindegyik végén ott vár minket egy főellenség, aki nem éppen barátságos fickó, mivel orrán-száján át halálos sugarak ömlenek ki.
A játék még az egyszerűbb fokozatokon is kihívást jelent (e cikk írójának például máig nem sikerült a harmadik pályánál továbbjutnia), mindazonáltal a feszültségek levezetésére tökéletes.

TuxRacer

Nem is lehet kétséges: a játék fejlesztői igazi Linux-„őrültek”, ugyanis főhőséül nem mást, mint magát Tuxot, a Linux hivatalos pingvinjét választották. A feladat annyi, hogy a hegyoldalról lefelé hason csúszva minél előbb célba érjünk, mindeközben minél többet begyűjtsünk az útközben fellelhető heringekből. (Az újabb változatban Tuxon kívül további három karakter is rendelkezésünkre áll: egy jegesmedve, egy fóka és egy rózsaszínben pompázó pingvinlány).
A játék egyébként Windowson is elérhető. A teljes változat sajnos nem ingyenes, de egy játszható bemutató letölthető, ami nem is igazán csak bemutató, mivel rengeteg pálya választható, amelyek teljesítése beletelik néhány téli éjszakába. A játék hivatalos honlapja a ➲ http://www.tuxracer.com címen érhető el.

Loki-játékok

A Loki Games egy olyan vállalat volt, amelyik felvállalta, hogy a legnépszerűbb, eredetileg Windowsra készült játékokat Linuxra ülteti át, bizonyítva, hogy kedvenc operációs rendszerünk nem csak kiszolgálónak, illetve kemény munkára alkalmas. Ez a cég azonban anyagi nehézségek miatt sajnos kimúlt.
A régebben fejlesztett játékok közül igen bő választék áll a játékmegszállottak rendelkezésére, mint például a Heroes of Might and Magic 3, Civilization: Call to power, Alpha Centauri, Tribes 2, Rune, Solider of Fortune, Unreal Tournament, SimCity 3000, Heretic 2, Myth 2, RailRoad Tycoon 2 és a sort még folytathatnánk.
Természetesen az itt felsorolt játékok linuxos változatait sem adják ingyen, mindenesetre játszható bemutatójuk a Loki honlapjáról (➲ http://www.lokigames.com/) letölthető.
A játékok eszközigénye többé-kevésbé megegyezik a windowsos változatok igényeivel. Futtatásukhoz szükség lesz még legalább egy 2.2.x-es rendszermagra, XFree86-3.3-ra és a glibc-2.1 vagy frissebb változatára. Az újabb (vagy a legfrissebbtől számítva egy párral régebbi) terjesztések ezeket minden bizonnyal biztosan tartalmazzák.
A hangkártya megszólaltatását az OSS-re bízzák, tehát nem árt, ha zenélőeszközünket a Linux-rendszermag is támogatja.

Garzó András (garzoand@interware.hu)
Körülbelül három éve foglalkozik Linux- és más Unix-rendszerekkel. Legjobban az operációs rendszerek lelkivilága érdekli, de nyitott egyéniség. Kedvenc étele a palacsinta, és van egy Richard nevű macskája. Minden észrevételt, megjegyzést, levelet szívesen fogad.

OpenACS-lapok létrehozása

Reuven bemutatja, hogyan fejlesszünk APM-mel saját web- és adatbázis-alkalmazásokat.

A múlt hónapban folytattuk OpenACS-felfedező körutunkat és megnéztük, hogyan kell az alkalmazásokat csomagokba rendezni az APM (ArsDigita Package Manager) segítségével. Minden OpenACS-alkalmazás általában adatbázistáblákat és kiszolgálóoldali programokat tartalmaz, következésképpen a csomagok telepítése és frissítése a fájlok egyszerű másolásával nem oldható meg. Mint a múlt hónapban láthattuk, az APM alkalmazása leegyszerűsíti a csomag telepítését és különféle címek alá történő helyezését.
Ez nagyszerű dolog, amennyiben minket csak a már létező APM-ek érdekelnek. A legtöbb OpenACS-telepítés azonban új, egyedi csomagokat is tartalmaz, amelyek saját adatmodellel és programmal bírnak. Bár elméletileg APM nélkül is lehetséges OpenACS-alkalmazásokat fejleszteni, ez meglehetősen megnehezíti programunk terjesztését, a változatkövetést vagy a telepítés menetének szabványosítását.
Ebben a hónapban azzal foglalkozunk, hogyan fejleszthetünk saját web-, illetve adatbázis-alapú alkalmazásokat az APM segítségével. A végeredmény egy olyan alkalmazás lesz, amit bárki be tud tölteni a saját OpenACS-rendszerébe.

Vázcsomag létrehozása

Az új OpenACS-alkalmazás létrehozásának első lépése egy új vázcsomag készítése a webalapú APM programmal. Alapértelmezés szerint ez a program az OpenACS-rendszereken csak a rendszergazdai jogosultságokkal rendelkező felhasználók számára érhető el, így elképzelhető, hogy a fejlesztési munka megkezdése előtt meg kell kérnünk a rendszergazdát, módosítsa a jogosultságainkat.
A legtöbb OpenACS-rendszeren az APM program a */**acs-admin**/**apm**/* URL alól indul. A program az összes telepített APM-et bemutatja, megjelenítve valamennyi nevét, változatát és az alkotó fájlok számát. A fájlszámlálásba általában az ***.sql***- (adatbázis-meghatározások létrehozására és eltávolítására használt) fájlok, a ***.tcl***- (a Tcl programkódot tartalmazó) fájlok, az ***.adp***- (ASP vagy JSP-szerű websablon) fájlok és a ***.xql***- (SQL-lekérdezéseket tartalmazó) fájlok számítanak bele. Természetesen maga az APM más fájlokat, képeket, szöveget vagy valami szokatlanabb dolgot is tartalmazhat, például Flasht.
Mint azt a múlt hónapban láthattuk, ezt a képernyőt használhatjuk kezdőképernyőként rendszerünk csomagjainak telepítésénél, vizsgálatánál és módosításánál. Ha azonban a lap aljára megyünk, és a ***create a new package*** hivatkozásra kattintunk, elkezdhetjük új alkalmazásunk létrehozását.
A ***create a package*** kezdőképernyője számos jellemző megadását kéri tőlünk, amelyek a csomagokat leíró ***.info*** fájl létrehozásában segítik az APM-et. Feltételezve, hogy mi csak egy ***Szia, világ*** alkalmazást szeretnénk a lehető legkevesebb munkával előállítani, csak kevés mezőt kell ténylegesen kitöltenünk:

- A csomagkulcs (package key) legyen `atf-szia`. Az APM-ek esetében nincs kötelező névtérszerkezet, de a legtöbb fejlesztő saját nevet (vagy hasonló azonosító jelet) használ a csomag neve előtt az – esetleges ütközéseket elkerülendő.
- A csomag neve lehet ***Szia*** vagy tetszés szerint bármi más; a fejlesztők ezt használják arra, hogy a sajátjukat a többi csomagtól megkülönböztessék.
- Alkalmazást (application), és nem szolgáltatást (service) fejlesztünk.
- A változatszáma legyen 0.1d, jelezve, hogy a fejlesztés korai szakaszában járunk.
- Az összefoglalás (summary) és a leírás (description) mezőket tetszés szerint kitölthetjük vagy üresen hagyhatjuk. Természetesen valós alkalmazás készítésekor nem árt ezeket a mezőket is kitöltenünk.

Ha végeztünk, győződjünk meg róla, hogy a ***write a package*** (csomagírás) jelölőnégyzet be van-e kapcsolva, majd kattintsunk a ***create package*** (csomag létrehozása) gombra a lap alján. A megfelelő lap kezelőlapjára fogunk jutni. Ha belepillantunk az OpenACS-eszközkészlet telepítési útvonalának ***packages*** alkönyvtárába, ott egy érvényes XML formátumban íródott ***atf-szia.info*** fájlt tartalmazó ***atf-szia*** könyvtárat fogunk találni.

Adatmodell létrehozása

Most, hogy az OpenACS már felismeri a csomagunkat, elkezdhetünk dolgozni lapunk adatmodelljén. Mint azt minden tapasztalt web- és adatbázis-fejlesztő tudja, a táblák megtervezése a feladat legjava, az adatot törlő, létrehozó és módosító alkalmazások megírása már meglehetősen egyszerű.
Alkalmazásunk egy egyszerű vendégkönyv lesz, ahol a látogatók rögzíteni tudják a lappal kapcsolatos észrevételeiket. (Az OpenACS már eleve rendelkezik hasonló, de sokkal hatékonyabb és kifinomultabb megoldással, ami a honlap bármely lapján működik; ezt azonban most figyelmen kívül hagyjuk, minthogy célunk éppen a csomaglétrehozás gyakorlása.) Adatmodellünk a következőképpen fog kinézni:

```
CREATE TABLE atf_szia_postings (
  posting_id   SERIAL    NOT NULL,
  user_id      INTEGER   NOT NULL  REFERENCES
               ➥users
               ON DELETE CASCADE
               ON UPDATE CASCADE,
  entry_date   TIMESTAMP  NOT NULL  DEFAULT
               ➥NOW(),
  posting      TEXT       NOT NULL
               CHECK (posting <> ''),
  PRIMARY KEY(posting_id)
);
```

Tisztán szakmai szempontból a fenti táblameghatározás valószínűleg bárki számára elfogadhatónak tűnik, aki már dolgozott korábban PostgreSQL alatt. A `posting_id`-t

használtuk egész elsődleges kulcsként, az `user_id` lesz a külső kulcsunk, van egy `timestamp` mezőnk (ez tartalmazza a dátum- és időadatokat), illetve egy szöveges mezőnk az üzenet tárolásához. Figyeljük meg, hogy a táblanév ***atf_szia*** előtaggal kezdődik, jelezve, hogy az ***atf-szia*** APM része.
A táblanevek és a csomagnevek egységes elnevezése a névtérrendszerezés egyik kezdetleges formája, de elég jól működik, amíg mindenki tartja magát hozzá.
A fentiek jól működnek PostgreSQL alatt, de korántsem biztosak Oracle-lel. Tudván, hogy az OpenACS-közösség mindig arra törekedett, hogy mindkét adatbázis alatt átlátszó módon dolgozhasson, mit tegyünk, hogy ne bosszantsuk fel nagykiszolgálós munkatársainkat?
A válasz a következő: amikor az ***atf-szia*** csomagot feltelepítjük, az APM egy ***sql/atf-szia-create.sql*** nevű fájlt fog keresni. Ha létezik ilyen nevű állomány, feltételezi, hogy minden támogatott adatbázis-kezelőn működik. Ha nem létezik, az APM megnézi, van-e ***postgresql*** és ***oracle*** nevű alkönyvtár, és a megfelelő könyvtárban végrehajtja az ***atf-szia-create.sql*** állományt. Ha tehát rendszerünk PostgreSQL-t használ, a fenti SQL-t a ***sql/postgresql/atf-szia-create.sql*** néven kellene mentenünk. A hivatalos OpenACS-csomagok elvileg Oracle és PostgreSQL alatt egyaránt működnek, tehát elég ritkán találni olyan csomagot, amelyik csak az egyik vagy a másik ág alatt futna (az e havi példák szavatoltan csak PostgreSQL alatt működnek, bár Oracle alá történő átvitelük sem lenne túl bonyolult).
Az OpenACS tisztogató (cleanup) parancsfájlok létrehozását is lehetővé teszi ***sql/CSOMAG-drop.sql*** néven – ez a létrehozó parancsfájl által készített valamennyi táblameghatározást és tárolt függvényt eltávolítja, e célból hozunk létre egy ***sql/postgresql/atf-szia-drop.sql*** nevű fájlt is.
Az APM a csomag első telepítésekor már tudni fogja, hogyan készítse az adatmodellt, de a fejlesztés időszakában erre még nem képes. Az ***atf-szia*** adatbázisba illesztését tehát nekünk kell kézzel elvégeznünk:

```
psql -f atf-szia-create.sql openacs4
```

Természetesen itt most feltételeztük, hogy az OpenACS adatbázis neve ***openacs4***, a PostgreSQL-kiszolgáló és a psql-ügyfél azonos gépen fut, illetve, hogy pillanatnyi felhasználónevünk a megfelelő adatbázis-hozzáférési jogosultságokkal bír.

OpenACS-sablonok

Most, hogy adatmodellünk elkészült, ideje elkészíteni azt az alkalmazást is, ami majd használni fogja. Az OpenACS 4-ben jelent meg az ADP (amely az AOLserver ASP/JSP megfelelője) alapú sablonozórendszer, amelyet az OpenACS egyik legjobb részének tartok.
Az ASP-szerű lapokat a nem programozók könnyebben megértik, mint a hagyományos kiszolgálóoldali programokat. Ugyanakkor a HTML-kód hibrid lapok szűk keresztmetszetté válhatnak, hiszen a tervezők és a fejlesztők nem tudnak egyazon fájlon egyidőben dolgozni.
E kihívásra az OpenACS-sablonok üdítő megoldást kínálnak. A lapot két részre osztjuk: az egyik (a ***.tcl***-lap) a programozóké, a másik (az ***.adp***-lap) a tervezőké. A ***.tcl***-lap rövid összefoglalóval kezdődik, ahonnan megtudhatjuk, milyen értékeket vár, illetve milyen értékeket fog átadni az ADP-lapnak. A Tcl-lap az `ad_return_template` hívással fejeződik be, ami kikeresi az azonos nevű ***.adp***-lapot, behelyettesíti a megfelelő változóértékeket, majd lefuttatja a lapon az ADP-értelmezőt.

1. lista posting.tcl

```
ad_page_contract {

    @author Reuven M. Lerner
(<reuven@lerner.co.il>)
    @creation-date 2002-September-18

} {

} -properties {
  postings:multirow
}

# Az eddigi  sszes  zenet megszerzØse

set sql "SELECT PE.first_names || ' ' ||
          PE.last_name
          as user_name,
             PA.email, PO.posting,
             PO.entry_date
          FROM Persons PE, Parties PA,
            atf_szia_postings PO
          WHERE PE.person_id = PA.party_id
          AND PE.person_id = PO.user_id
          ORDER BY PO.entry_date DESC"

db_multirow postings get_postings $sql

ad_return_template
```

A Tcl-lap adatokat adatforrások alakjában tud átadni az ADP-lapnak, ami valójában a változó csinosabb neve. Ha a Tcl-lapon ezt látjuk:

```
set five 5
```

az ADP-lapon bárhol elővehetjük ezt az értéket, ha a változó nevét @ jelek közé helyezzük, valahogy így:

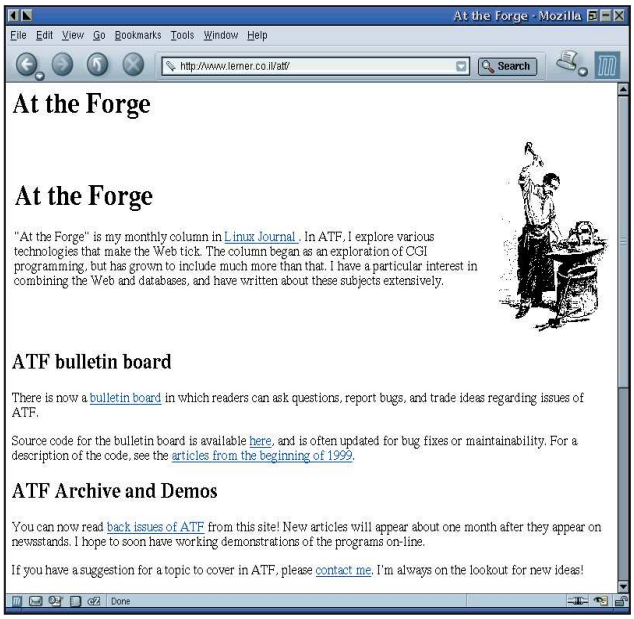
```
@five@
```

Ha a változónév a megfelelő lapon nem volt megadva vagy exportálva, az OpenACS futásidejű hibát jelez, és figyelmeztet, hogy ilyen változó nem létezik.
A feladatoknak ez a fajta megosztása azt jelenti, hogy a tervező és a programozó függetlenül tud dolgozni, mindaddig, amíg az előre megbeszélt, a lapmeghatározásban leírt csatolófelület (a Tcl-lap bemenete–kimenete) változatlan marad.

Tcl-lapok létrehozása

Az OpenACS számos olyan szolgáltatást tartalmaz, ami a programozókat segíti a HTML-űrlapok gyors és könnyű kialakításában. A jelenlegi bemutató céljaira ezeket a szolgáltatásokat most nem használjuk; ehelyett egyszerű nyers HTML-t fogunk alkalmazni. Alkalmazásunk két URL-t tartalmaz:

- Az egyik az ***atf_szia_postings*** tábla jelenlegi bejegyzéseit fogja időrendben megjeleníteni, illetve az új üzenetek űrlapja is itt jelenik meg. Az eredmények lekérdezéséhez és a `multirow` változóhoz történő csatolásához az OpenACS adatbázis-felületét fogjuk használni, ami könnyebben



használható, mint a beépített AOLserver adatbázis-felület (és nem kell aggódnunk a szálak és adatbázis-tárazás – pooling – miatt sem). Ezt a változót aztán az ADP-lap adatforrásaihoz adjuk, ahol az megjeleníthető. Az űrlap-művelet megadása (action) a másik URL-re fog mutatni.

- A második lap Tcl program lesz, amely fogadja a HTML-űrlapot, beviszi az új sort az adatbázisba, és visszavezeti a felhasználót az első lapra.

Más szavakkal alkalmazásunk létrehozásához két Tcl- és egy ADP-lapra lesz szükségünk. Valamennyit az atf-szia alatti www könyvtárba kell helyeznünk; amennyiben ez a könyvtár még nem létezik, hozzuk létre, majd kétszeresen bizonyosodjunk meg afelől, hogy a tulajdonosa azonos azzal a felhasználóval, amely alatt az AOLserver is fut.

Az *1. listában* található Tcl-lap (***posting.tcl***), nem vár értéket, és egyetlen ***postings*** nevű adatforrást ad tovább. A Tcl-lap az `ad_page_contract` függvényhívással kezdődik, amely lehetővé teszi, hogy lejegyezzük a fájl tulajdonosát és célját (az első érték), a megkapott bemenő értékeket (a második érték, ***posting.tcl***) és a továbbadott adatforrásokat (a harmadik érték, amelyet történeti okokból `properties`-nek neveztünk el). A Tcl-lap az `ad_return_template` hívással végződik, amely kikeresi a jelen ***.tcl***-fájllal azonos nevű ***.adp***-fájlt.

Bár minden adatforrás egyszerű Tcl-változó, az OpenACS sablonozórendszere néhány dologban leplezi a változók valódi természetét: minden adatforráshoz nevet és típust rendel (`multirow`, `list`, `onevalue` vagy `onerow`). Esetünkben a ***postings*** adatforrás `multirow` típusú, ami azt jelenti, hogy a `SELECT` lekérdezésünk eredményének több sorát is tartalmazhatja. A `db_multirow` függvény három értéket vár: a változó nevét, amelybe a sorokat be kell olvasni, a lekérdezés nevét, illetve magát az SQL-t.

A nevesített lekérdezés egyszerre áldás és átok az OpenACS-ben. Áldás, mert lehetővé teszi, hogy a lekérdezést egy külső ***.xql***-fájlba helyezve (ezek valamilyen adatbázis-kezelőhöz tartozó XML alakú fájlok) több adatbázissal dolgozzunk. A gond ezzel a megoldással az, hogy az OpenACS előbb a lekérdezés nevéhez rendelt ***.xql***-fájlt keresi ki, és csak akkor veszi figyelembe a ***.tcl***-lapunkban leírt SQL-t, ha az XML-t nem találja. Sok kezdő OpenACS-programozó tapasztalja meglepetten, hogy a rendszer figyelmen kívül hagyja a ***.tcl***-lapon végrehajtott SQL-módosításait, mivel valójában az ***.xql***-lapot nézi.

2. lista posting.adp

```
<master>

<!-- Az eddig kapott  zenetek megjelen tØse,
     ha van ilyen -->

<if @postings:rowcount@ ne 0>
<h1>Postings</h1>
  <multiple name="postings">
     <p><b>@postings.entry_date@,
        by @postings.user_name@
        ➥(@postings.email@)
     </b></p>

<blockquote>@postings.posting@</blockquote>
  </multiple>
  <hr>
</if>

<!-- œj  zenet hozzÆadÆs-->

<h1>Add a new posting:</h1>
  <form method="post" action="posting-add">
  <input type="text" name="posting_text">
  <input type="submit" value="Add some
              ➥text">
  </form>
</master>
```

Az ADP-lap létrehozása

Amint a ***posting.tcl*** kilép az `ad_return_template` meghívásával, az OpenACS sablonozórendszere kikeresi a ***posting.adp*** fájlt. Minthogy az összes Tcl-programkódot a Tcl-lapunk tartalmazta, ADP-lapunkban már nem lesz szükségünk a hagyományos `<% %>` tagokra. Ugyanakkor módot kellene találnunk arra, hogy adatforrásainkat HTML alatt jobban használhatóvá alakítsuk. Ahogy azt a *2. listában* (***posting.adp***) bemutatjuk, az OpenACS sablonozórendszere néhány új tagot vezet be, amelyek lehetővé teszik, hogy a ***.tcl***-lapon meghatározott értékeket egyszerűen elérjük:

- Feltételhez kötve az `<if>` cimke/tag segítségével a kimenő lapba beilleszthetünk valamilyen HTML-részletet, ami két értéket hasonlít össze (az `eq` és `ne`, azaz egyenlő és nem egyenlő relációk segítségével). A ***posting.adp*** példában a postings adatforrás sorainak számát (`@postings:rowcount@` lekérdezéssel) hasonlítjuk össze nullával. Ha nincs mit megjeleníteni, akkor egyáltalán semmit nem fogunk kirakni.
- Amennyiben vannak megjelenítendő lapok, végiglépkedünk rajtuk a `<multiple>` cimkével/taggal. A `<multiple>` teg blokkján belül az egyes adatbázismezőket a `@NAME.column@` szerkezetű kifejezéssel érhetjük el, amint azt a ***posting.adp*** forrásában megfigyelhetjük.

A megjelenített sorok számától függetlenül a ***posting.adp*** mindig beilleszt egy rövid HTML-űrlapot, ami a tartalmát a `posting-add` programnak adja át. Ez a ***posting-add.tcl*** nevű

3. lista posting-add.tcl

```
ad_page_contract {

  Add a new posting

  @author Reuven M. Lerner
  ➥(<reuven@lerner.co.il>)
  @date 2002-September-18
} {
  posting_text:trim
} -properties {
}

# pillanatnyi felhasznÆl  ID megszerzØse

set user_id [ad_get_user_id]

#  SQL-lekØrdezØs lØtrehozÆsa œj bejegyzØs
#  beszœrÆsÆhoz
set sql "INSERT INTO atf_szia_postings
            (user_id, posting)
          VALUES (:user_id, :posting_text)"

#  zenet beszœrÆsa

db_dml insert_posting $sql

ns_returnredirect posting
```

kód (amelyet a *3. listában* láthatunk) az `ad_page_contract` hívással kezdődik, amely egyetlen bemenő értéket sorol fel (`posting_text`). A bemenő értékek alapértelmezés szerint kötelezőek, de megjelölhetjük őket elhagyható értékként, illetve az `ad_page_contract` bejegyzésének módosításával alapértelmezett értéket rendelhetünk hozzájuk. Jelen esetben megkérjük az OpenACS-t, hogy a megkapott szövegből távolítsa el a kezdeti és a szövegvégi szóközöket.
Ezután lekérdezzük a pillanatnyi felhasználói azonosítót (ID) a beépített `ad_get_user_id` függvénnyel, és ezt az értéket rendeljük a `user_id` változóhoz. A `db_dml-t` használjuk fel az üzenet adatbázisba történő illesztéséhez. Figyeljük meg, hogy a `db_dml` változóneveihez kettőspontokat használtunk (dollárjelek helyett) – ez ugyanis az OpenACS adatbázis-felület szabványa, ami biztosítja, hogy nem lesznek idézőjelezési nehézségeink, ha az adatbázis-kiszolgálónak adatot adunk át.
Végül a ***posting-add.tcl*** a felhasználót átirányítja a `posting`-ra, ami a ***posting.tcl***-t futtatja le, megjelenítve a ***posting.adp***-t.

Az utolsó simítások

Most már visszatérhetünk az APM-be, és létrehozhatjuk a sablonjainkat, valamint az adatbázis-készítő parancsfájljainkat tartalmazó csomagot. Kattintsunk az atf-szia csomag nevére, majd a ***manage file information*** (fájladatok kezelése) hivatkozásra a lap aljának környékén. Ezután ebben a csomagban válasszuk az ***scan for additional files*** (további állományok keresése) lehetőséget – itt láthatjuk az általunk telepített ***.sql-***, ***.tcl-*** és ***.adp***-lapok listáját. Jelezzük, hogy az összes itt található fájl a csomag része lesz, majd térjünk vissza az ATF Szia APM fő kezelőképernyőjére, kattintsunk a ***generate a new atf-szia.info file (új atf-szia.info*** fájl készítése) hivatkozásra.
Készen állunk egy olyan APM létrehozására, amelyet azután már bármely OpenACS-felhasználó használni tud. Kattintsunk a ***generate a file*** gombra, és a terjesztés fájladatai alatt megtekinthetjük, hogy mekkora lett az új APM. Ha erre a hivatkozásra kattintunk, az APM letöltődik a rendszerünkre.
Hogyan telepítsünk fel egy új APM-et, amit valakitől kaptunk? A legegyszerűbb módszer, ha az APM-et a kiszolgáló fájlrendszerébe helyezzük. Ezt követően a böngészőnkkel térjünk vissza a fő APM-lapra (***/acs-admin/apm/***) és kattintsunk az ***install link***-re (a hivatkozás telepítése). Mondjuk meg a rendszernek, hol találja az APM-et, és máris bekerül a ***packages*** alkönyvtárba. Ettől kezdve a múlt hónapban megismert APM-telepítő segítségével folytathatjuk a munkát. Az adatmodell az adatbázisba kerül, és a weblapok elérhetővé válnak az érdekelt csoportok számára. Természetesen ha egy csomag felkerült a rendszerre, az ACS-honlaptérkép alkalmazásával tetszőleges URL-ek alatt új példányokat hozhatunk belőle létre.

Mit hagytunk ki?

Ez a kis példa az OpenACS alkalmazásfejlesztés jéghegyének csak a csúcsát tudta bemutatni. A lehetőségek sokrétűek:

- A sablonozórendszer önműködő űrlapkészítő rendszerrel is fel van szerelve, amellyel könnyedén készíthetünk érvényesítő visszakérdezésekkel és adathelyesség-ellenőrzéssel ellátott HTML-lapokat.
- A Tcl-függvényeket indításkor is betölthetjük az AOLserverbe, ha a csomag ***tcl*** könyvtárában határozzuk meg őket.
- A nevesített SQL-lekérdezések – mint korábban említettük – lehetővé teszik, hogy a Tcl program az Oracle és PostgreSQL rendszerek bármelyikét átlátszóan érje el .
- Minden egyes csomagpéldányt a társaitól teljesen függetlenül kezelhetünk – az OpenACS rendszer logikájának köszönhetően.
- Minden példány saját változókkal rendelkezhet, ami telepítésfüggő beállításokat tesz lehetővé.
- Minden csomag saját jogosultságkészletet határozhat meg (vagy használhat), így lehetővé válik, hogy rendszerünkön az egyes felhasználókhoz és csoportokhoz egyedi jogosultságokat, illetve elérési listákat alakítsunk ki.

Összegzés

Az OpenACS összetett rendszer, és az APM sem éppen a legkönnyebben megtanulható fejlesztőeszköz, hiszen olyan sok bonyolult esetet próbál megoldani, amelyekkel a web- és adatbázis-fejlesztők gyakorta találkoznak. Ugyanakkor még soha nem láttam egyszerűbb módszert a web- és adatbázis-alkalmazások terjesztésére ilyen fokú modularitás, adatbázisok közötti hordozhatóság és sablonok adta rugalmasság mellett. Az efféle alkalmazások létrehozásának egyszerűsége az adatmodellek gazdagságával és a hatalmas mennyiségű elérhető alkalmazással kiegészítve az OpenACS-t egyértelműen életképes és hasznos környezetté teszik a hálózati közösségek világában.

Linux Journal 2002. december, 104. szám

Reuven M. Lerner (➲ http:/www.lerner.co.il/atf) Nyílt forrású programokra, valamint web- és adatbázis-alkalmazásokra szakosodott tanácsadó. Könyve, a Core Perl, 2002 januárjában jelent meg a Prentice Hall gondozásában. Reuven feleségével és lányával Izraelben, Modi'in-ben él.

Egy Smörgåsbord-folyamat

Marcel folyamataink szemmel tartásának néhány valóban tudattágító módját szolgálja fel.

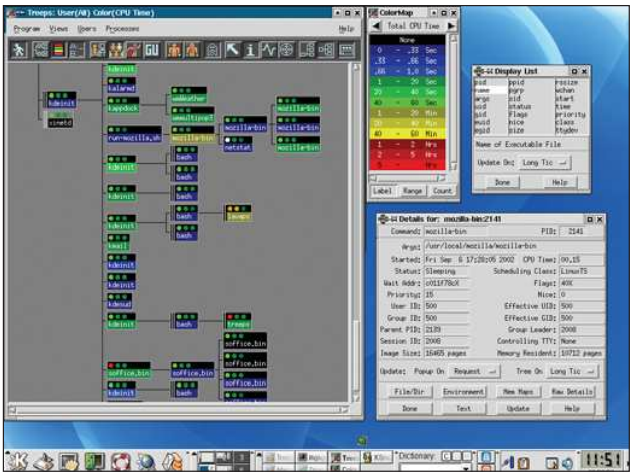
François, vite, a vendégeink bármelyik pillanatban itt lehetnek! Quoi? Azt mondod, már mindent előkészítettél? Kitűnő, François! Látom, egészséges mennyiségű utánpótlást hoztál fel az 1998-as Barossa Valley Shirazból. Ez remekül illik majd a mai menühöz, nem gondolod? Qu'avez-vous dit? Ja igen, a mai téma… a rendszerfelügyelet, mon ami, és úgy döntöttem, ma este a folyamatok kezelésébe fogunk egy kicsit betekinteni.

Á, megérkeztek a vendégek! Isten hozott, mes amis, Chez Marcelnél, a kiváló Linux-konyha, a páratlan hangulat és a kitűnő borok házában. Az asztalotok és a borotok csak rátok vár. Foglaljatok helyet, hűséges pincéretek mindjárt teletölti poharaitokat! El kell mondanom, hogy François ma szokatlanul tevékeny. Igazán nem tudom, mi ütött belé. Egy újabb dolog, ami válaszra vár, nem igaz?

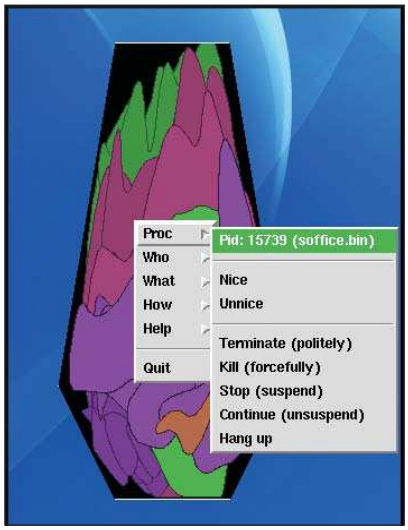
Ahogy azt az éttermet rendszeresen látogatók már tudják, minden, ami linuxos gépünkön fut – mindegyik héj, minden élő internetkapcsolat, minden játék –, egy-egy folyamat. Néha a programok is létrehozzák a saját folyamataikat, ezek a gyermekfolyamatok. Szakmai szempontból nézve egy kivétellel minden olyan folyamat gyermekfolyamatnak tekinthető, amelyik egyetlen közös őstől származik. Ez a fő folyamat az `init`. A gyermekfolyamatok további folyamatokat hozhatnak létre, azok pedig még továbbiakat. Listázásukra a `ps` parancsot használhatjuk, de a viszonyok követése igen fárasztóvá válhat. Mon Dieu, mes amis, azt hiszem, ezen a ponton innom kell egy kortyot. Ahhoz, hogy gyors pillantást vethessünk rá, melyik folyamat melyiknek a leszármazottja, a `pstree` parancsot használhatjuk. Figyeljük meg az alább látható kimenet első sorait és az `init` helyzetét:

```
init-+-apmd
     |-atd
     |-bdflush
     |-cardmgr
     |-crond
     |-gpm
     |-kalarmd
     |-kapmd
     |-kappdock-+-wmWeather
     |          `-wmmultipop3
     |-kdeinit-+-artsd
     |         |-autorun
     |         |-kdeinit
     |         |-kdeinit---2*[bash]
     |         |-kdeinit---bash
     |         |-kdeinit-+-bash---lavaps
```

Bár elég rendezettnek tűnik, mintha valahogy mégis adatszegény lenne. Régi barátunk, a `ps` parancs használatával ugyanezt a hatást érhetjük el, csak egy kicsivel több adattal. Az `f` kapcsoló hatására úgynevezett erdőt kapunk, melyben ha megengedtek egy bugyuta viccet, mes amis – láthatjátok a folyamatok fáit. Amikor ***George MacDonald*** megalkotta Treeps nevű programját, az járhatott a fejében, hogy ezt a tengernyi folyamatot egy olyan alakban legyen képes megjeleníteni, amiből minden lényeges adat rögtön kiolvasható. A program egy felhasználói beavatkozást igénylő grafikus folyamatkövető eszköz, állandóan frissülő, színes kijelzővel, amelynek segítségével az egyedi programok könnyen kiszűrhetővé válnak. Mindenképpen megéri a letöltést. A forráskódhoz a ➲ http://www.orbit2orbit.com/gmd/tps/treepsfm.html címen férhetünk hozzá.



1. kép A Nirvana-folyamat a Treeps jóvoltából



2. kép Szemléletettágító folyamatkezelés Lava-Lamp módra

A Treeps fordításához először a forráskódot kell kicsomagolnunk (a `tar -xzvf treeps-1.2.1.tar.gz` paranccsal), majd a telepítőkönyvtárból futtatnunk a `./Setup` parancsfájlt. Miután az előzetes fordítás beállításai megtörténtek, a `make install` parancs. Ezután a `treeps &` begépelésével már indítható is a Treeps. A kezdőnézet a saját folyamatainkat mutatja, olyan módon, ahogyan az `init` elindította őket – és az igazi élvezet most kezdődik! Az mutatót valamelyik folyamat fölé mozgatva megjelennek a folyamat legfontosabb jellemzői, éppen úgy, miként a `ps x`

parancs mutatná. Jobb egérkattintásra egy helyi menüből kiválaszthatjuk a folyamat fontosságának megváltoztatását (renicing), a hozzá tartozó súgóoldalak megjelenítését és így tovább. A felső gombsorból a nézet beállítására nyílik lehetőségünk. Saját folyamataink mellett választhatjuk a démonfolyamatok megjelenítését vagy egyszerűen az összes futó folyamatét. Ha az információs gombra kattintunk (az ***I*** betűvel ellátott gomb), mutató is nagy ***i*** betűvé változik. Ekkor bármelyik folyamatra rákattintva annyi részletező adatot kapunk, amennyiről nem

3. kép Vérbosszú egy folyamaton: psDooM

is gondolnánk, hogy létezik. Ebből az ablakból még mélyebbre is áshatunk az adatok között. A ***File/Dir*** gombra kattintva az összes olyan fájlt megjeleníthetjük, amit az adott folyamat nyitott meg. A nagyon kíváncsiak számára a ***Mem maps*** gomb még azt is megmutatja, hogy az egyes programrészletek hol helyezkednek el a memóriában.
A program enyhén szólva is sok lehetőséggel bír, ám ami engem a leginkább megragadott, az a színkódolás. A program futása közben a ***color-bar*** gombra kattintva kapcsoljuk be a ***color map viewer***-t (színtérképet). A különböző lehetőségeken végigpásztázva a folyamatok felhasználóazonosító (user ID), csoportazonosító (group ID), a felhasznált összes processzoridő, a pillanatnyi processzorterhelés, folyamatállapot (alvó, futó, zombi stb.), a folyamathoz rendelt memória mennyisége, képméret és még sok egyéb szempont szerint kiemelhetők.
A ***Program*** menü alatt további élvezeteket találunk ***System Info*** néven – ez a ***System Information App Launcher***-t indítja el. Ebből a gombokkal bőven megrakott ablakból a rendszerünkkel kapcsolatos adatok tömkelegére nyílik rálátásunk: a betöltött modulok útvonaltáblájától kezdve a rendszermagszinten, a PCI-eszközökön, a működési időn és a lemezfelosztások adatain át sok más egyéb társaságában egészen a futási szintekig. Különböző nézőpontokból szemlélve még az ismerős dolgok is új értelmet nyerhetnek. Ez a kis felfedezőút valóban tágítja a látókörünket, nemde? Amennyire furcsának tűnik mai menünk következő fogása, olyannyira sok élvezetem leltem benne a vele való munka és vizsgálódás közben. Ezt a programot attól függően találhatjuk hasznos folyamatmegfigyelőnek, hogy milyen érzésekkel viseltetünk a lávakitörések iránt. ***John Heidemann*** programjának alapötlete a higgadt gépkezelés (calm computing) gondolatából származik, amelynek „The Coming Age of Calm Technology" (A higgadt technológia eljövendő korszaka) címen ***Mark Weiser*** és ***John Seely Brown*** volt az elindítója.
Az alapgondolat az, hogy a folyamatokat egy lávalámpa folyadékcseppjeihez hasonlóan ábrázolják. Minél nagyobb a csepp, annál nagyobb a memóriafelhasználása. Minél gyorsabban mozog, annál jobban igénybe veszi a processzort. Mint minden rendes folyamatkövető rendszer, ez is lehetővé teszi a folyamatok azonosítását, fontonosságuk megváltoztatását és kilövését. Az ismerkedést kezdjük a csomag egy példányának a letöltésével a Lavaps honlapján (➲ http://www.isi.edu/~johnh/SOFTWARE/LAVAPS/index.html). A Red Hat felhasználói előrecsomagolt RPM-fájlokat is találnak, de a többiek se essenek kétségbe – a LavaPS egyszerűen a jól ismert (mondhatnám: „klasszikus") ötlépéses kicsomagoló–fordító eljárás újabb példájának felel meg:

```
tar -xzvf lavaps-1.20.tar.gz
cd lavaps-1.20
./configure
make
su -c make install
```

A lava lamp elindításához a `lavaps &` parancsot kell kiadnunk. Ekkor egy kis lávalámpa jelenik meg az asztalunkon; jobb egérkattintásra egy helyi menü kínálja a lehetőségek széles választékát. A ***proc*** menüpontból a folyamatazonosítót és a folyamat nevét tudjuk meg – ugyancsak innen küldhetünk különböző jeleket (többek között `kill` jelt is) a folyamatnak a kényszerített leállítástól az ideiglenes felfüggesztésig.
A LavaPS-nak a folyamatok szemmel tartására és felügyeletére történő használata teljesen új szemléletmódot tár fel számunkra. Az egyetlen dolog, ami nem tetszett, az az, hogy az alapértelmezett láva 1024×768-as felbontású megjelenítőmön igen kis méretűnek hatott. Ennek megoldása az X erőforrásainak beállítását igényli, amit egyszerűen a ***$HOME/.lavapsrc*** beállítófájl módosításával érhetünk el. Az enyémben egyedül a méret megváltoztatására volt szükség, ***.lavapsrc*** fájlomban a beállítás így fest:

```
lavaps.geometry: 204x404+700+0
```

Ha már a teljes szemléletmódváltásnál tartunk, mes amis, a folyamataink titkait fürkésző legfurcsább kirándulásra a Doom – az ID Software cég már-már avitt lövöldözős játéka – kalauzol el bennünket. 1997-ben az ID Software felszabadította a Doom forráskódját, amit számos változat követett. Ezek egyike az Xdoom, egy Unix X Window-változat, amelyen ***David Koppenhofer*** psDooM nevű programja alapul. Miként a psDooM-ot az Xdoom ihlette, Davidet ***Dennis Chao***, Dennis pedig ugyanígy építhetett ***Vernor Vinge*** munkájára. Ha érdekelnek a részletek, látogass el Dennis „Doom as a tool for system administration"

KAPCSOLÓDÓ CÍMEK

Treeps
➲ http://www.orbit2orbit.com/gmd/tps/treepsfm.html
LavaPS
➲ http://www.isi.edu/~johnh/SOFTWARE/LAVAPS/index.html
Doom as a tool for system administration
➲ http://www.cs.unm.edu/~dlchao/flake/doom
DoomWorld
➲ http://www.doomworld.com
A psDooM honlapja ➲ http://psdoom.sourceforge.net
Marcel borlapja
➲ http://www.marcelgagne.com/wine.html

(A Doom mint a rendszerfelügyelet egyik eszköze) című oldalára (lásd a ***Kapcsolódó címek*** között).
Akárhogy is, a psDooM megvalósítása mögött a folyamatkezelés egy szokatlan lehetőségének gondolata húzódik meg. A termekben kószáló szörnyek feje felett vörös folyamatazonosítók lebegnek a parancs nevének utolsó hét karaktere társaságában. A forrás csomagok a psDooM honlapján megtalálhatók, de a telepítés legkönnyebb módja az előre lefordított bináris állományok használata. A telepítés nagyon egyszerű, csupán futtassátok az `install.sh` parancsállományt:

```
tar -xzvf psdoom-2000.05.03-bin.tar.gz
cd psdoom-bin
su -c "./install.sh"
```

A psDooM futtásához szükség lesz egy IWAD állományra, mégpedig a Doom 1, Doom 2 vagy az Ultimate Doom fájljára. A Doom 1 feltételekhez kötötten ingyenes (shareware) változatának IWAD-ja is megfelelő. Ha esetleg nem rendelkezünk vele, a ➲ http://www.doomworld.com oldalról tölthetjük le, én is innen töltöttem le egy példányt:

```
unzip shareware_doom_iwad.zip
su -c cp DOOM1.WAD
➥/usr/local/games/psdoom/doom1.wad
```

Ennyi az egész. Most már készen állunk a psDooM futtatására:

```
cd /usr/local/games/psdoom
./psdoom -2
```

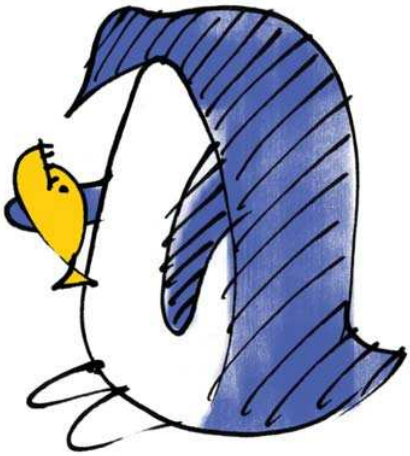
Figyeljünk a fenti `-2` kapcsolóra! Alapértelmezésben a képernyő igen kis méretű – ez a beállítás növeli a képméret alapértelmezett értékét. Ha korábban nem játszottatok a Doommal, figyelmeztetnem kell benneteket, hogy egy kicsit erőszakossá tehet. Egy folyamat szörnyének megsebesítése a folyamat fontosságának megváltoztatásával egyenértékű (renice +5). Tartsátok tűz alatt, így megsemmisíthetitek a folyamatot, de természetesen csak a rendszerjogosultságok tiszteletben tartásával! Ugyan egy másik felhasználóhoz tartozó folyamatszörnyet is megölhetünk, de az fel fog támadni, csakis a sajátjaink pusztulnak el végleg. Fontos figyelmeztetés: érdemes egyszerű felhasználóként kipróbálni, és természetesen nem a cégünk kiszolgálógépén.
Úgy tűnik, az idő már megint a záróra felé közeledik. Remélem, élveztétek a folyamatfelügyelet területén tett felfedező utunkat. El kell ismernem, hogy a psDooM-élmények még mindig túl elevenek bennem. Talán egy kis bor lecsillapítja az idegeimet. François, ha volnál szíves újratölteni vendégeink borospoharait, és természetesen az enyémet is…!
Viszontlátásra a következő hónapban. A votre santé!
Bon appétit!

Linux Journal 2002. december, 104. szám

Marcel Gagné (mggagne@salmar.com)
Mississaguában, Ontario államban él. Ő a szerzője a Kiskapu kiadásában szeptemberben megjelent Linux-rendszerfelügyelet (ISBN 96-9301-40) című könyvnek (jelenleg is egy könyvön dolgozik).

A rendszergazdákról

Lehet, hogy a rendszergazdák új nemzedéke már nem is érti pontosan a dolgok működését?



A rendszerfelügyelet kifejezés a különböző emberek számára eltérő jelentésekkel bír. Van, aki úgy gondolja, hogy bárki, aki felügyeleti teendőket lát el egy gépen, az rendszergazda. Olyat is hallottam már, hogy aki képes grafikus felületen a rendszerhez egy felhasználót hozzáadni, az is rendszergazda. Számomra a rendszergazda olyan személy, aki érti, mi történik a színfalak mögött. Nemrég felvételi beszélgetésre hívtam néhány embert a cégemhez. Hálózati rendszergazdáknak vallották magukat. A legtöbbnek ötlete sem volt arról, hogy a hálózati maszk mire való. Tudták, mi az az ARP, de az ARP és az IP kapcsolatáról már fogalmuk sem volt. Ismerték az OSI modellt, de azt már nem tudták, hogy az ARP, az IP, a TCP vagy az UDP hol foglal benne helyet. A dolgok működésének megértése megkönnyíti a hibaelhárítást. Ettől lesz valaki igazi rendszer- vagy hálózatgazda. Az alábbiakban ehhez próbálok némi segítséget nyújtani, és néhány szórakoztató programra is fel szeretném hívni a figyelmet.

Katoob

Számos formátum kezelésére alkalmas szövegszerkesztő. Az angol és az arab nyelvet támogatja, menüsorának ikonjaival például jobbra és balra lehet igazítani a szöveget, de egyéb műveleteket lehet végezni. Ugyan könyvtár- és memóriahasználati szempontból nagy az étvágya, de aki X-felületen dolgozik, azt ez nem nagyon fogja zavarni. Futtatásához szükséges: libgtk-x11-2.0, libgdk-x11-2.0, libatk, libgdk_pixbuf-2.0, libm, libpangoxft, libpangox, libpango, libgobject-2.0, libgmodule-2.0, libdl, libglib-2.0, glibc, libX11, libXi, libXft, libXrender, libXext és libfreetype.
➲ http://www.arabeyes.org/project.php?proj=Katoob

Euchre

Ha szereted az euchre nevű kártyajátékot, a lenti címen egy nagyszerű változatát érheted el. A gépi játékosok három tudásszinten mérkőzhetnek meg, a szerző pedig a játékban járatlanok számára is segítséget nyújt. A játszmák pörgősek és könnyűek. Futtatásához szükséges: libgtk, libgdk, libgmodule, libglib, libdl, libXi, libXext, libX11, libstdc++, libm és glibc.
➲ http://sourceforge.net/projects/euchre

pgmaint

Ha valaki Postgres segítségével naplózza a Snort adatait (különösen nagy terhelésű kiszolgáló esetében), bizonyára tisztában van az adatbázis rendszeres kitakarításának, illetve tartalma elemzésének a fontosságával. Ha sok ilyen adatbázissal rendelkezel, pucolgatásuk hamar unalmassá válik. A `pgmaint` képes ezek kezelésére, akár `cron` segítségével is. Futtatásához szükséges: Perl, DBI, Config::Simple és Getopt::Mixed Perl-modulok.
➲ http://sourceforge.net/projects/pgmaint

Childsplay

Lehetőség szerint több gyermekeknek szánt játékot szeretnék szemlézni, ugyanis gyermekeim mindig újabb és újabb játékokat keresnek. Feleségem szerint egy hétéves kislánynak még nem való a Quake, a Barbie.com játékai viszont hosszú órák után sem válnak unalmasakká. A gcompris sem rossz, a Childsplay a gcompris által igényelt Gnome könyvtárak garmadája nélkül is remekül szórakoztatja a kicsiket. Futtatásához Python és pygames szükséges.
➲ http://childsplaysourceforge.net

DNS Sleuth

Perl parancsfájl, ami parancssorból vagy a mellékelt CGI parancsfájl segítségével webkiszolgálón keresztül is futtatható. Ellenőrzi, hogy a megadott tartománynév megfelel-e az RFC-k előírásainak, majd a vonatkozó RFC bekezdésekre mutató hivatkozásokkal együtt jelzi a hibákat, melyeket így pontosan elemezhetsz és megfelelően javíthatsz. Futtatásához szükséges: Perl, Net::DNS Perl-modul, igény szerint CGI-támogatással rendelkező webkiszolgáló.
➲ http://atrey.karlin.mff.cuni.cz/~mj/sleuth

reportdhcp.pl

Csipetnyi Perl, némi tájékozódás a beállító- és naplófájlok helyéről, kevéske utómunkálat, és máris hasznos eszköz áll a rendelkezésedre, ami a webböngészőn keresztül teszi lehetővé a DHCP-bérletek követését. Az útmutatások érthetők, egyszerűek. Két perc alatt előkeresheted az IP- és MAC-címeket, az ügyfelek nevét, kimutatásokat érhetsz el a kiszolgálóról és a többi. Futtatásához Perl, webkiszolgáló és -böngésző szükséges.
➲ http://www.omar.org/opensource/reportdhcp

Crossword Generator (csak letöltés)

Ha szereted a keresztrejtvényeket, ez a program minden vágyadat kielégíti. Neked csak a táblát és a szavak listáját kell megadnod, a többit a program elvégzi. Jó, ha kéznél van egy nagyszótár is, így a meghatározások mezőibe rokon értelmű kifejezések vagy rövid magyarázatok kerülnek, és nem maga a megfejtés. A leírás – és a szótárak, valamint az egyéb kiegészítők – egyelőre csak spanyolul érhetők el. Futtatásához szükséges: libstdc++, libm, glibc, TeX és LaTeX.
➲ http://www.ldc.usb.ve/~96-28234/crossword-0.8.tar.gz

Ennyit erre a hónapra!

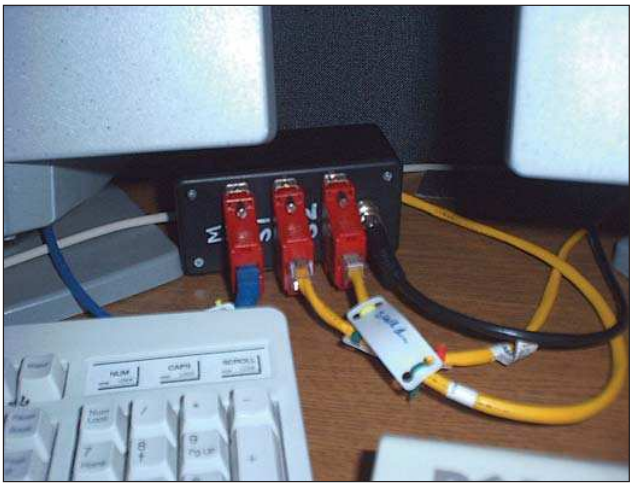
Linux Journal 2002. december, 104. szám

David A. Bandel (dbandel@pananix.com)
Jelenleg Panamában él, Linux- és Unix-tanácsadással foglalkozik. Társszerzője a Que Special Edition: Using Caldera OpenLinux című könyvnek.

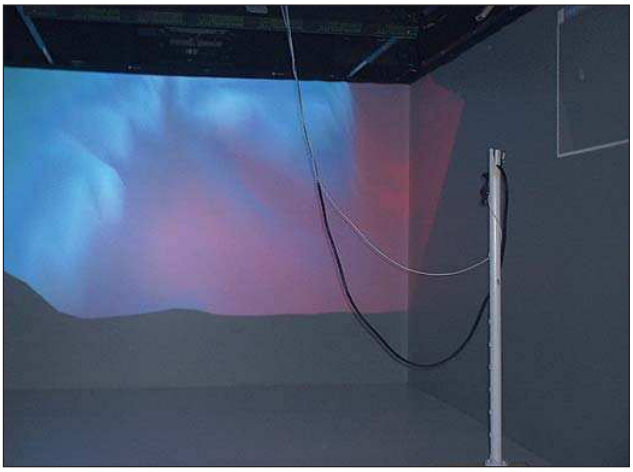
Négyfalas 3D-s megjelenítő Linuxra

Szuperszámítógépes térmozi helyettesítése négyfalas immerzív megjelenítőrendszert meghajtó egyszerű PC-teleppel.

A PC-telepeket ár–teljesítmény arányuk a hagyományos felsőkategóriás képalkotó szuperszámítógépek figyelemre méltó ellenfelévé emeli. A közönséges PC-alkatrészek gyors fejlődése lenyomta az árakat, ugyanakkor felgyorsította elavulási ciklusukat. Ha az SGI-tól vásárolunk grafikus teljesítményt, általában a következő szabályt követhetjük: grafikus vezetékenként (pipe) 250 ezer dollár költséggel számolhatunk. Ezzel szemben kísérleti telepünk csomópontonként mindössze ezer dollárba kerül. A videomátrix kapcsolóval együtt ez összességében is kevesebb, mint 15 ezer dollár.
Most egy olyan kísérletet mutatunk be, amelyben a próbatelepünket egy már létező négyfalas megjelenítőrendszerrel, a Mechdyne Vállalat Surround-Screen Visualization System (SSVR) rendszerével kötöttük egybe. Célunk aktív sztereomegjelenítés elérése több falon, külön figyelve a képvezérlésre (genlocking), a lapozóvezérlésre (swap-locking) és az adatzárolásra (data-locking). A felsőkategóriás képalkotó szuperszámítógépek támogatják az egyesített többfalas és aktív sztereomegjelenítést. A kép grafikus adatainak sztereomegjelenítését és vezérlését a számítógép önműködően kezeli valamilyen egyedi alkatrésszel vagy egy üzleti programkönyvtár hívásain keresztül. A telepet eredetileg a négyfalas rendszerünk meghajtására használt, öregedő SGI számítási felszerelésünk lecserélésére akartuk felhasználni. Most mégis azon kapjuk magunkat, hogy az Infinite Reality 2-es grafikájú Onyx 2-es rendszer képességeit számolgatjuk, és egyre nagyobb számú poligont próbálunk árnyalással megjeleníteni az aktív sztereóhoz szükséges képfrissítés (framerate) mellett. Amikor telepet készítünk, ezekkel a kérdésekkel sorban foglalkoznunk kell, ha egy időben több képernyőn egységes képet szeretnénk kapni.
A telepcsomópontok közti kapcsolattartás létfontosságú. A csomópontok között olyan adatok cserélődnek, mint pixelek, geometriai primitívek vagy akár a kép grafikus adatai. Az adatkezelési módszer és az elküldött adat típusa nagymértékben befolyásolja a telep hálózati sávszélességigényeit. A grafikus telep kapcsolattartó programszerkezeteinek tervezésénél két alapvető megközelítés létezik: az ügyfél-kiszolgáló és a mester-szolga viszony.
A kiszolgálóalapú megközelítésnél egyetlen csomópont szolgáltatja az adatokat a grafikus leképezést (rendering) végző ügyfeleknek. Az ilyen felállás előnye, hogy az árnyalást végző ügyfélgépekkel dolgozó kiszolgálón több alkalmazást is elhelyezhetünk. Ez a környezet nagyon rugalmas. Hátránya a hálózati sávszélesség nagyobb igénybevétele. A legtöbb kiszolgálóalapú rendszer a viszonylag költséges Myrinet vagy gigabit ethernet alkatrészekre épül.
A mester-szolga megközelítés, amelyet ebben a projektben is használunk, több csomópontot tartalmaz. Ahol a grafikus telep minden csomópontja helyileg tárolja és futtatja a grafikus alkalmazás egy-egy példányát. Következésképpen az adatnak csak kis részét kell a csomópontok között megosztanunk, így a hálózati sávszélesség már sokkal kevésbé számít. Az információ itt mindössze a bemeneti eszköz adatait és időpontokat



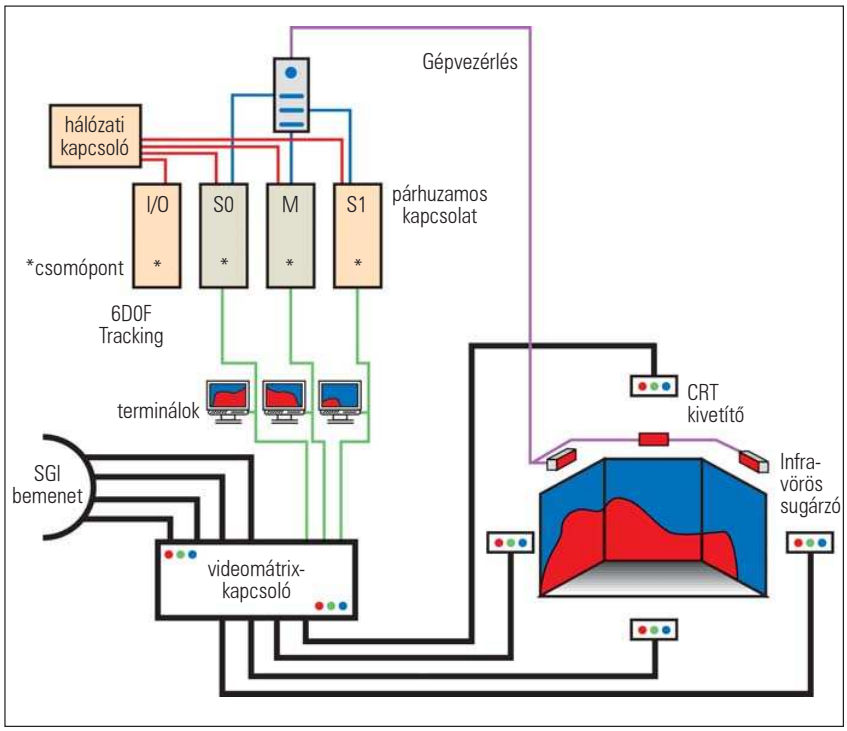
1. kép A házi készítésű vezérlődobozkánk



2. kép A próbaalkalmazás futtatása

jelent. Ennél a felállásnál az alkalmazás állapotváltozásait a mester kezeli.
Minden grafikus csomópontnak ki kell elégítenie a következő feltételeket:

- ***Képvezérlés (Genlocking):*** a képkockákat- (video frame) összehangoló folyamat a telep minden egyes csomópontján, hogy folyamatos és egységes képet állítsanak elő. A Genlocking program segítségével vagy egyedi eszközzel is elérhető.
- ***Lapozóvezérlés (Swap Locking):*** az árnyaláshoz és a csereterülethez használt képverem összehangolását végző folyamat. Erre azért van szükség, mert a kép minden egyes nézőpontja különböző mennyiségű adatot és árnyalandó



A rendszer felépítése

poligont tartalmaz. Ennek következtében minden csomóponton minden képkockához különböző árnyalási idő tartozhat.

- ***Adatvezérlés (Data Locking):*** a képernyők közötti egység fenntartása érdekében szükséges nézőpont-összehangolást végző folyamat. Ez a kérdés azért itt merül fel, mert a képkockákat minden csomópont helyileg tárolt adatból állítja elő.

Szabványos összeállítású PC-ket használtunk MSI G4Ti4600 grafikus kártyával, amelyet az nVidia GeForce4 Ti grafikus feldolgozóegysége hajtott meg, és 128 MB DDR videomemóriával rendelkezett. Bár nem feltétlenül lett volna szükséges, a PC-k egyformák voltak, ami leegyszerűsítette a programtelepítést.
A PC-k 100BaseT hálózati átalakítóval tartották a kapcsolatot 100BaseT hálózati kapcsolón (switch) keresztül.
Az SSVR vetítőit az Extron CrossPoint Plus 124 mátrix videováltóhoz (switcher) kapcsoltuk. A váltó 12 forrásból képes videobemenetet fogadni, a kimenetet pedig négy forrásra helyezheti ki.
Mivel a képvezérlést és az adatzárolást a párhuzamos kapun keresztül programmal végeztük, a jelvezérlés pontos végrehajtására egy kis eszközt készítettünk (lásd az *1. képen*). Ezt az eszközt szintén a boltban kapható termékekből raktuk össze, kevesebb mint húsz dollárért. A váltó vezérlése mellett ez a doboz adta ki a Crystal Eyes infravörös sugárzók képvezérlés jeleit is.

A rendszer megvalósítása

A telep aktív sztereo- és képösszehangolási igényeinek kielégítésére többféle programot is használnak. A telepet szabványos Red Hat Linux 7.2 telepítésére alapoztuk, majd a rendszermagot foltoztuk, hogy támogassa a valós idejű alkalmazásfelületeket (Real-Time Application Interface – RTAI), ami gyors válaszidőt és feladatbefejezés-időzítést tesz lehetővé.

A SoftGenLock és az RTAI rendszert együttesen használjuk fel az aktívsztereó megoldást nyújtó programban. Az RTAI rendszermagmodul érzékeli a monitor függőleges frissítését, és a videokártya memóriájában megváltoztatja azt a mutatót, ami a képernyőre rajzolandó területre mutat. Az alkalmazásnak kétszeres méretű tárterületre van szüksége, amit Xfree86 alatt a képernyőre ténylegesen kirajzolt képterületnél kétszer nagyobb virtuális képernyőterület megadásával ér el. Ha például 1024×768-as felbontásban szeretnénk aktív sztereóhatást, virtuális asztalunk mérete 2048×768 lenne. Az RTAI rendszermag modul kettéosztja a képtárterületet (framebuffer), és a képet 1024×768 darabokban jeleníti meg. Az alkalmazás sztereóban rajzol az X képterület jobb és bal oldalára, külön-külön a két szem számára.
A telep gépeit párhuzamos kapukon keresztül hangoltuk össze, így valósítottuk meg a Genlock-adatzárat. Az RTAI rendszermagmodul a párhuzamos kapu egyik lábára írva és egy másikról olvasva győződhet meg arról, hogy a telep többi gépe befejezte-e már a képkocka rajzolását.
A mester utasítja a többi csomópontot, hogy mikor kell rajzolni, a csomópontok pedig visszajeleznek, amint készen állnak egy új képkocka rajzolására. Az adatzárolást úgy oldottuk meg, hogy elértük, a telep szolgagépei mindig a helyes szemen dolgozzanak, amikor a párhuzamos kapu egy bizonyos állapotban van.
A SoftGenLock a telep egyes gépei között nem hangolja össze az alkalmazásokat – csak az adatzárolással és a sztereóhatással foglalkozik. Az alkalmazásban használt nézetcsonk (viewing fustrum) és az animációk összehangolása már az alkalmazás feladata. Végül, minthogy a SoftGenLock kizárólag VGA-regisztereket használ, minden valószínűség szerint valamennyi videokártya alatt működik.
Amikor PC-telepet telepítünk, az összeállítás előtt számos dolgot figyelembe kell vennünk. Következzen hát néhány fontos szempont és pár javaslat a rendszer helyes felállítására:

- Bizonyosodjunk meg róla, hogy megfelelő légtechnika, táp és hálózati elérés áll rendelkezésünkre ott, ahol a telepet fel szeretnénk állítani.
- A telepet és a megjelenítőrendszert külön helységekben állítsuk fel, hiszen a ventilátorokból, meghajtókból és egyebekből áradó zaj kiábrándító lehet.
- Készüljünk fel számos, a kábelfektetéssel kapcsolatos nehézségre. Ha ezen a ponton egy kis időt szánunk a címkézésre és a megfelelő kábelméretek és hosszak ellenőrzésére, sok időt takaríthatunk meg a későbbiekben. Így enyhíthetők a jelgyengülésből eredő gondok, illetve elkerülhető a kábel eltűnése a „spagettiban”.
- Készítsünk egy főkapcsolót, amivel az összes egységet egyszerre ki lehet kapcsolni.

Mi háromfalas telepünket kevesebb mint két hét alatt készítettük el. A telepet aktív sztereó használatára állítottuk be, de a csereterület-zárolásra és az adatzárolásra is képes.

Kipróbálás és értékelés

PC-telepünket egyszerű megjelenítő programrendszer összeállításával tettük próbára. A VR Juggler átjárható programszerkezetet használtuk, amely többféle programozási absztrakciót nyújt különböző megjelenítők felületeihez, számító és nyomkövető rendszerekhez és számos kapcsolattartó eszközhöz (interaction devices). A programnak, amelyet terepmodell-megjelenítésre használunk, több ábrázolási módja is létezik, ezek egyike a sztereó. A programot újra lehet fordítani, ha különböző számítási szerkezeteken akarjuk működtetni és futásidőben újra beállítható különféle alapértelmezett eszközbeállításfájlokkal akarjuk használni. Az alkalmazás könnyedén együttműködött a teleprendszerrel, és jobban teljesített, mint azt vártuk. A megjelenítés minősége és a sztereónézet összehasonlítható egy azonos programot futtató, három falat alkalmazó IR2 grafikájú SGI Onyx 2 gép teljesítményével. Sőt a telep nagyobb teljesítménnyel volt képes megjelenítni az adatokat. Az *ábránkon* a telepprogramot mutatja be, amint a négyfalas megjelenítőrendszer három oldalfalán fut.

Összegzés és tanulságok

A PC-telepek alkalmazása mára már versenyképes, alacsony költségű választási lehetőségévé vált az egyetlen, nagyteljesítményű szuperszámítógép alapú megjelenítő alkalmazásnak. A PC-khez bőségesen találhatunk programokat is – például a VR Jugglert –, amelyek lehetővé teszik, hogy az alkalmazások többféle megjelenítőrendszert és beviteli eszközt kezeljenek.

Telepprobánk rámutat, hogy már meglévő drága nagyszámítógépünk leépítése és helyettesítése megvalósítható. Az eredmény megmutatta, hogy ilyesfajta átalakításokat a közeljövőben végre tudunk hajtani, és egyben reméljük, tapasztalataink másokat is arra indítanak majd, hogy kövessék példánkat.

Köszönetnyilvánítás

A szerző szeretné megköszönni a projekt támogatását és az következő személyek hozzájárulását: ***Aaron Bryden, Greg S. Schmidt, Ian Roth és J. Edward Swan II.***

A szakirodalom és a kapcsolódó címek a 43. CD Magazin/3D könyvtárában találhatóak.

Linux Journal 2002. december, 104. szám

Douglas B. Maxwell
(vrdeity@yahoo.com)
gépészmérnök, akit mostanában helyeztek át a newporti Naval Undersea Warfare Center (Tengerészet Tengeralatti Hadiközpontjának) Weapons and Countermeasure Control (Fegyverek és Ellencsapás-vezérlés) csoportjához Rhode Islandra. Azelőtt a Washington DC-i Naval Research Laboratory Virtual Reality Laboratóriumban dolgozott. Tervezéselemzésre, szimulációkra és virtuális környezetbeli oktatásra szakosodott.

Perlekedjünk Linux alatt?

Azt hiszem, erre nem lesz szükségünk – feltéve, hogy kábelek és modemek rengetegét akarjuk megszüntetni. Ezt a feladatot hivatott megoldani a Perle PCI-RAS Multi-Modem kártya, ami négy-, illetve nyolckapus változatban kapható. A mohó felhasználók egyszerre többet is berakhatnak, így 12–16 vagy akár több modemünk is lehet egyetlen gépben, ami nagy helymegtakarítást jelent a kiszolgálógép körül. Mi egy nyolckapus modemkártyát kaptunk kipróbálásra a Mission Critical Linuxtól. Biztosan akad olyan kedves olvasóink között, aki emlékszik még a régi, teljes hosszúságú kártyákra. Nos, miután kezembe vettem ezt a szép kivitelezésű kártyát, elfogott a nosztalgia: Vesa Local buszos VGA-kártya, párhuzamos kapus ISA-kártya – bizony-bizony, ezek mind használatban voltak egykoron. A modemkártya mérete ugyanekkora, ez azonban már PCI-buszhoz kapcsolódik. A doboz tartalma a következő volt:

- leírás a gyors használatbavételhez,
- CD-ROM a meghajtóprogramokkal,
- PCI-RAS modemkártya.

Felélesztés

A kártyát egyszerűen behelyezzük egy üres PCI-os helyre, és elindítjuk a gépet. Nekem ehhez a művelethez a kártya hossza miatt ki kellett vennem a ház elejében lévő ventilátort. A gépet elindítva a rendszermag a kártyát multimodem kártyaként önműködően felismerte, így a beállításával nem kellett bajlódnom. A kártya készítői szerencsére Conexant lapkákat használtak, ezzel biztosítva a gondtalan telepítést. Vezérlőprogramként a Hylafaxot választottam – könnyedén beállíthattam nyolc modememet, már csak egyetlen kérdés maradt nyitva: mire is használjam őket?
Úgy döntöttem, hogy négy kapu faxokat fog küldeni egymásnak, négy kapu pedig adatforgalmat fog bonyolítani egymás között. A vonalakat a szerkesztőség ISDN-központján keresztül állítottam be, így a mellékeket is használni tudtam. A kártya faxkiszolgálóként és „adatforgamistaként" is megállta a helyét. Ügyfélprogramként a gfaxot telepítettem – ez a program nagyszerűen együtt tud működni a Hylafax kiszolgálóval, beállítása és használata könnyű.

Összegzés

Aki faxkiszolgálót, esetleg betárcsázó kiszolgálót kíván üzemeltetni, annak nyugodt szívvel ajánlható ez a termék. A kipróbálás során egyetlen hiba keletkezett, az is beállítási hiba miatt, úgyhogy saját magamat rovom meg miatta! A kártyát egyébként SCO OpenServer, SUN Solaris, Linux és Windows alatt is használhatjuk.

A termék neve: Perle PCI-RAS Multi-Modem kártya
Ár: 382. 800 Ft + Áfa
Forgalmazó: Mission Cririca Linux Kft.
➲ http://www.missoncritical.hu
Gyártó ➲ http://www.perle.com

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

Kipróbálás és értékelés

PC-telepünket egyszerű megjelenítő programrendszer összeállításával tettük próbára. A VR Juggler átjárható programszerkezetet használtuk, amely többféle programozási absztrakciót nyújt különböző megjelenítők felületeihez, számító és nyomkövető rendszerekhez és számos kapcsolattartó eszközhöz (interaction devices). A programnak, amelyet terepmodell-megjelenítésre használunk, több ábrázolási módja is létezik, ezek egyike a sztereó. A programot újra lehet fordítani, ha különböző számítási szerkezeteken akarjuk működtetni és futásidőben újra beállítható különféle alapértelmezett eszközbeállításfájlokkal akarjuk használni. Az alkalmazás könnyedén együttműködött a teleprendszerrel, és jobban teljesített, mint azt vártuk. A megjelenítés minősége és a sztereónézet összehasonlítható egy azonos programot futtató, három falat alkalmazó IR2 grafikájú SGI Onyx 2 gép teljesítményével. Sőt a telep nagyobb teljesítménnyel volt képes megjelenítni az adatokat. Az *ábránkon* a telepprogramot mutatja be, amint a négyfalas megjelenítőrendszer három oldalfalán fut.

Összegzés és tanulságok

A PC-telepek alkalmazása mára már versenyképes, alacsony költségű választási lehetőségévé vált az egyetlen, nagyteljesítményű szuperszámítógép alapú megjelenítő alkalmazásnak. A PC-khez bőségesen találhatunk programokat is – például a VR Jugglert –, amelyek lehetővé teszik, hogy az alkalmazások többféle megjelenítőrendszert és beviteli eszközt kezeljenek.

Telepprobánk rámutat, hogy már meglévő drága nagyszámítógépünk leépítése és helyettesítése megvalósítható. Az eredmény megmutatta, hogy ilyesfajta átalakításokat a közeljövőben végre tudunk hajtani, és egyben reméljük, tapasztalataink másokat is arra indítanak majd, hogy kövessék példánkat.

Köszönetnyilvánítás

A szerző szeretné megköszönni a projekt támogatását és az következő személyek hozzájárulását: ***Aaron Bryden, Greg S. Schmidt, Ian Roth és J. Edward Swan II.***

A szakirodalom és a kapcsolódó címek a 43. CD Magazin/3D könyvtárában találhatóak.

Linux Journal 2002. december, 104. szám

Douglas B. Maxwell
(vrdeity@yahoo.com)
gépészmérnök, akit mostanában helyeztek át a newporti Naval Undersea Warfare Center (Tengerészet Tengeralatti Hadiközpontjának) Weapons and Countermeasure Control (Fegyverek és Ellencsapás-vezérlés) csoportjához Rhode Islandra. Azelőtt a Washington DC-i Naval Research Laboratory Virtual Reality Laboratóriumban dolgozott. Tervezéselemzésre, szimulációkra és virtuális környezetbeli oktatásra szakosodott.

Perlekedjünk Linux alatt?

Azt hiszem, erre nem lesz szükségünk – feltéve, hogy kábelek és modemek rengetegét akarjuk megszüntetni. Ezt a feladatot hivatott megoldani a Perle PCI-RAS Multi-Modem kártya, ami négy-, illetve nyolckapus változatban kapható. A mohó felhasználók egyszerre többet is berakhatnak, így 12–16 vagy akár több modemünk is lehet egyetlen gépben, ami nagy helymegtakarítást jelent a kiszolgálógép körül. Mi egy nyolckapus modemkártyát kaptunk kipróbálásra a Mission Critical Linuxtól. Biztosan akad olyan kedves olvasóink között, aki emlékszik még a régi, teljes hosszúságú kártyákra. Nos, miután kezembe vettem ezt a szép kivitelezésű kártyát, elfogott a nosztalgia: Vesa Local buszos VGA-kártya, párhuzamos kapus ISA-kártya – bizony-bizony, ezek mind használatban voltak egykoron. A modemkártya mérete ugyanekkora, ez azonban már PCI-buszhoz kapcsolódik. A doboz tartalma a következő volt:

- leírás a gyors használatbavételhez,
- CD-ROM a meghajtóprogramokkal,
- PCI-RAS modemkártya.

Felélesztés

A kártyát egyszerűen behelyezzük egy üres PCI-os helyre, és elindítjuk a gépet. Nekem ehhez a művelethez a kártya hossza miatt ki kellett vennem a ház elejében lévő ventilátort. A gépet elindítva a rendszermag a kártyát multimodem kártyaként önműködően felismerte, így a beállításával nem kellett bajlódnom. A kártya készítői szerencsére Conexant lapkákat használtak, ezzel biztosítva a gondtalan telepítést. Vezérlőprogramként a Hylafaxot választottam – könnyedén beállíthattam nyolc modememet, már csak egyetlen kérdés maradt nyitva: mire is használjam őket?
Úgy döntöttem, hogy négy kapu faxokat fog küldeni egymásnak, négy kapu pedig adatforgalmat fog bonyolítani egymás között. A vonalakat a szerkesztőség ISDN-központján keresztül állítottam be, így a mellékeket is használni tudtam. A kártya faxkiszolgálóként és „adatforgamistaként" is megállta a helyét. Ügyfélprogramként a gfaxot telepítettem – ez a program nagyszerűen együtt tud működni a Hylafax kiszolgálóval, beállítása és használata könnyű.

Összegzés

Aki faxkiszolgálót, esetleg betárcsázó kiszolgálót kíván üzemeltetni, annak nyugodt szívvel ajánlható ez a termék. A kipróbálás során egyetlen hiba keletkezett, az is beállítási hiba miatt, úgyhogy saját magamat rovom meg miatta! A kártyát egyébként SCO OpenServer, SUN Solaris, Linux és Windows alatt is használhatjuk.

A termék neve: Perle PCI-RAS Multi-Modem kártya
Ár: 382. 800 Ft + Áfa
Forgalmazó: Mission Cririca Linux Kft.
➲ http://www.missoncritical.hu
Gyártó ➲ http://www.perle.com

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

Képzeletbeli teremtmények irányítása Linuxszal

Vajon hogyan képes a beágyazott Linux kielégíteni a The Jim Henson Company robotjainak és számítógépes 3D-modelljeinek sokszínű, valós idejű igényeit?

A Jim Henson Company képzeletbeli karakterek megalkotójaként vált ismertté. Az egyszerűbb karakterek, mint a Muppet Show tagjai, nem igényelnek csúcsszintű műszaki eszköztárat, de a robotok – legyen szó akár kisegérről, akár sárkányról, nem is beszélve a 3D-s számítógépes grafikával életre hívott teremtményekről – már igen. Élőben, valós időben kell mozgatni őket, így látszólag kapcsolatba léphetnek a valódi színészekkel és filmre vehetők – csakhogy ehhez különleges műszaki kihívásoknak kell megfelelni.

Jim Henson eredeti célkitűzéseinek egyike az volt, hogy minden karaktert egy ember irányíthasson, ami így sokkal szabadabban viselkedhet, letisztultabb személyiséget kaphat; ezeket a célokat sokkal nehezebb elérni, ha egy-egy karakterrel többen dolgoznak. Bámulatos, hogy amikor végre sikerül életre kelteni egy ilyen karaktert, mindenki elfeledkezhet róla, hogy ki vagy mi irányítja valójában, és teljes természetességgel lépnek vele kapcsolatba. A színészek és nézők úgy kezdenek el beszélgetni egy kutyával, békával vagy hóemberrel, mintha ő is ember lenne.

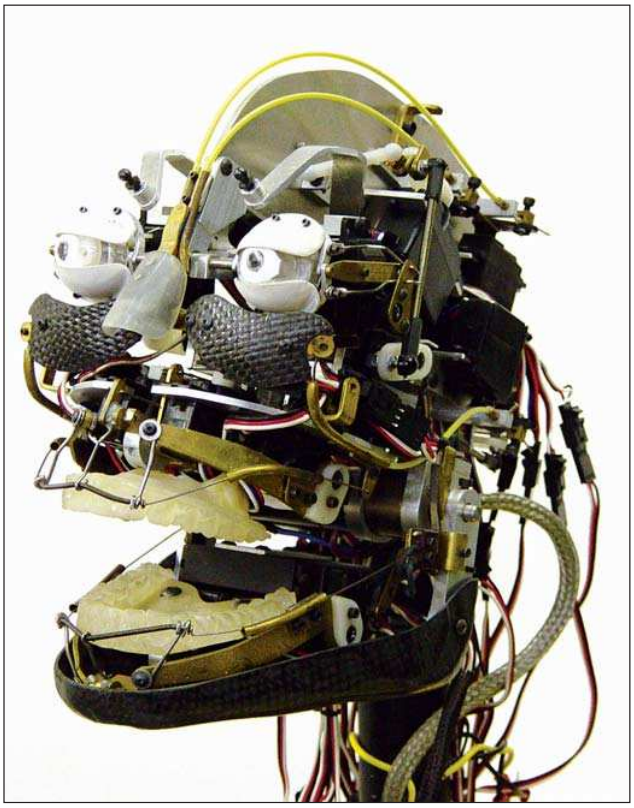
A szervomotorok az 1980-as évek elején terjedtek el a robotokban, ám a növekvő számú szervorobot irányítása egyre nehezebbé vált, így számítógépes vezérlést kezdtek el fejleszteni. Az elmúlt 15 év alatt a Jim Henson Creature Shop vezérlőrendszerek nemzedékeit fejlesztette ki, köztük olyat is, ami 1992-ben elnyerte a Technical Achievement Academy Award díjat. A legújabb Henson Performance Control System (HPCS) a korábbi rendszerekből a legjobb jellemzőket vette át, miközben az összes elérhető új számítástechnikai megoldást is magába olvasztja – így került képbe a Linux is.

A rendszer fejlesztése ***Jeff Forbes*** számítástechnikai-elektronikai igazgató vezetésével kezdődött 1998 elején. Az elképzelés az volt, hogy egy általános felépítésű rendszer a cég minden igényét ki tudná elégíteni. ***Steve Rosenbluth*** ekkor csatlakozott a tervezethez vezérlőrendszer-fejlesztőként, ***Michael Babcock*** pedig a multimédiás programozásért felelt. Igényeink azonban egyre sokrétűbbek lettek, és egyedül a Linux volt képes arra, hogy változásaikat különösebb gond nélkül kövesse.

A rendszernek kétféle hátteret, világot is támogatnia kellett: a robotizált bábokat és a számítógépes grafikát. A karakterek tehát valódi robotbábok, illetve poligonokból és képpontokból felépülő virtuális modellek is lehetnek. A két világ különkülön, de akár együtt is kezelhető.

Miután az egy bábuhoz tartozó programok a helyükre kerültek, akár a területen teljesen kezdők is néhány óra alatt begyakorolhatják a robotbábok irányítását. A beviteli eszközök kezelése a zenéléshez hasonlítható. A báb kezelője egy idő után eljut egy pontra, amikor már nincs tudatában annak, hogy pontosan mit is tesz – egyszerűen csak csinálja.

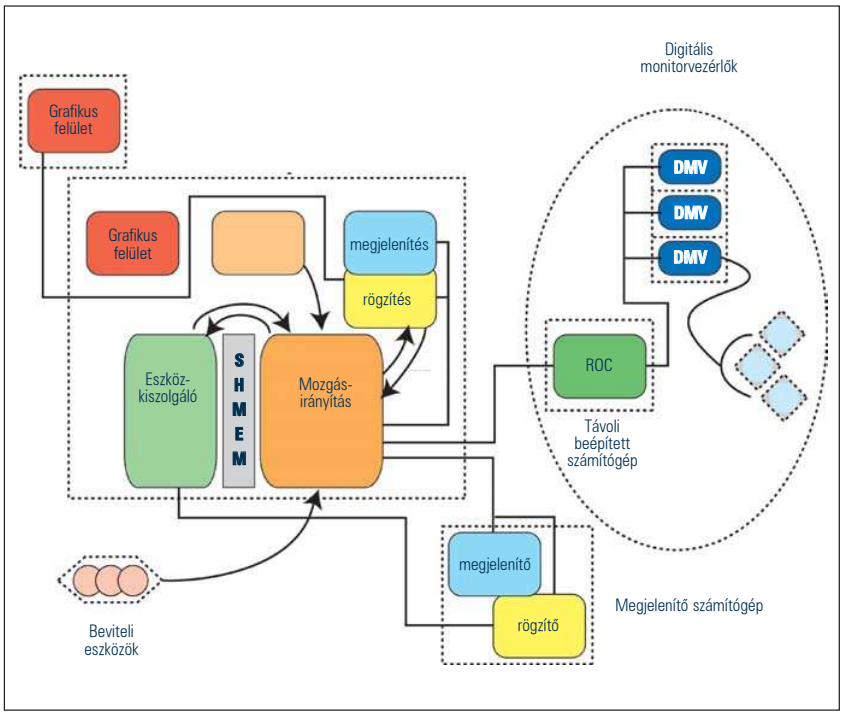
A Henson beviteli eszközök nem mozgásrögzítő megoldások. A mozgásrögzítés egyrészt csak a kezelő követésére képes, másrészt jellemzően nem programozható. A mozgásrögzítésnél a kezelőnek – avagy a színésznek – például a karját közvetlenül a karakter karjának feleltetik meg, a térdét a térdének és így tovább. A kezelő ezeket a megfeleltetéseket nem módosíthatja és nem bővítheti. A Henson beviteli rendszere viszont nem közvetlen megfeleltetést végez, és a felhasználó által újraprogramozható. A beviteli eszközök elvont értelmezést is támogassanak például lehetséges, hogy a művész mutatóujjának a mozgása a teremtmény egész arckifejezését a barátságosból arányosan a gúnyosba viszi át. A kezelő emellett könnyedén átprogramozhatja a karakter mozgását, akár jelenetek közben is. Mozgásrögzítéssel elég nehéz lenne például egy nyolckarú polipot eljátszani, ezzel a vezérlőrendszerrel viszont semmi gondot nem okoz.



Egy jellegzetes robotbáb feje szervomotorokkal és gépkarokkal

A vezérlő számítógép és a mozgásmotor

A rendszer lelke az RTLinuxot futtató vezérlő számítógép, amely a mozgásokkal kapcsolatos adatok feldolgozását és a robotokhoz való elosztását végzi. A vezérlő-számítógépen futó, a mozgásokat egyesítő algoritmust megvalósító folyamatot mozgásmotornak nevezzük, ezt Steve Rosenbluth írta C++ nyelven. A kezelők mozdulatait jeladók továbbítják a mozgásmotornak, és ez vezérli a hálózatba kapcsolt bábokat. A mozgásmotor az élő adatokat különféle algoritmusokkal dolgozza fel, és kiszámítja a gépkarok végleges helyzetét. A gépkarok úgymond a bábok izmai, a robotizált bábokban lehetnek elektromechanikus vagy hidraulikus szervomotorok, a számító-



A PCS/HDPS rendszer vázlata

gépes karakterben pedig a poligonháló alakváltoztatásait kell elvégezni. A mozgásegyesítési viszonyok alkalmazásból módosíthatók; lehetnek egy–több és több–egy jellegűek, az összetettebb keveréseket pedig felsőbb szinten lehet végrehajtani.
Az olyan fizikai hatások, mint a nehézkedés vagy a simítások, akkor adhatók hozzá a mozgásadatokhoz, amikor a mozgásmotor feldolgozza őket.

Az eszközkiszolgáló

A mozgáskeverő algoritmusokat az eszközkiszolgálónak nevezett folyamat által rendelkezésre bocsátott programok biztosítják. Az eszközkiszolgáló közvetlen hozzáféréssel rendelkezik a megosztott memóriában található objektumokhoz és adatcsomagokhoz, és létrehozza a kapcsolatot az élő mozgásadatok és a mozgásmotor által rajtuk futtatott algoritmusok között. Tervezését Michael Babcock végezte, és mint neve is utal rá, egy olyan aszinkron kiszolgálóról van szó, ami a vele grafikus felületen érintkező ügyfelekkel létesít kapcsolatot.

A grafikus felhasználói felület

A grafikus felület egy „nem túl vékony” ügyfél, ami egy foglalaton (socket) keresztül csatlakozik az eszközkiszolgálóhoz. A Michael Babcock által írt jelenlegi alkalmazás GTKmm alapú, és a ***Robert McNally*** tanácsadó által tervezett magas szintű felületterv alapján készült. A kiszolgálómódú felépítés lehetővé teszi, hogy a műszaki munkatársak vagy a bábkezelők a grafikus felületből akár több példányt is futtassanak.
A forgatások idején előfordul, hogy a műszaki munkatársak különféle kisegítő tevékenységekkel segítik a bábkezelőket – a hálózati felületnek hála ezt úgy tehetik meg, hogy közben nem kell másokat a rendszerből kirúgniuk. A grafikus felület és az eszközkiszolgáló egyedi protokollt használva tartja a kapcsolatot egymással. Annak lehetősége, hogy egy műszaki munkatárs a vezérlőrendszert távolról felügyelje, különösen fontos számunkra, hiszen a forgatások a világon bárhol lehetnek; a The Creature Shopban lévő támogatási személyzet mégis teljes hozzáféréssel rendelkezik, így el tudja hárítani a hibákat, illetve egyéb jellegű segítséget tud nyújtani.
Amint *az ábrán* is látható, a vezérlőrendszer mozgásmotor mögötti–alatti része módosulhat. A robotok vezérlését beágyazott processzorral rendelkező, távoli helyszíni számítógépekkel is el tudjuk végezni, a 3D-s számítógépes modellek életre keltése pedig egy „nézegető” háttérrendszerrel történik.

A távoli helyszíni számítógép

A mozgásmotor minden képkockánál kap egy pillanatképet az analóg–digitális átalakítóktól a fizikai beviteli eszközök állapotáról, a karakter beállításai alapján lefuttatja a mozgáskeverő algoritmusokat, majd az adatokat átadja a távoli helyszíni számítógépes ügyfeleknek.
Esetünkben a távoli helyszíni számítógépek DR DOS-t futtató beágyazott PC-k voltak. Steve Rosenbluth félig-meddig objektumközpontú C-kódot írt a távoli helyszíni gépekhez. A programozás során mindvégig szem előtt kellett tartania, hogy a kód gyorsasága és megbízhatósága alapvető fontosságú. A távoli helyszíni számítógépek protokollja lehetővé teszi, hogy egy összeköttetést – ami jelenleg egy RS-232 kapcsolat – több eszköz is használjon. A meglehetősen idősnek számító kapcsolattípus ugyan nem éppen korszerű eszköz, ám ezt a legkönnyebb üvegszál, rézkábel vagy rádiós összeköttetés felett használni. Az RS-232 valós idejű felület előre megjósolható jelekkel és késleltetésekkel, és nekünk erre van szükségünk. Ha egy robot vezetékkel rendelkezik, akkor üvegszál, egyéb esetben rádiós összeköttetés felett továbbítjuk az RS-232-jeleket.

Nézegetők

A számítógépes grafikai nézegető az a programmodul, ami a számítógépes modelleket leképezi és megjeleníti a képernyőn. Az ábrán szereplő nézegető lehet a számítógépes grafikai modellezőcsomagok és játékok motorjaiból álló gyűjtemény tagjainak egyike, ha elég gyorsan le tud képezni egy élő OpenGL-környezetet. A Henson Company háza táján ***Hal Bertram*** elektronikai részlegvezető a London Creature Shopban az 1990-es évek elején honosította meg a számítógépes karakterek készítését. Mostanában bizonyos PC alapú grafikai alkalmazásokat – Discrete 3D Studio Max, Side Effects Houdini, Kaydara Filmbox és Alias/Wavefront Maya – használunk, amelyekhez Michael Babcock készítette az irányítástechnikai beépülő modult.
Az, ami a vezérlésben egy gépkar, a számítógépes grafika világában egy skaláris csatorna adata, amely a 3D-poligonháló alakváltozását vagy módosulását írja le. A vezérlő-számítógép mozgásmotorja felől UDP-protokoll feletti kapcsolaton keresztül élő mozgásadatok kerülnek a nézegetőbe. A nézegetők a mozgásmotor nézőpontjából távoli helyszíni számítógépként viselkednek, legalábbis ugyanazt a protokollt használják. Kétprocesszoros AMD Athlon gépekkel körülbelül 100 képkocka/másodperc frissítést sikerült elérnünk, ami ugyanabban a jelenetben akár több karakter mozgatását is lehetővé teszi. ***Jeff Christie*** karakterfejlesztési műszaki igazgató egy 3D-modellbe-

állítás tökéletesítésével segített gyors, élethű megjelenést kicsalni a számítógépes grafikai modellekből – így vált teljessé a kép. A nézegető egyidejűleg több mozgásmotorral is kapcsolatot tud tartani, tehát egy olyan helyszínen, ahol több karakter is található, mindegyikük saját vezérlőrendszerrel és kezelővel rendelkezhet, akár egy hálózati játékban.

A felvevő

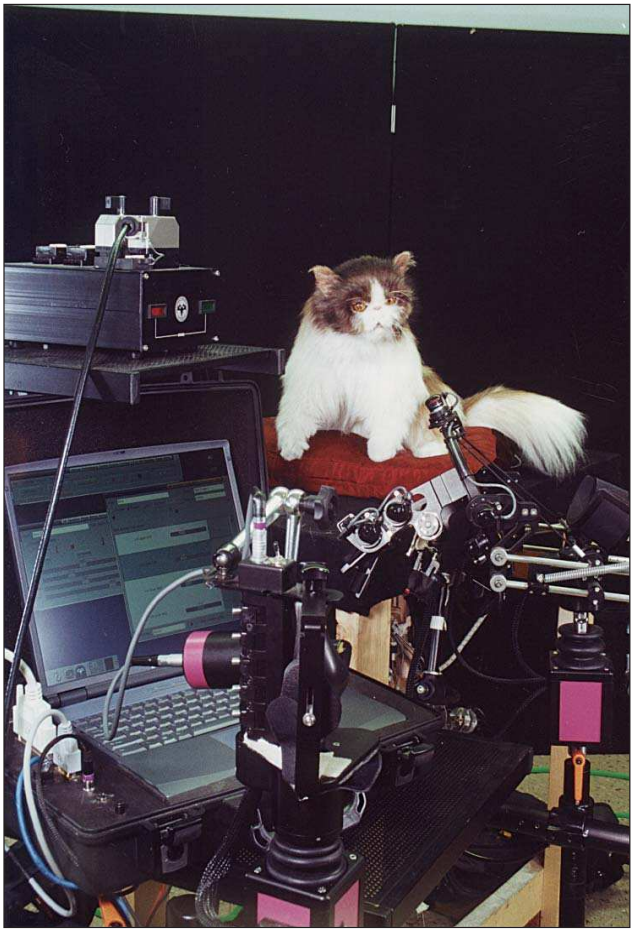
A mozgások és hangok rögzítését és visszajátszását egy nemlineáris multimédia-szerkesztő, a Michael Babcock által fejlesztett felvevő végzi. Az eszköz felépítését Michael és Steve dolgozták ki, és egy többszálú, a mozgásmotorral UDP-kapcsolatot tartó folyamatból áll. A felvevő a mozgásmotorral összhangban dolgozik, kimenetét ugyanis egy távoli helyszíni ügyfélgépnek adja át, illetve a tárolt adatok visszasugárzását is elvégzi a mozgásmotor felé, ami továbbítja őket a többi távoli helyszíni ügyfélgépnek. A hálózati szervezés lehetővé teszi, hogy minden folyamat – a többi zavarása nélkül – saját időzítéssel, be- és kiviteli jellemzőkkel rendelkezzen, ahogy az eszközkiszolgáló–mozgásmotor kapcsolatnál ezt láttuk.
Mivel a felvett mozgások élőben rendezhetők és visszajátszhatók, a művész egy-egy jelenetet úgy állíthat össze, mintha többsávos hangfelvételt készítene. Ez különösen a szájmozgás összhangba hozatalát igénylő helyzetekben hasznos, ahol a karakter ajkának mozgását más időpontban is ki lehet dolgozni, majd vissza lehet játszani, miközben a művész a bábu egyéb részeit élőben irányítja.
Dan Helfman egy hangfelvevő résszel járult hozzá az SDL-hez, a felvevőben használt nyílt forrású multimédia API-hoz.

A kapcsolatkezelő és a bábuügyfelek

A mozgásmotor kapcsolatkezelő modulja végzi a hálózati szórást és a távoli helyszíni ügyfelekkel való kapcsolattartást, legyen szó bármilyen hálózati kapcsolatról vagy megvalósításról. Így egy-egy művész egy karakternek akár több megvalósítását is vezérelheti. Előfordul például, hogy egy robotmacska és annak számítógépes modellje egyszerre mozog. Míg a karakter testét és arcát veszi a kamera, a számítógép mozgatja a száját, és a két képet azonnal össze is lehet vágni. Minden megtestesülés tehát azt végzi, amire a leginkább alkalmas. A valódi „lény” összetett megvilágítása, fizikája és a számítógépes szájmozgás az utólagos feldolgozás során, a film véglegesítése előtt tovább finomítható. Az élő előnézet lehetővé teszi, hogy a rendező a bábok mozgatását közvetlenül irányítsa, a színészek pedig bábtársaikkal kapcsolatba léphetnek.

Feldolgozás

A mozgásmotor, az eszközkiszolgáló, a grafikus felület és a felvevő feladatkörei élesen elkülönülnek egymástól. Mivel a bonyolult multimédiás és hálózati modulok olyan programmegoldásokat igényelnek, amelyek kedvezőtlenül hathatnak a feldolgozás ütemezésére és üzembiztosságára, Steve Rosenbluth és ***Tim McGill*** olyan rendszert hozott létre, amely egyfajta falat képez a mozgásmotor körül. A cél az volt, hogy a mozgásmotor a lehető legegyszerűbb legyen, így folyamatos működését a lehető legkevesebb dolog veszélyeztesse. Az eszközkiszolgáló – amelynek nagyra és bonyolultra hízása valószínűsíthető – ezzel szemben leállhat vagy újraindulhat, ez azonban nem befolyásolja a mozgásmotor működését. A grafikus felület szintén szükség szerint indítható és állítható le, ez sem az eszközkiszolgáló, sem a mozgásmotor, sem a felvevő működését nem érinti. Ennek érdekében a rendszer folyamatmodulokból épül fel, amelyek Unix IPC-hívásokkal és hálózaton keresztül tartják egymással a kapcsolatot.
Az eszközkiszolgáló és a mozgásmotor egy közös, megosztott System V memóriablokkal rendelkezik, így a legfontosabb adatobjektumok módosítása haladéktalanul elvégezhető. Az ütemezés szempontjából létfontosságú üzeneteiket két FIFO-n keresztül továbbítják. A mozgásmotor és a felvevő az adatokat UDP hálózati foglalatok használatával, adatfolyamok révén valós időben továbbítja egymásnak. A mozgásmotort az egyik legfontosabb elem, meghibásodása kellemetlen következményekkel járhat. Ha forgatás közben következik be a leállás, a



PCS összeállítás robotizált bábok irányításához, a bemeneti vezérlés és a grafikus felületet futtató hordozható vezérlőszámítógép

költségek óránként több ezer dollárra rúghatnak. A filmipar másik jellegzetessége, hogy a színészek és a segítők közeli kapcsolatban lehetnek a gépi karakterekkel. Elég érdekes lenne, ha egy robotkutya addig harapná valamelyik színészt, míg a műszaki munkatárs be nem jelentkezik, és újra nem indítja a megfelelő programot. Ez az, amiért a grafikus felület és az egyéb nélkülözhető részek kikerültek a mozgásmotor kódjából. Mivel rendkívül kényes körülmények között dolgozunk, a Linux üzembiztos működése fontos előnyt jelent számunkra.
A független folyamatokból felépülő rendszer tette lehetővé azt is, hogy a fejlesztést és kipróbálást egyéni programozók modulonként, a saját területükre összepontosítva végezzék. Így alkalmat kaptak arra, hogy biztonságosan használjanak egyedi, sokszor élvonalbeli programozási megoldásokat, amelyek más moduloknál esetleg szükségtelennek vagy oda nem illőnek bizonyultak volna.

Ütemezés és időzítés

Az egyes összetevők időzítési követelményei változók. A mozgásmotort pontosan 60 Hz-es gyakorisággal kell meghívni, hogy a robotok mozgása egyenletes legyen. Ha a motoroknak kiküldött helyzetleíró adatok időtartományban ingadoznak, hirtelen gyorsulások jelennek meg bennük, és a robotlények karjai remegni kezdenek. Mindenképpen szerettük volna a legnagyobb pontosságot elérni a mozgásmotor 60 Hz-es meghívásában, ám ez csak harmadszori nekifutásra jött össze. Először időzített, programozott megszakításokat használtunk, a mozgásmotort a megszakítást kezelő alkalmazásként hívtuk meg. Emellett `POSIX.1b SCHED_FIFO` elsőbbségi sorrendmeghatározást használtunk, hogy miután a futása megkezdődött, a rendszermag ütemezője ne vegye el az erőforrásokat a mozgásmotortól. Így a mozgásmotor felhasználói szinten, és ami még fontosabb, hibakeresőben is remekül futott. A hasonló riasztáskezelők viszont két szempontból hátrányosak:

1. Időzítésük akár több időosztásnyit is ingadozhat, ha a rendszermag ütemezője terhelt.
2. Ütemezésük a rendszermag időzítési egységeitől függ, ezért pontosan nem jósolható meg.

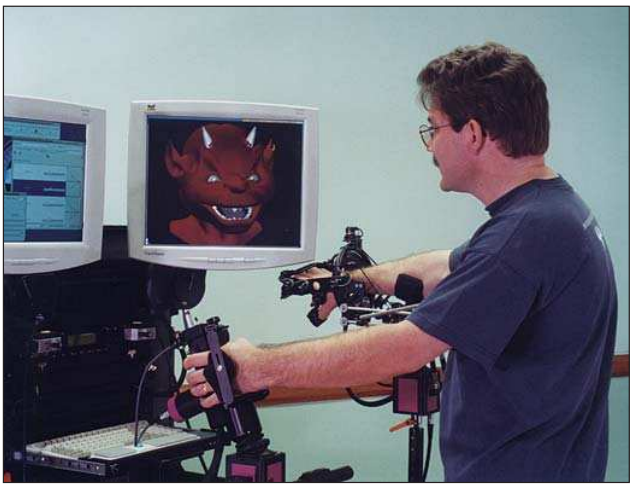
A rendszermagokat tehát újrafordítottuk, hogy növeljük a rendszermag időzítési egységeinek a számát, ezt az utóbbit két okból kifolyólag szerettük volna: a mozgásmotor futtatásának időzítését közelíteni akartuk a 60 Hz-es értékhez, illetve biztosak akartunk lenni abban, hogy a kisebb fontosságú vezérlőfolyamatok újraütemezése gyakrabban történik meg, így jobban követni tudják a mozgásmotor állapotváltozásait.

Második próbálkozásként RTLinux alatt egy szigorúan valós idejű, periodikus futású szálat hoztunk létre 60 Hz-es órajellel. Az RTLinux területen futó szál körülbelül három nagyságrenddel pontosabb időzítéssel futott. Ezt a szálat szigorúan valós idejű időzítőnek kereszteltük el – ő adja az ütemet. Amikor szóhoz jut, egy RTLinux FIFO vezetékbe RTLinux területről egy jelzőt helyez el, amelynek megérkezésekor a felhasználói területen futó, a FIFO által megakasztott mozgásmotor feléled. Ugyan a mozgásmotor meghívása továbbra is a Linux-rendszermag feladata, ezzel a megközelítéssel a vártnál pontosabb eredményt sikerült elérni, ugyanis a be- és kiviteli műveletek nagyobb fontosságúak a rendszermag számára, mint a jelzők kezelése. A FIFO alapú megoldás átlagos késleltetése – ha más, a rendszert leterhelő program nem futott – 40 μsec alatt volt, tehát a mozgásmotor meghívásának gyakorisága a processzor időzítői által lehetővé tett mértékben megközelítette a 60 Hz-es értéket.

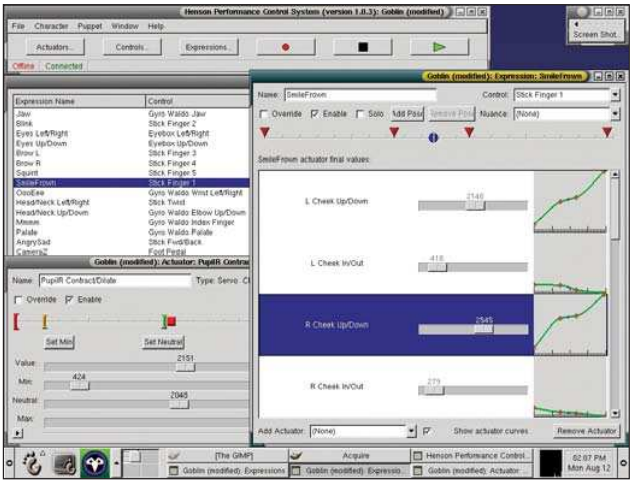
Nagyobb terhelés esetén a rendszermag nem feltétlenül indítja el időben a mozgásmotort, a rendszer tehát nem szigorúan vett, előre tervezhető módon valós idejű, habár éles környezetben, munka közben is remekül működött. Egy kétprocesszoros Athlon alaplappal fenntartható a mozgásmotor meghívásának pontossága, miközben elfut rajta a grafikus felület, az eszközkiszolgáló és a folyamatosan OpenGL-színhelyeket leképező nézegető!

Harmadik és egyben utolsó próbálkozásként egy olyan RTLinux-bővítésnek az elkészítésével bíztuk meg az FSMLabst, amely lehetővé tette a felhasználói területen futó Linux-folyamatok előre kiszámítható ütemezését. A PSC-nek nevezett rendszer révén képesek leszünk arra, hogy egyfajta ugrást hajtsunk végre az RTLinux rendszeres időközönként futó szálából a felhasználói területre, ahol a mozgásmotor fut, majd a műveletsor az RTLinux területre való visszalépéssel fejeződjék be.

A szerződés egy pontja értelmében a forráskód RTLinux-csomagként, nyílt forrással bárki számára elérhető.



Bábkezelő munkában a HDPS digitális bábállomásán



Képernyőkép a HDPS grafikus felületéről

Beviteli eszközök

A vezérlőrendszerben használt beviteli eszközök könnyű és rugalmas használatra tervezett lineáris potenciométerekből állnak össze. Kialakításuknál az elsődleges szempont a bábkezelők által megkívánt mozgások előállításának megkönnyítése volt. Eleinte nagyra törő terveink voltak: az egész rendszert egy linuxos hordozható gépen akartuk futtatni, amit könnyedén a forgatások helyszínére lehetett volna vinni. Csak akkor kezdtünk el 19”-os, állványra szerelt vezérlőrendszereket építeni, amikor CGI-megbízásokat teljesítettünk.

A hordozható gépek kapcsán a legnagyobb gond az volt, hogy 64 analóg csatornát kellett bevezetni egyetlen gépbe. Nem volt megfelelő A/D-átalakító illesztőprogram, így Steve Rosenbluth írt egyet a Computer Boards DAS16s/16 típusához. Még ma sincs olyan PCMCIA A/D-átalakító kártya, amely 16 csatornánál többel bírna, ezért Steve egy külső, analóg multiplexert is tervezett. A multiplexert minden képkockánál (16,6 ms) mind a négy bankon végig kellett kapcsolni – ezt a feladatot az RTLinuxra bíztuk, amit az 20 μ alatti pontossággal, előre kiszámítható módon végzett el.

Ugyan alacsony szintű A/D-átalakítóhoz dicső dolog illesztőprogramot írni, mi jobban is fel tudtuk volna használni kutatási–fejlesztési erőforrásainkat. 2000-ben örömmel értesültünk arról, hogy a United Electronics Industries Linux és RTLinux

Állványra szerelt HDPS összeállítás háromdimenziós, számítógépes karakterekhez – látható a kézi vezérlés, a grafikus felület és a nézegető

illesztőprogramot is kínált PCI-felületre csatlakozó Powerdaq A/D-átalakító kártyáihoz. 64 csatornás PD2-MF-64-333/16L kártyájuk kifogástalanul működött, és igényeinket az illesztőprogram fejlesztésekor az UEI figyelembe is vette.

Robotvezérlő eszközök

Ugyan mindvégig készen kapható termékeket igyekeztünk használni, a különleges igényeinknek megfelelő eszközöket nekünk kellett megterveznünk. A legtöbb gyártó által „kicsinek" gondolt dolgok egyszerűen nem férnek el például egy robothörcsög belsejében.
A digitális motorvezérlő (Digital Motor Controller) egy nagyjából két postai bélyeget kitevő méretű eszköz, a robotizált bábok vezérlésének kulcsfontosságú eleme. PWM- (Pulse With Modulation) jelek tucatjait juttatja el a motorokhoz a robot belsejében. ***Glenn Muravsky*** segítségével Steve egy egylapkás, a Texas Instruments 320LF2406 motorvezérlő DSP-jére épülő gépet tervezett. A lapkában található párhuzamosságnak hála olyan dolgokat is meg lehet vele tenni, amit egy PC-s adatbusszal nem. Steve egy ciklikus PID-algoritmust is készített az egyedi szervomotorok vezérlésére.

Miért pont Linux?

A Linuxot eredetileg, rendszermagszinten nem szigorúan valós idejűre tervezték, ám a valós idejű kiterjesztésekkel elértük azt, hogy a fontosabb feladatok kellő elsőbbséget élvezzenek, miközben továbbra is egy általános célú operációs rendszert használhattunk.
Időosztásos többfolyamatúság, memóriavédelem, a folyamatok közötti kapcsolattartási lehetőség, hálózati és multimédiás API-k – meglétük alapvető feltétel a többi, nem szigorúan valós idejű dologhoz, amit meg akartunk valósítani. Úgy véltük, az RTLinux szerkezete, amelyben maga a Linux a legkisebb fontosságú feladatként fut, számos támogatási alkalmazásunk használatát lehetővé teszi azon a gépen, amelyen a valós idejű feladatok is futnak.
A legnagyobb kihívást nem valamelyik adott rendszerösszetevő kezelése jelentette, hanem a rengeteg eltérő követelmény összehangolása úgy, hogy közben a kirakó egyik elemének igényeit se kelljen csökkenteni. Nem szigorúan valós idejű kódunknak esélyt kellett adni a futásra; mozgásvezérlő kódunk futását kisebb fontosságú feladatok nem szakíthatták meg; a létfontosságú kódrészeket védenünk kellett a kisebb fontosságúaktól; általános rendszermag-szolgáltatásokat kellett alkalmazunk; a segédprogramokat terjesztésünk részeként kellett használnunk. Az operációs rendszer magja által nyújtott lehetőségeknek, illetve a köré szerveződött segédprogramoknak hála, a fejlesztés felgyorsult, hiszen nem kellett mindenből a saját változatunkat elkészítenünk. Ha bármi gondunk volt, tudtuk, hogy a forráskód elérhető, így magunk irányítjuk a sorsunkat. Az összes elvárásunkat figyelembe véve lényegében a Linux volt az egyetlen választási lehetőségünk.
Megjegyezném, hogy vezérlőrendszereink az elmúlt három év során folyamatosan működtek, az újraindítások között eltelt átlagos idő több hónap volt. Gyakran szükség van rá, hogy a gépeket egyik helyről a másikra költöztessük, de forgatás közben egyik vezérlőrendszer sem állt le soha.

Munkáink

Hol látható munkánk eredménye? Először 1999-ben léptünk színre a Webisodes című alkotásban, amelyben a Henson ➲ http://www.muppetworld.com címen elérhető weblapján is látható karakterek szerepelnek. Rengeteg bemutatót tartottunk kiállításokon, köztük a SIGGRAPH 2000 megnyitásán egy interaktív számítógépes békakarakterrel, Brekivel (a Muppet Showból) jelentünk meg.
Az új vezérlőrendszer a Walt Disney Snowdogs című filmében, 2001-ben kezdte meg pályafutását. A film főszereplőjét, a Demon nevű huskyt a nehezebb jelenetekben egy robot helyettesíti – tessék kitalálni, melyek ezek a jelenetek! Mint reméltük is, a rendszer remekül működött. A Stuart Little 2 2001-es forgatásán egy robotvadászsólymot keltettünk életre, Horace D'Fly pedig a Henson hamarosan megjelenő „Kermit's Swamp Years" című filmjének egyik számítógépes karaktere. Jelenleg azt tervezzük, hogy a Warner Brothers' Kutyák és macskák című filmjének folytatásában – továbbá egyéb robotokat igénylő forgatásokon is – használjuk a rendszert. A szórakoztatóipar részéről nagy az érdeklődés: számítógépes grafikai karaktereinket a közeljövőben filmekben, tévéműsorokban és videojátékokban szeretnék használni. Találkozunk tehát a moziban!

Linux Journal 2002. november, 103. szám

David Barrington Holt
A Jim Henson Creature Shop Los Angeles-i igazgatója. Londonban ipari tervezésből szerzett kitüntetéses diplomát, később divattervezéssel, grafikával, fényképezéssel és műszaki modellezéssel is foglalkozott.

Steve Rosenbluth
34 éves, nős és van egy macskája. 13 éve foglalkozik robotokkal és a velük kapcsolatos műszaki megoldásokkal. Korábban bábfilmkészítést, szobrászatot, elektronikát, programozást és kisvállalkozás-fejlesztést tanult.

Michael Babcock (michael@kanji.com)
Montanában nőtt fel, hegyek, fák és szarvasok között. 1992 óta használ Linuxot. Programozási oldalról nézve érdeklik a többnyelvű programok, illetve az elvont programtervezési módszerek. A Jim Henson Creature Shopnál, Los Angelesben négy évig dolgozott egy bábvezérlő rendszeren, különösen annak hálózati részén, felhasználói felületén és 3D-s grafikai megoldásain.

Linux: felhasználói ismeretek I. és II.

Szerencsére egyre többen meg akarnak ismerkedni kedvenc rendszerünkkel. Az ismerkedésben viszont komoly nehézséget jelent az elérhető segítség hiánya. Az angol vagy német nyelvű irodalomhoz képest rendkívül le vagyunk maradva. Vannak azonban kiadványok, melyek kézenfogják az olvasót, és érthető stílusban tanítják a GNU/Linux-rendszerek használatának alapjait. ***Pere László*** két könyve is ezek közé tartozik. A szerző több éve oktatja a Linux használatát, mely az összeállított anyagon is meglátszik. A cél elsősorban egy olyan jól használható tankönyv létrehozása volt, amiből (akár egyedül, akár tanfolyam keretében) bárki elindulhat a profi felhasználóvá válás rögös útján. Az egyes témákra sajnos gyakran csak rövid bekezdések jutottak, a könyvek mentségére legyen szólva, hogy a kötetenként 250 oldalba ennél több nem is igen fér bele.

Mint minden rendes linuxos könyv, A Linux: felhasználói ismeretek első kötete is a karakteres felülettel kezd, elmagyarázza a könyvtárszerkezeteket, majd a héjak használatával foglalkozik, továbbá a héjprogramok lelkivilágával – elég részletesen ahhoz, hogy az érdeklődők akár kisebb héjprogramokat írjanak. Fontos szempont, amit gyakran elfelejtenek a szerzők, hogy a felhasználó kapcsolatot szeret, sőt akar tartani munkatársaival, barátaival. A kötet ennek mikéntjét is körbejárja (üzenetek, beszélgetés, információkérés stb.).

Előnye a könyvnek, hogy komoly anyagot igyekszik érinteni, amikor a hálózatok témájára is kitér, de (ismét a terjedelem miatt) csupán az alapokra és a fontosabb felhasználásokra tér ki. Ez kiemelten igaz a grafikus felületekkel foglalkozó részre, hiszen tízegynéhány oldalban nem lehet részletesen leírni egy teljes X Window-beállítást. A szerző elmondása szerint a komolyabb anyagrészeket a rendszergazdáknak szánt köteteibe tervezi, melyek valamikor 2003 második felében kerülnek csak a polcokra. És a végére egy kis érdekesség is került: harmincoldalnyi bevezetőt kapunk a LaTex világába.

A második kötet már komolyabb területre hajózik. Az adatokkal és azok kezelésével foglalkozik a szerző, könyve itt is elsősorban átgondolt tananyag benyomását kelti, semmint ismeretterjesztő könyvét. Ez szerintem előny, de a szerkezetből következik az is, hogy a könyvet tanfolyami anyagként érdemes végigrágni. Hogy mire is gondolok? Az emberek általában táblázatkezelőket használnak mindenféle számításhoz, míg a GNU-rendszerek nagy többségében található egy bc nevű „számológép”, ami akár programozható és rengeteg egyéb szolgáltatása van. Ha nem nőttünk hozzá a grafikus felülethez, remek eszközzé válhat.

I. kötet: A főbb témakörök

- A Linux felépítése
- Bejelentkezés a rendszerbe és a felhasználók azonosítása
- A konzol használata
- A könyvtárbejegyzések
- A héj használata: parancsok, utasításcsövek, a héj testreszabása és programozása
- Állománykezelés: hivatkozások, meghajtók beillesztése a rendszerbe, tömörítés és a biztonsági mentések készítésének rejtelmei
- Szabályos kifejezések és a szűrők használata
- Feladatvezérlés: a folyamatok irányítása és figyelés
- Kapcsolattartás és levelezés
- Szövegszerkesztés: a vi és a Midnight Commander használata
- A héjprogramozás alapjai
- A számítógép-hálózatok és a Világháló
- Grafikus felületek: az X Window rendszer és az ablakkezelők
- Kiadványszerkesztés Linux alatt

Adatok

Cím:	Linux: felhasználói ismeretek I.
Szerző:	Pere László
Kiadó:	Kiskapu Kft.
ISBN:	963 9301 31 x
Ár:	2660 Ft

II. kötet: A főbb témakörök

- A bc használata és nyelve
- Az awk használata parancssorból
- Az awk nyelve
- A relációs adatbázismodell
- Adatbázisok tervezése
- A PostgreSQL és a MySQL telepítése
- Az SQL parancssor
- Az SQL nyelv alapjai
- Összetett SQL parancsok
- Az adatbázis-rendszergazda
- Ügyfélprogramok készítése

Adatok

Cím:	Linux: felhasználói ismeretek II.
Szerző:	Pere László
Kiadó:	Kiskapu Kft.
ISBN:	963 930146 9
Ár:	2660 Ft

A bc elemzése után már egyértelműen adatkezeléssel és -feldolgozással találkozunk. Egyrészt az awk programot ismerhetjük meg (ez egy héjprogramokhoz remekül használható „programértelmező”, amellyel komolyabb szöveges és számítási feladatokat is végezhetünk), majd a PostgreSQL rendszert. Ez utóbbiról lapunk hasábjain is részletes bevezetőt olvashatunk ***Nyíri Imre*** tollából (48–51. oldal), úgyhogy bővebb kifejtésétől eltekintek.

Röviden összefoglalva: az első kötetet mindenkinek ajánlom, aki ***tényleg*** meg szeretne ismerkedni a Linuxszal, nemcsak kattintgatva, de a színfalak mögé nézve is; a második kötetet pedig mindenkinek javaslom, aki azt tervezi, hogy Linux alatt fog programozni, illetve azok számára, akik linuxos felületen adatokkal dolgoznak.

Szy György
(Szy.Gyorgy@linuxvilag.hu)
A Linuxvilág főszerkesztője, a Kiskapu Kiadó vezetője. Mindenki levelét örömmel várja.