© Kiskapu Kft. Minden jog fenntartva

A következő felvonás

Azt mondják, december végén nemcsak Jézus, de az új nap születését is ünnepeljük. Innen kezdve napról napra erősebb a fénye. Szerencsére már érezzük is, hogy mennyire igaz e régi mondás, bár még nincs tavasz, mégis, a nyüzsgő élet visszatért a világba.

Mit hoz a kikelet?

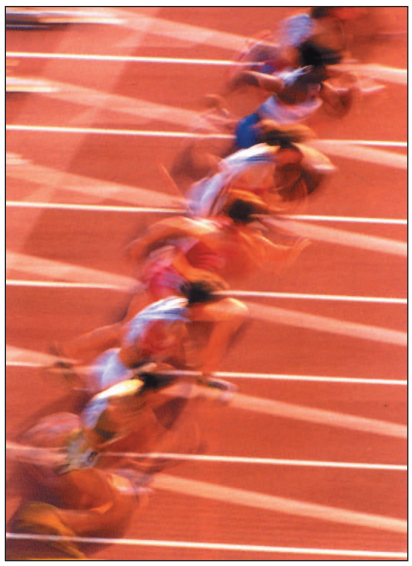
Idén, ahogy előre látható volt, a lap életében is számos változás történik. Tisztázódott, hogy egy évben összesen tizenegy lapszámmal jelenünk meg, mint azt könnyű kitalálni, a január–februári szám oldja meg a látszólagos ellentmondást a hónapok és a lapok száma között. Sajnos, anyagilag is döntéshelyzet elé kerültünk, ugyanis (mint mindenhol) a lapkészítés területén is tovább növekedtek a költségek. Mivel nem szeretnénk, hogy a lap bemosódjon a tömegsajtó sekélyes tengerébe, valamint a szakmai színvonalát sem kívánjuk csökkenteni, ugyanakkor olvasóink véleménye alapján az is fontos, hogy az újságot ne „hígítsuk fel" hirdetésekkel. Így azonban helyzetünk sajnos megkövetelte, hogy a lap árát megemeljük. Az új ár mellett reményeink szerint biztosítani tudjuk, hogy a lap színvonala megmaradjon, sőt, azt is, hogy a szakmai részek mennyisége se csökkenjen. Remélem, hogy a változás ellenére is mindenki örömmel forgatja majd magazinunkat.

Új területek, új ötletek

Sok olyan ötlet merült fel, melyeket szeretnénk megvalósítani. Ezek természetesen további időt, figyelmet és szakértelmet igényelnek. Ahogy az a 67. oldalon látható, keresünk is új munkatársakat, akik részt kívánnak venni csapatunk munkájában.
Érdekes, hogy beszélgetéseim közben sokszor találkozom olyan emberekkel, akik GNU/Linux közelében szeretnének dolgozni, azonban jelenlegi állásuk ezt nem engedi meg, ugyanakkor számos olyan munkáltató akad, aki pedig linuxos szakembereket keres – kevéske sikerrel. Ezért gondoltuk, hogy mind az újságban, mind a hamarosan megújuló weblapon külön rovatot tartunk fenn hirdetések számára, ahol találkozhatnak a linuxőrültek a „felvilágosult" munkaadókkal.

Már megint magyarul!

Mert már egyre több munkahelyen reflektorfénybe kerül a Pingvin. Igaz, van még vele gond is, hiszen a felhasználók által kevéssé ismert, még mindig nagyszámú olyan egyedi terület akad, melyre nehéz vagy lehetetlen tökéletes programot beszerezni és külön pokoljárás a magyar nyelv támogatása, a honosított programok futtatása.
Szerencsére már komoly fejlődés történt



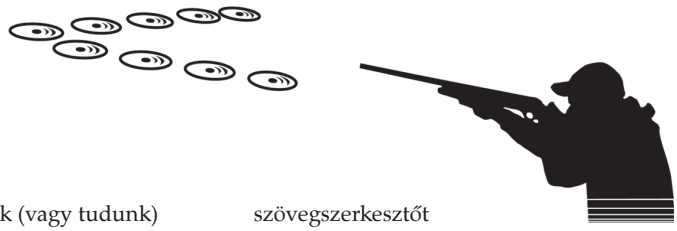
e területen, a KDE és a Gnome is beszéli nyelvünket, valamint több irodai program(csomag) is egyre jobb eredményeket mutat fel e téren. A honosítások szépen folynak, bár nem minden területen zökkenőmentesen. Már megszokhattátok tőlem, hogy elég nyíltan kimondom véleményem, most egy olyan történést mesélek el, amely nekem nagyon kedves. Történt ugyanis, hogy elkezdődött az OpenOffice magyarítása, ezt bizonyára mindnyájan tudjátok, volt támogatás, pályázat, igény, szóval minden, ami szükséges.
A folyamat sajnos mégsem működött megfelelően, ennek például szakmai akadályai is voltak. Olyanokra gondolok, hogy a fordítók nem látták, hogy az általuk kapott szó vagy mondatrész hova kerül (egy gomb, egy menüpont, esetleg egy előugró üzenet?), valamint a fordítás közben a terméknél változatlépés is történt, így aztán teljesen összekuszálódtak a szálak. Mint mindig, most is voltak „szemfülesek", az eredeti terméket például egy cég (egy másik jogszerződés keretében) lefordította, amit azután pénzért árusít.
Felhasználóként ennek örülnünk kell, hiszen már most is lehet (habár pénzért) egy másik lehetőséget felajánlani a főnökségnek. Érdemes azonban hozzátenni, hogy e terméknek a legtöbb cégnél csak addig van létjogosultsága (ahol fontos a pénzkérdés, és/vagy van főállású, GNU/Linuxhoz értő munkatárs stb.), míg meg nem jelenik szabad testvérének magas színvonalú magyarított változata.
És itt kezdődik a gond, ugyanis a csapat hónapok óta nem jutott egyről a kettőre. Hiába a támogatás, hiába a szerződés, hiába a sürgető igény. Az LME néhány lelkes tagja viszont megelégelte ezt az áldatlan helyzetet, és kitalálták, hogy egy szabad hétvégén összeülnek egy fordítási maratonra. Ez a hétvége sajnos a lapzárta utánra esik, így nem tudjuk az eredményt felrakni e szám CD-mellékletére, remélem viszont, hogy a következő számunkban külön foglalkozhatunk a hétvége eredményével. Addig is, előre is, sok sikert és kitartást kívánok nekik!

Az LME háza tája

Mostanában egy kicsit kevesebbet foglalkoztunk a Linux-felhasználók Magyarországi Egyesületével. Pedig az élet ott is zajlik. A már emlegetett pályázatok szép lassan révbe érnek, és újra rendszeressé vált a szerdánkénti egyesületi gyűlés, a pontos adatok a ➲ http://www.lme.linux.hu címen találhatók. Amennyiben valakit részletesebben érdekel, hogy mi minden történik az egyesület berkein belül, érdemes körbenézni a honlapon, esetleg feliratkozni az érdekesebb levelezési listákra.

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

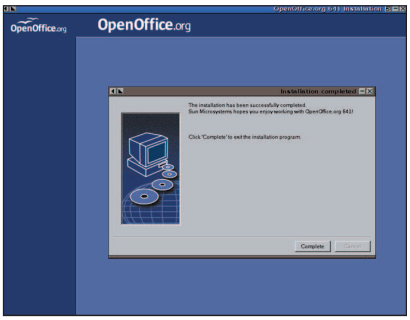
Programvadászat



Várakozással telve kezdjük az új esztendőt. Mint előző számunkban már írtam, szinte az összes programnak idénre várható a megbízható végleges változata. Gőzerővel folyik az irodai csomagok „honosítása”, február első hétvégéjén (e cikk írása után, azonban megjelenése előtt) maratoni OpenOffice-magyarítást tartanak vállalkozó szellemű emberek. Ezúton is szeretném üdvözölni kezdeményezésüket és remélem, hogy ebből végre születik valami használható eredmény, mert az összes eddigi próbálkozás, finoman fogalmazva is „gyengére”sikeredett! A tervek szerint teljesen magyar felületen keresztül fog hozzánk szólni a program, magyar helyesírás-ellenőrzőt is tartalmaz majd, tehát minden olyan dolog adva lesz, amellyel teljes mértékben kiváltható a Microsoft Office-csomagja. Az ár/teljesítményviszony sem elhanyagolható, hiszen majdnem ingyen (azaz alacsony költséggel) hozzájuthatunk egy teljes értékű, az előzőekben említett rendszer formátumaival bíró irodai csomaghoz. Bővebb információ elérhető a ➲ http://oo.geekfinder.hu weboldalon. Nos, akkor nézzük részletesebben a CD-mellékletek tartalmát.

OpenOffice 641c

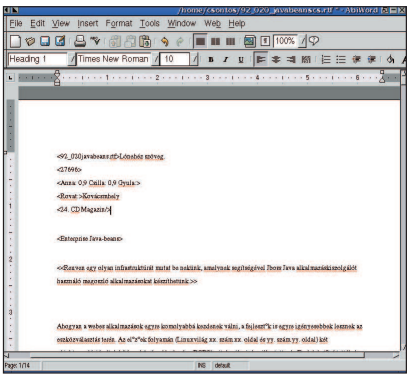
Az előbbiekben már szóltam egy pár szót erről a programról, segítségével, ha a fenti kísérlet sikerrel jár, ingyenes, törvényes irodai munkaállomásokat állíthatunk össze. Az idő bebizonyította, hogy sokan alaptalanul félnek a Linuxra történő áttéréstől, mivel szöveget szerkeszteni szinte minden programban ugyanúgy kell. A nyomtatási ikonra kattintva nyomtathatunk, a lemez ikonra kattintva menthetünk, tehát minden a helyén van. A program legutóbbi változatát találhatják meg a 26. CD Iroda/Openoffice könyvtárban. Attól függően, hogy csak saját magunknak telepítjük, vagy esetleg a rendszer összes felhasználója számára – különböző kapcsolókat kell használnunk. Miután kicsomagoltuk az ***install641C_linux_intel.tar.gz*** fájlt, a könyvtárban lévő `setup` paranccsal indíthatjuk el a telepítési folyamatot. Amennyiben az összes felhasználó számára szeretnénk (vagy tudunk) elérést adni a program használatához, akkor a `setup /net` parancs kiadásával telepítsük rendszerünkre. Ezután minden felhasználó a telepítéskor létrehozott könyvtárban lévő `setup` paranccsal telepítheti fel magának. Ekkor egy úgynevezett munkaállomás-telepítés történik, mely mérete körülbelül 1,5 MB. Abban az esetben, ha ezt a programot többen is használják, akkor mindenképpen erőforráskímélőbb megoldás.



➲ http://www.openoffice.org
➲ http://www.openoffice.hu
➲ http://www.staroffice.hu

Abiword 0.9.6

A másik irodai program az Abiword, szintén visszatérő vendég korongjainkon. Ez leginkább csekély erőforrásigényével és méretével hódíthatja meg a



szíveket és a korosabb számítógépeket. Ugyan nem zökkenőmentesen, de azért tapasztalataim szerint 16 MB memóriával is eldöcög. Természetesen ez a méret nem megy a tudás javára, egy a mindennapi feladatok ellátására képes kényelmesen használható remek kis szövegszerkesztőt kap kézhez vele az ember. Telepítése az előre elkészített rpm, deb, illetve ***tar.gz*** csomagokból lehetséges. Aki kedvet érez hozzá, a könyvtárban fellelhető forrás segítségével le is fordíthatja saját rendszerére. A 26. CD Iroda/Abiword könyvtárában található.
➲ http://www.abisuite.org

Böngészők

A böngészők piaca eléggé színes a Linux oldalán is, mindenki találhat kedvére valót ez lehet a Mozilla, a Netscape, a Konqueror, az Opera vagy bármelyik más, egyéni ízlésünknek megfelelő. Mostani korongunkra a Mozilla, a Netscape és az Opera böngészők legfrissebb változatai kerültek fel. A Netscape és Mozilla használóinak mindenképpen érdemes frissítenie, mivel eddig hibásan kezelték a sütiket (cookie).

Mozilla 0.9.7

A program a Web/Mozilla könyvtárban található többféle formátumban. A Red Hat 7.x-es rendszerekhez teljes telepítő-

készlet áll rendelkezésre, míg a többi Linux-változatra a ***tar.gz*** végződésű fájlokban található telepítőprogramokkal telepíthetjük.
➲ http://www.mozilla.org

Netscape 6.2.1

Mint már fentebb említettem, mindenkinek ajánlom a Netscape 6-os sorozat frissítését a korongunkon megjelenő 6.2.1-es változatra, ugyanis ebben már kijavították azt a „romlott sütikezelést”, amelynek segítségével illetéktelen kezekbe kerülhettek személyes adatok is.
➲ http://www.netscape.com

© Kiskapu Kft. Minden jog fenntartva

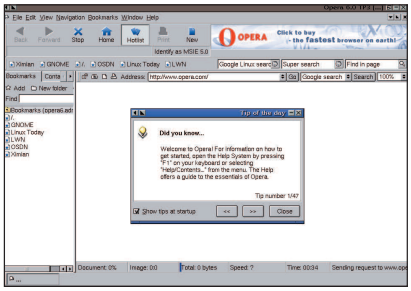
© Kiskapu Kft. Minden jog fenntartva

Netscape 4.79

Ez a régi és egyre elavultabb böngésző is tartja még magát a fronton.
A 26. CD Web/Netscape/4.x könyvtárban megtalálható a Communicator (a teljes Netscape-változat) és a Navigator (csak a böngésző). Ezek méretbeli különbsége lehet leginkább az az ok, ami miatt az egyszerű böngésző telepítése mellett döntenek a felhasználók.
➲ http://www.netscape.com

Opera 6 TP3

Az Opera fejlesztői is belehúztak a munkába, s immár a TP3-as változat



érhető el weboldalukon. Vigyázat ez még nem a megbízható változat!
A korongunkon deb, rpm és tgz formátumban is megtalálható.
➲ http://www.opera.com

Rendszermag

A Linux rendszermagjának fejlesztése gőzerővel folyik, az Interneten nagyon sok helyen azt olvashatjuk, hogy a 2.4-es sorozat még nem eléggé kiforrott a kiszolgálók piacának meghódítására, így ezeken a helyeken azt javasolják, használjuk inkább a 2.2-es sorozat legutóbbi tagjait. Nos, természetesen mindenkinek saját döntésére van bízva, melyiket választja. Jelenlegi rendszermag-válogatásunkban megtalálható a 2.0.39-es is, bár valószínűleg ez inkább már csak emlék a legtöbb linuxos számára, azonban biztosan vannak még olyan rendszerek, amelyek ilyen rendszermagot használnak.

2.2.20

Ez a legújabb a 2.2-es sorozatból, melyben sok újdonság található a 2.4-es sorozatból is.

2.4.17

A rendszermag fő fejlesztési vonalának terméke, számos hibajavítással és kiegészítéssel.

2.5.2

A fejlesztőknek szánt változat, mindenki csak saját felelősségére használja!
➲ http://www.kernel.org

Xfree86 4.2.0

Megérkezett a régóta várt, de késésben lévő XFree grafikus felület legújabb változata. Legfontosabb változásaira legalább érintőlegesen érdemes kitérni.

- ***Meghajtó programok***
  - támogatottak a régebbi s3-as lapkakészlettel rendelkező grafikus kártyák is,
  - a vmware-meghajtók hibajavítása és a VMWare 3.0-s változat jobb támogatása,
  - támogatás a Trident *BladeXP lapkakészlethez,
  - Xv-támogatás a Trident TGUI lapkákhoz,
  - újra támogatott a régi Trident ISA/VLBus-változatok,
  - támogatás a glint-meghajtóban a 3DLabs Permedia4, GLINT R4 és Gamma 2 lapkákhoz,
  - a i810-meghajtó kiegészítése az i830-as lapkákhoz is (csak Linuxon lett kipróbálva),
  - kiegészített ATI radeon-meghajtó a Radeon 7500 (2D és 3D), a Radeon 8500 (csak 2D) és a Rage128ProII kártyákhoz,
  - Matrox G550-támogatás,
  - különféle nVidia meghajtó-kiegészítések,
  - az fbdev-meghajtó támogatja a forgatást,
  - különféle frissítések apm-, ark-, chips- (C&T), cirrus-, i128-, neomagic-, newport-, s3virge-, siliconmotion-, sis-, tdfx-, tseng-, vesa-, és vga-meghajtókban.
- ***Beviteli eszközök***
  - az egérmeghajtó támogatja az újra csatlakoztatást a PS/2 egereknél Linux alatt,
  - Linux USB billentyűzetelérés működik, ha nincs PS/2-vezérlő.
- ***X-kiszolgáló és kiegészítés-frissítések***
  - újra összehangolás az X.Org X11R6.6-tal,
  - Mesa-frissítés post-3.4.2-re,
  - DRI-meghajtók összehangolása a legutóbbi DRI-forrásokkal,
  - különféle frissítések az ***Xft*** könyvtárakban,
  - a PEX- és XIE-kiegészítések többé nem az alapértelmezett rendszer részei.
- ***Ügyfél- és könyvtárfrissítések***
  - FreeType2 2.0.6-os változat,
  - libGL súgóoldalak,
  - Xterm-frissítés patch level 165-re,
  - eltávolították a SuperProbe-t.
- ***I18N és betűkészlet-frissítések***
  Új Luxi betűk (TrueType és Type1) a Bigelow és Holmes cégtől. Ezeket a betűket ***Kris Holmes*** és ***Charles Bigelow*** tervezte.
  - még több nemzetközi billentyűzet-kiosztás,
  - modulrendszerű I18N-támogatás az ***Xlib***-ben az X11R6.6-ból,
  - a fontenc-réteg frissítése és a ***fontenc*** könyvtár más alkalmazások számára is elérhetővé vált.

Telepítésének a 27. CD Xfree86/Linux-ix86-glibc22 könyvtárában lévő ***Xinstall.sh*** fájl elindításával kezdhetünk neki. Ezt a parancsot természetesen rendszergazdaként futtassuk. A telepítő végigvezet bennünket az egész folyamaton, miközben különböző kérdésekre kell válaszolnunk, sajnos ezek a kérdések nem a legegyszerűbbek, így ezt a telepítést össze sem lehet hasonlítani az egyes Linux-változatok grafikus telepítő- és beállítóprogramjaival.
➲ http://www.xfree86.org

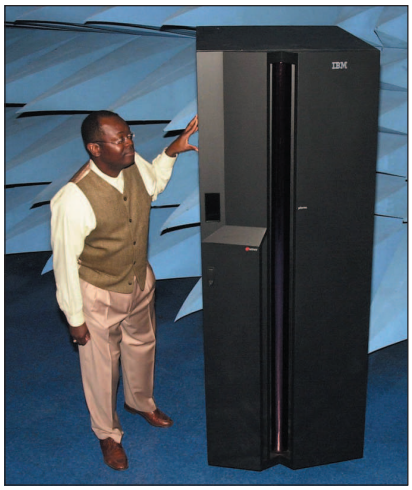
Red Hat-frissítés

A 27. CD Frissítések könyvtárban a Red Hat 7.2-es kiadáshoz megjelent összes frissítés fellelhető. Ezek között található biztonsági hibajavítás és programok új változatára történő frissítése egyaránt.
Kellemes felfrissülést!
➲ http://www.redhat.com

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

IBM szuperszámítógép az Országos Meterológiai Szolgálatnál

Az Országos Meterológiai Szolgálat (OMSZ) IBM p690-es szuperszámítógép szállításáról szóló megállapodást írt alá az IBM Magyarország Kft.-vel. A Meterológiai Szolgálat a berendezést számszerű időjárás-előrejelző modelljének fejlesztésére, futtatására, továbbá alapkutatásokra fogja használni. Ez a rendszer látja el többek között az ország intézményeit,



hírcsatornáit és repülésirányítási rendszereit meteorológiai adatokkal. Az IBM eServer p690 többéves fejlesztés eredményeként jött létre, és olyan újdonságokat vonultat fel, mint például a nemrégiben bejelentett POWER4 mikroprocesszor, illetve a nagygépes technológiából átvett, magas rendelkezésre állást biztosító önfelügyeleti és önjavító eljárások. A gigahertzes tartományban működő IBM POWER4 mikroprocesszor, amely elismerten legalább egy nemzedéknyivel a versenytársak lapkái előtt jár, az első olyan chip a világon, mely két processzort tartalmaz. A csúcstechnológiák használatának köszönhetően lehetővé válik az eServer p690 energiatakarékos üzemeltetése, miközben nagyobb teljesítményt nyújt, mint a kétszer annyi vagy még több processzort tartalmazó hagyományos kiszolgálók. A memóriagyorsítótár és a processzor közötti adatáramlás közel 125 GB/s sebességgel folyik, ekkora sebességgel 25 teljes hosszúságú DVD-lemez tartalmát lehetne továbbítani egyetlen másodperc alatt. A 32-utas szuperszámítógép rendkívül nagy számítási teljesítőképességgel bír a Meterológiai Szolgálat számára, mely hosszú távon kielégíti az időjárási és éghajlati modellezés, a kutatás és fejlesztés, továbbá a futtatási területen jelentkező számítási igényeit. Az OMSZ IBM eServer p690-e egyben Magyarország leggyorsabb szuperszámítógépe, mintegy ötven százalékkal nagyobb számítási teljesítményt nyújt, mint az országban jelenleg üzemelő leggyorsabb szuperszámítógép.
Világszerte számos meteorológiaiszolgálat és kutatási központ használ IBM-szuperszámítógépeket, így például az egyesült államokbeli Nemzeti Atmoszférakutató Központ, valamint az Európai Középtávú Időjárás-előrejelző Központ is – a Magyar Köztársaság is társult tagja –, mely adatokat szolgáltat az OrszágosMeteorológiai Szolgálat részére.
➲ http://www.ibm.hu

HP

A HP ingyenesen hozzáférhetővé tette a HP Application Server 8.0 teljes változatát.
Az új nemzedékbeli HP Application Server 8.0 (alkalmazáskiszolgáló (HP-AS)) piaci bevezetésével egyidejűleg a Hewlett Packard bejelentette, hogy mind a fejlesztők, mind a végfelhasználók számára ingyenesen hozzáférhetővé teszi új alkalmazáskiszolgálóját. A vállalat a rendkívül kedvező áron, processzoronként 5500 dollárért megvásárolható HP Application Server 8.0 Resilient Edition piaci bevezetéséről is beszámolt. „Az egyszerű, alapszintű alkalmazáskiszolgálók hamarosan az operációs rendszer szerves részévé válnak, s a J2EE (Java 2 Enterprise Edition) előírásainak megfelelő szabványos termékké növik ki magukat" – állapította meg nemrégiben a Gartner Group elemzője. Az informatikai szállítók közül elsőként a HP szorgalmazta a módszer széles körű elterjesztését, hogy felgyorsítsa a webes szolgáltatások (Web Services) bevezetését. Az ingyenes HP Application Server (HP-AS) teljes szolgáltatással rendelkező, méretezhető J2EE alkalmazáskiszolgáló, amely különösen a működés legfontosabb és legérzékenyebb megoldások fejlesztése és bevezetése terén használható sikeresen. Most hogy a középréteg (middleware) ingyenessé vált, a fejlesztők jóval alacsonyabb összköltséggel tervezhetik meg, fejleszthetik és vezethetik be HP-S-alapú megoldásaikat. A HP-AS alapját képező harmadik nemzedékbeli alapszolgáltatás-szerkezet (Core Services Framework) már önmagában is izgalmas újításnak számít.
A HP Application Server ingyenesen letölthető a HP weboldaláról.
➲ http://www.hpmiddleware.com/hp-as/
➲ http://www.hp.hu

Compaq Evo D500

A Compaq új ultravékony Evo D500 asztali gépe nyerte el a legjobb számítógép díját a CES legjobbjait jutalmazó 2002. évi díjkiosztáson. A nyerteseket a Tech tévék értékelését végző versenybíróság választotta ki újszerű megjele-

nése, hasznossága és megfizethető ára alapján. Ugyanebben a tekintélyes osztályban a Compaq Presario 8000 asztali gépe is az első öt között szerepelt.
➲ http://www.compaq.hu
➲ http://www.compaq.com
➲ http://www.compaq.com/products/desktops/d500usd/subfamily.html

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Felemás események a jogi fronton

Két egymástól független bírósági fejleményről értesültünk – az egyik a DVD-vel kapcsolatos, a másik a Microsoftot érinti. Az előbbi esetében a kaliforniai fellebbviteli bíróság egyhangúlag megsemmisített egy korábbi bírósági végzést, amely megtiltotta a DeCSS-kód nyilvánossá tételét. Ez ugyanis a DVD-k dekódolásával lehetővé tenné, hogy bármilyen számítógépen le lehessen őket játszani, ne csak a fogyasztóielektronikai kartell tagjai által gyártott DVD-lejátszókon. Az eset valójában az egyike két DVD-vel kapcsolatos ügynek.
A másik, a Universal Studios és az Amerikai Egyesült Államok kontra Corley jelenleg az államok másodfokú fellebbviteli bírósága (US 2nd Circuit Court of Appeals) előtt van. Az eredeti New York-i tárgyaláson a 2600 Magazine kiadójától megvonták a DVD forráskódjának, s egyéb DVD-vel kapcsolatos adatok közzétételének a jogát – egy olyan bíró vezetésével, aki korábban a Time Warner alkalmazásában állt.
A Microsoft ügyében az Egyesült Államok és a Microsoft előzetes megállapodásra jutott, amely valószínűleg véget vet hosszantartó jogi csatározásaiknak.
A DeCSS-ügyben a kódot közzétevők alkotmányos szabadságjogai és a DVD-t terjesztő szórakoztatóipari vállalatok üzleti titkainak joga csapott össze. Ezek a vállalatok kimondottan azt ellenezték, hogy olyan DeCSS-program jelent meg a Világhálón, amelyet 1999 őszén azzal a céllal írtak, hogy Linux alatt is létezzen DVD-lejátszó. 2000 elején a DVDCCA, a filmstúdiók DVD-engedélyezési szervezete pert indított több száz programozó és webhelytulajdonos ellen a DeCSS közzétételének betiltásáért. A Santa Clara megyei *William Elfving* bíró 2000. január 21-én helyt adott a betiltás iránti kérésnek. A fellebbviteli bíróság azonban kimondta, hogy Elfving megsértette *Andrew Brunner*-nek, az egyik alperesnek a First Ammendment által szavatolt alkotmányos szabadságjogait, amikor arra kötelezte őt, hogy távolítsa el a kódot a saját weboldaláról (a First Ammendment az amerikai Alkotmány szabadságjogokat tartalmazó fejezete).
Az alsóbb szintű bíróság az üzleti titok jogtalan elsajátítására alapozta a döntését, holott Brunner egy szerzői jogilag nem védett területen (a Slashdot portálján) találta a programot, és pusztán újból közreadta. Az ügy várhatóan idén visszakerül Elfving bíró elé.
Az Egyesült Államok kontra Microsoft ügyet illetően a szakértők többsége egyetért abban, hogy a megállapodás előnyben részesíti a Microsoftot, amit az ügy eredeti bírája, *Thomas Penfield Jackson* előzőleg felosztásra ítélt. *Dan Gillmor*, a San Jose Mercury News újságírója ezt írta:
„Ez az alku, már ha foganatosítják, felér egy szerelmi vallomással a legarrogánsabb és legmegátalkodottabb monopolista felé, ami a Standard Oil óta létrejött. Felhívás a harácsolás folytatására, és a szabad verseny lábbal tiprására korunk legfontosabb piacán, az informatikai piacon."
A CNet-es *John Borland* szerint ez az egyezség „jutalom, és nem jogorvoslat". A Userland munkatársa, *Dave Winer* független médiareklám-fejlesztő, aki dolgozott a Microsofttal a SOAP (lásd Linuxvilág áprilisi. szám 70. oldal) és az XML-RPC protokollok kapcsán, így értelmezte a megegyezést:
„Nyilvánvalóan nem tisztázza eléggé a helyzetet, ráadásul bizalmas és nem éppen célravezető módon vonja be a kormányt a Microsoft operációs rendszerének és hálózati szolgáltatásainak (mint például a SOAP) felépítésébe. Noha a megegyezés nem sérti a vállalat egységét, a Microsoftnak fel kellene hagynia azzal a gyakorlattal, hogy eredetialkatrész-gyártóit (OEM) a Windows terjesztésére kényszeríti. Ezenkívül operációs rendszereinek egyes elemeit is elérhetővé kellene tennie, hogy a versenytársaknak egy kicsivel nagyobb esélyük legyen ténylegesen versengeni a Microsoft saját alkalmazásaival."
A „bizalmas és nem éppen célravezető" megjegyzés egy „szakmai bizottságra" utal, melynek tagjait a Microsoft és a Bíróság együttesen nevezné ki, és a Microsofthoz kihelyezve felügyelné a megállapodás teljesítését.
Az egyetlen nyilvánvaló jó hír a Linux számára az alkatrészgyártók felmentése a Microsoft OEM-szerződések alól, amely előírta a Windows terjesztését. Így – legalábbis átvitt értelemben – sokkal nyitottabb alkatrészcsatorna állna rendelkezésre a Linux terjesztéséhez.

Doc Searls

Linux és más operációs rendszerek

Biztosan mindenki gondolt már arra, hogy milyen jó is lenne, ha bizonyos programokat, melyeket bár nem Linuxra írtak, Linux alatt is tudna futtatni. Ez legtöbbször, ugyanis, azt hiszem a Windows operációs rendszerek programjaira vonatkozik, ezt a környezetet ismerik a legtöbben. A Linux azonban nem elégszik meg ennyivel, számomra a legérdekesebb a Macintosh-környezet megteremtése X86-os processzorokon. Az általam legjobbnak ítélt program erre a feladatra a Basillisk II program. Ennek segítségével megteremthetjük programjaink számára az eredeti Mac 68 K környezetet.
A programot letölthetjük a hivatalos honlapjáról a ➲ http://www.uni-mainz.de/~bauec002/B2Main.html címről.
A telepítés után szükségünk lesz egy eredeti Mac ROM-ra, ehhez azonban rendelkeznünk kell egy eredeti Macintosh számítógéppel, melyből ki tudjuk nyerni. Miután ezzel elkészültünk, indulhat a MacOS 7.5 telepítése, melyhez az operációs rendszert ingyenesen letölthetjük az Apple honlapjáról.

Ők mondták

Miközben a Der Spiegelt fordítottam angolra a Babelfish segítségével, felfedeztem, hogy a „lépfene" angol megfelelője a „léptűz". *(Anita French)*

Elképzeltem, milyen lenne, ha a programokat visszafelé fejlesztenék – az ötlet nem is rossz. Így az 1.0-s változat egy halom látványos szolgáltatást tartalmazna. A későbbi változatok egyre kevesebb szolgáltatást nyújtanának, míg végül egy szép napon mind letisztulna, és már csak azok a változatok maradnának, amelyek igazi bombázóvá teszik az alkalmazást. *(Brent Simmons)*

Nemrég… rájöttünk, hogy a teljes tulajdonosi költséget tekintve – beleértve a számítógépek, az alkatrészek, az alkalmazások, a munkaerő, a beszerzések és a nyugdíjazás költségeit is – a Unixnál lényegesen kevesebb költséggel jár, ha a hálókiszolgálásra és a hasonló feladatokra három éven át Linuxot használunk. *(Dan Kusnetzky, International Data Corp. – IDC)*

Az a tanító, aki összhangban van azokkal, akiket tanít, eggyé válik velük, és többet tanul tőlük, mint amennyit ő tanít nekik. Aki semmit sem tanul a tanítványaitól, az véleményem szerint maga is értéktelen. Én mindig tanulok attól, akivel beszélgetek. Többet kapok tőle, mint amennyit én adok neki. Ily módon az igazi tanító a diákjai diákjának tartja magát. Ha ilyen hozzáállással tanítod a növendékeidet, az nagymértékben a javadra válik. *(Ghandi)*

Mindegy, mivel foglalkozik, minden vállalat az útjában áll valakinek. *(Deborah Branscum)*

A Microsoft az összes szórakoztatótechnikai ágazatra pályázik. Annyi oldalról támad, hogy könnyen célt téveszt. *(David Strom)*

Nincsen törvény arra, hogy egy vállalatnak feltétlenül fenn kell maradnia. Olyan törvény azonban van, hogy minden, amit az ember alkot, mulandó. Ritka az a vállalkozás, amely több mint 25 éven át sikeres tud maradni. Az örökkévaló vállalat gondolata Wall Street-i téveszme. Az Internet fő hatása nem gazdasági, hanem pszichológiai. Nincsen új gazdaság. Az Internet mindössze a régi gazdaságot terjesztette ki nagymértékben. *(Peter Drucker)*

A pénzügyi szakértők sosem fogják megérteni az üzlet lényegét, mert meggyőződésük, hogy a vállalatok pénzt csinálnak. A vállalatok azonban például cipőt készítenek, és ezt a pénzügyi vátészek nem fogják fel. Ők azt hiszik, hogy a pénz a valódi, pedig a cipők a valódiak. A pénz csak a végeredmény. *(Peter Drucker)*

Habár tagadhatatlan, hogy nyílt forráskódú programot írni nem akkora dicsőség, mint életeket menteni egy égő házból, azonban azt is el kell ismerni, hogy nem felforgató tevékenység. Sokkal inkább önzetlenség – olyasvalami, amit Amerikában mindig szívesen fogadnak. *(Russell Pavlicek)*

A vezetők olyan látnokok, akikben csak csekély mértékben fejlődött ki félelemérzet, és fogalmuk sincs az előttük álló nehézségekről. *(Dr. Robert Jarvik)*

Manapság az ROI-számítások igen nagy fontosságúak, és a nyílt forráskód az egyetlen üdvözítő módja annak, hogy olyan költséggel készítsenek igényes alkalmazásokat, amelynek a megtérülési ideje elfogadható. *(David A. E. Wall – idézet egy Yozuns-vázlatból)*

A szellemi tulajdon (intellectual property, IP) már vagy ötmillió éve irányítja az emberi fajt. Az utóbbi száz évben egyre nagyobb mértékben szipolyozták ki a közvetítők és a paraziták, akik enélkül egy sörösdoboz kinyitásához sem értenének. *(Tom Matrullo)*

Nincs süti

Az elmúlt év elején *Don Marti* továbbított nekünk egy ritka rejtélyes elektronikus levelet, amit az egyik sajtóosztály alkalmazottja írt. Marti levelének címe ez volt: „Rossz sajtófiú! Nem kap jutalomfalatot!" Kis híján megpukkadtam a nevetéstől, mivel – bár szégyellem bevallani – magam is jelentős időt töltöttem sajtóügyintézőként. Azóta az RSNKJF bevett kifejezéssé vált a Linux Journal berkeiben. Hadd adjak közre néhány idézetet a levélből:

- „Ha beszélni szeretnél a vállalat vezetőivel a jövőbeli terveiről és arról, hogy a Linux 7.1 milyen hatással van az Internetre, a hálózati környezetekre és az információtechnológia világára…"
- „A nem alkalmaz hátrányos megkülönböztetést a Linux operációs rendszert használó számítógépekkel szemben sem. A nyílt szerkezetű, ami azt jelenti, hogy a Linux-felhasználók a saját hálózati igényeikhez szabhatják, ha alkalmazásaikat a programjára telepítik. "
- „Igazán klassz volt a 70-es években élni. Akkoriban az Earth, Wind and Fire, a hippiszerelés és a kommunák voltak divatban. Harminc évvel később, 2001-ben a 70-es évek még mindig menőnek számítanak. Úgyhogy ne dobd el a régi, földet söprő farmert, a nyakpántokat és a nagyszámítógépeket – a régi most a menő, csak „retrónak" kell hívni. A segítségével a régi technológia visszaillleszthető napjaink legkorszerűbb felületére."

Az utolsó meg sem említi, mi is az „A" vagy mire való. Ennek ellenére azt kell hinnünk, hogy valóban a Linux a legkorszerűbb felület.

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

© Kiskapu Kft. Minden jog fenntartva

Iciri-piciri...

„A világ legkisebb PC-je” – állítja az EZgo nevű apró számítógépről gyártója, az Atoz. Műszaki és filozófiai fejtegetésekbe bocsátkozhatnánk arról, hogy hol is kezdődik a PC, illetve mi számít a legkisebbnek, a tények viszont tények maradnak: az EZgo súlya 1 kg alatti, méretét tekintve pedig valóban tenyérnyi.
Megpróbálhatunk belekötni a PC fogalmába az elérhető szolgáltatások oldaláról is, ám ez esetben nehéz dolgunk lesz, hiszen az EZgo semmiben sem nyújt kevesebbet, mint asztali társai. A fejlesztők ugyan nem kerülhették el, hogy a hordozható gépekből megismert alkatrészekből építkezzenek, ám a gépbe például asztali lapkakészlet került, nem pedig valamilyen különlegesség, amit a Windows kivételével a legtöbb operációs rendszer csak félig-meddig, vagy semennyire sem támogat.

Előnyök

- Csendes működés
- Teljes értékű gép, lebutított vagy különleges összetevők nélkül

Hátrányok

- Kicsit magas ár

Belső szervek

Soroljuk fel gyorsan, mit is találunk a tesztgép belsejében, ha felnyitnánk (azonban ne nyissuk fel, mert garanciavesztést okoz, csupán játsszunk el a gondolattal)!

- Intel 810E lapkakészlet,
- Intel Celeron vagy Pentium III processzor, a tesztgépben Celeron 800 MHz órajellel,
- mobil merevlemez (cserélhető),
- SDRAM memória (cserélhető, legfeljebb 256 MB méretig),
- CD/CR-RW/DVD-meghajtó (cserélhető).

A gép tudása ezzel még messze nem merült ki, hiszen az alaplapra szinte minden létező és elvárható csatlakozót ráépítettek, kivezetéseiket a gép oldalán találhatjuk meg:

- 10/100 Mbit sebességű ethernetcsatoló,
- 56 KB/s sebességű modem,
- hangkártya-, hangszóró- és fejhallgató-kimenet, valamint mikrofonbemenet,
- infravörös kapu,
- két USB-kapu,
- soros, párhuzamos kapu,
- S-Video és kompozit videokimenet,
- PS/2-es billentyűzet- és egércsatlakozó.

Hajlékonylemezes meghajtó alapállapotban nincs a gépben – az apró USB-s memóriák és a rendszerindításra is alkalmas CD-k megjelenése óta ez talán nem is nagy veszteség, főleg ha CD-újraíróval vesszük meg a gépet. Amennyiben mégsem tudjuk nélkülözni, külön kiegészítőként vehetünk hozzá párhuzamos kapura csatlakozó meghajtót.
Elsőként talán a 810-es lapkakészlet miatt kapjuk fel a fejünket. Ennek megbízhatatlansága megjelenése után legendássá vált, részben talán azért, mert sokan az olcsóbb árfekvésű 810-es alaplapokkal és számítógépekkel találkoztak. Lehetséges azonban 810-es lapkakészlettel is megbízható gépet építeni, főleg ha a kipofozott, az újabb processzorokat is támogató változatot vesszük elő. Ebben az esetben is ez történt, a gép aprócska alaplapja az FCPGA-tokozású Celeron és a Pentium III processzorokat is támogatja, megbízhatóságával pedig tapasztalataim szerint semmi gond nincs.
A mobil merevlemez és a mobil CD-meghajtó alkalmazása a kis méret következtében kevéssé meglepő. Érdekes, hogy maga a ház is a hordozható gépek vázára emlékeztető anyagból épült. Szintén a mobilgépek öröksége lehet, hogy hatékony hűtési megoldást kellett keresni, hiszen a gépbe a hordozható gépekkel ellentétben asztali, így nagyobb hőtermelésű processzor került, ám a méretbeli korlátozások miatt nem helyezhetünk rá nagyméretű hűtőbordát. A tervezők úgy lettek úrrá a gondon, hogy a hőt a processzortól egyrészt egy rézlappal vezetik el, másrészt a levegőt egy nagyobb és egy apró, oldalsó ventilátor segítségével gondosan kitalált útvonalon áramoltatják a gép belsejében. Az eredmény meggyőző: a 800-as processzort még nagyobb terhelés mellett sem lehet túlhevíteni, a gépház kívülről tapintva kellemes üzemi hőmérsékletre melegedett ugyan, de forrósodásról nem beszélhetünk.

A Linux támogatja az EZgót

Amikor kíváncsian kiemeltem apró dobozából a készüléket, nem foglalkoztam a leírás tanulmányozásával, hanem gyorsan kerítettem egy monitort, egy billentyűzetet és egy egeret, csatlakoztattam a tápegységet, és már indítottam is a rendszert – egyelőre csak a Windows 98 telepítő CD-jéről. Gondolva a Linuxra, a merevlemezt kettéosztottam, majd számítva arra, hogy meg kell küzdenem a különféle beépített kiegészítők támogatásával, feltelepítettem a Windows ME-t. Egészen meglepődtem, amikor kiderült, hogy nincs szükség a géphez mellékelt CD-re, valamint a rajta található illesztőprogramokra, ugyanis a rendszer minden összetevőt felismert és kezelt.
Ezekután némileg megnyugodva helyeztem be a Red Hat Linux 7.1-es változatának telepítőlemezét, és ekkor sem csalódtam. A telepítés gond nélkül váltott grafikus módba, és a rendszer első indítása után minden kiegészítő azonnal működött. Rendelkeztem hálózati kapcsolattal, a hangkiszolgáló engedélyezése után MP3-mat tudtam hallgatni, a gép tehát készen állt a munkára.
Ó igen, az MP3! A gépben helyet szorítottak egy apró hangszórónak. Ez nemcsak arra hivatott, hogy a bekapcsoláskor pityegjen egyet, majd dolga végeztével pihenjen, hanem zenehallgatásra is használható. Más kérdés, hogy méretéből fakadóan nem képes egy zsúfoltabb iroda zaját túlüvölteni. Ne számítsunk elképesztő hangminőségre ehhez vegyünk inkább hifitornyot, de az alapvető figyelmeztetések, hangjelzések biztosan el fognak jutni hozzánk. Ha a készülék

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

CD-meghajtójának ajtaját magunk felé fordítjuk, egy apró hangerőszabályzó is kézre esik, amivel szükség szerint a hangszórót is hangosítani-halkítani tudjuk külön program igénybe vétele nélkül.
Jogos kérdés lenne, miért ne működne az EZgo minden Linux alatt is, hiszen a Linuxot is támogatja. Nem mehetek el szó nélkül a dolog mellett: a gyártó a Windowsok mellett ugyan támogatott operációs rendszerként a Linuxot is feltüntette, de tényleges támogatást nem biztosít hozzá. A honlapján nem találunk semmilyen Linuxszal kapcsolatos letöltést, leírást, a mellékelt korongon is csak windowsos illesztőprogramokra bukkanhatunk – ha rendszermagfoltokat, segédprogramokat nem is gyűjtögettek a Linuxhoz, legalább egy telepítéssel kapcsolatos *olvass.el* fájl felkerülhetett volna a lemezre… Sokkal inkább arról van szó, hogy a Linux támogatja a miniatűr gépet, és nem fordítva.

Mit is lehet kezdeni egy ilyen géppel?

A hordozható gépekkel ellentétben nincs saját billentyűzete, külön egeret és monitort kell hozzá szerezni, valamint akkumulátor híján tápegységet is vinnünk kell, bárhova is indulunk vele. Bővíthetősége rendkívül korlátozott, gyakorlatilag felnyitni sem érdemes, hiszen a bővítőkártyáknak nem jut hely benne. Bemutatók tartására, elromlott gépek gyors pótlására, adatmentésre azonban ideális megoldás lehet, és „bővíthetetlenségét” egyben lényeges előnyének is tartom, hiszen ha nem nyúlkálunk folyton a gépbe, el sem ronthatjuk. Állandó munkagépnek is megfelelő, kis helyigényének köszönhetően pedig az elhelyezésével sem akadhat gondunk. Nem csap zajt sem, a két hűtőventilátor állandó susogása mellett legfeljebb a merevlemez halk kerregése töri meg a szoba csendjét – aludni is nyugodtan lehet mellette.
Fel kell készülnünk arra is, hogy néhány tényező miatt a gép teljesítménye kisebb lehet. A mobil merevlemezek teljesítménye a kisebb fordulatszám végett mindig alacsonyabb, mint az asztali példányoké, ami az operációs rendszer betöltésekor, az alkalmazások indításakor, valamint a lapozófájl használatakor egyaránt érződhet. A CD-meghajtó sebességét határozottan jónak találtam, 3D-játékok futtatására viszont más összeállítást válasszunk: a 3Dmark2001 még 500 pontot sem osztott ki a beépített VGA-vezérlőnek.

Az apró számítógépet autós tápegységgel is működtethetjük, valamint kisméretű billentyűzetet magyar nyelvű kivitelben is rendelhetünk hozzá.

Adatok

CD/CR-RW/DVD-meghajtó (cserélhető);
processzor: Intel Celeron 800 MHz CPU;
memória: 128 MB SDRAM (cserélhető, legfeljebb 256 MB méretig);
merevlemez: 10 GB merevlemez (cserélhető);
képernyő: nincs hozzá
végfelhasználói ár: kb. 190 000 Ft + áfa
További tájékoztatást a Sved Kft. nyújt.
telefon: 469-8000
e-mail: info@sved.hu
➲ http://www.atoz-egzo.com.tw
➲ http://www.sved.hu

Medgyesi Zoltán (mzx@axelero.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.

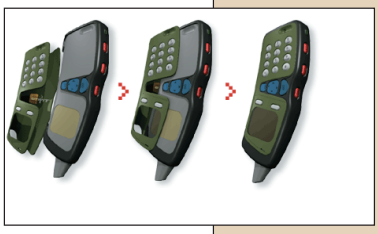
© Kiskapu Kft. Minden jog fenntartva

A Linux legyen veled, bármerre jársz is!

Rick néhány érdekes bejelentéssel foglalkozik – olyan játékszerekkel, amelyekben beágyazott Linuxot találunk.

A GITWiT (Kirkland, Washington) jelenleg egy Linux-alapú, vezeték nélküli kézikészüléket fejleszt, amely nemcsak egyesíti a távközlési és szórakoztatási lehetőségeket, de tervezése egyedülálló mértékben a testreszabhatóságot is lehetővé teszi. A cég színes, elsősorban a fiatalok igényeihez szabott szolgáltatásokat nyújtó telefonjával eleinte a tehetősebb tizenévesek rétegét szeretné megcélozni.

A GITWiT piackutatási eredményei szerint az Egyesült Államokban élő negyvenmillió tizenévesnek jelenleg körülbelül csak a negyede rendelkezik mobiltelefonnal, így további harminc-millió lehetséges vevőre vethetik ki a hálójukat.



A GITWiT két részből épül fel

Hogyan és miért: a két részből álló telefon

Amint a fényképen is látható, a GITWiT telefonja fizikailag két részből áll: a GITWiT-kialakítás szerinti készülék nyújtja a telefonos szolgáltatásokat; a programokat tartalmazó fedél, amelyet Smart Skinnek (intelligens bőr) neveznek, a központi készülékre illeszthető rá, így a készülék egyedi és könnyen módosítható kinézettel és szolgáltatásokkal ruházható fel.

Az IBM/Citizen WatchPad

Minden Smart Skin egy Smart Keyt, azaz intelligens kulcsot tartalmaz, amely az ipari szabványként elterjedt biztonságos intelligens kártyák műszaki megoldásaira épül. Az intelligens kulcsok rejtjelezett adatokat és programokat tartalmaznak, amelyek a bőr témájának megfelelően módosítják a készülék szolgáltatásait. Ha a felhasználó például a kedvenc együttesének megfelelő témájú bőrt húz készülékére, akkor egy csapásra új megjelenésű és hangulatú felhasználói felülettel, valamint egyedi csengőhangokkal, grafikákkal, előre beállított, vezeték nélküli kapcsolatokhoz készített könyvjelzőkkel szerelheti fel a telefonját, és azonnal elő is fizethet a csapat legújabb CD-jére vagy soron következő koncertjére.

A GITWiT várakozásai szerint fejlesztésük népszerű lesz a mobiltelefon-szolgáltatók körében is, hiszen használatával lehetővé válik a gyorsan változó divat és a szórakoztatóipari újdonságok követése – a drága központi részt ugyanis nem kell lecserélni, elég egy viszonylag olcsó intelligens bőrt piacra dobni.

Mit rejt a felszín?

Műszaki szempontból a GITWiT-készülékek három fő részből állnak: egy mobiltelefon-processzorból, a saját fejlesztésű Smart Key operációs rendszerből és egy beágyazott központi processzorból.

A központi processzor valójában egy Linuxot futtató beágyazott számítógép, ez a telefon szíve. ARM7 lapkára épül, és több beágyazott kiegészítőt, valamint be és kiviteli felületet is tartalmaz, amelyek a billentyűzettel, a színes LCD-kijelzővel és a bőrökkel tartják a kapcsolatot. A vezeték nélküli, mobiltelefonos távközlési feladatokat külön mobiltelefon-processzor látja el.

A GITWiT 2.4.5-ös változatú Linux-rendszermagot használ, amelyet *Russell King* és *Nicolas Pitre* írt, valamint a cég saját készítésű ARM-foltjaival látták el. A rendszer grafikus felületéhez bizonyos részeket a Microwindowsból, valamint a BusyBoxból vettek át.

Miért pont Linux?

„A Linuxot számos ok miatt választottuk" – mondta a GITWiT tervezési vezetője, *Peter Zatloukal*. „Olyan felhasználói felületet tervezünk, amely mérföldekkel a jelenlegi mobiltelefonokon található felületek előtt jár. Továbbá a Linux sokoldalú környezetet teremt elképzeléseink megvalósításához.

Mivel fejlesztői munkánk túlnyomó részét az alkalmazások és az objektumok szintjén végezzük, örömmel vesszük, hogy nyílt forrású rendszermagot használhatunk, és így mi is hozzájárulhatunk ahhoz a munkához, amelynek révén mások már könnyebben fejleszthetnek beágyazott megoldásokat.

Úgy véljük, hogy az új ötletek megvalósításának megkönnyítése a beágyazott termékek területén még akkor is a mi érdekünket szolgálja, ha ezáltal versenytársaink is megerősödnek, mivel az újabb szereplők hozzájárulnak a teljes piac bővítéséhez. Olyan világot szeretnénk, amelyben a könnyű piacra jutás ösztönzi a versenyt, a hangsúly pedig valóban a lényegre helyeződik – a jelenleginél élvezetesebb termékek fejlesztésére."

További tudnivalók a ➲ http://www.GITWiT.com címen találhatók.

Az IBM és a Citizen Watch közösen fejlesztik a Linux-alapú WatchPadet

Az IBM Research és a Citizen Watch bejelentette együttműködését, amelynek keretében a két cég Linux-alapú WatchPad-próbapéldányok és a velük kapcsolatos műszaki megoldások fejlesztésébe kezdett. A közös tervezet az IBM korábbi Linux Watch elképzeléseire épít, céljául pedig újszerű személyi adatelérési eszközök fejlesztését tűzte ki – a mindennapi életünk minden pilla-

natát átalakító számítógépes korszak jegyében. Az IBM Research az elmúlt évben mutatta be először a Linux Watchot, a cél akkor az volt, hogy a Linux sokoldalúságát szemléltesse, amely az S/390-es nagygépeken és a legapróbb készüléken egyaránt megállja a helyét. A Citizen azért döntött az IBM-mel való együttműködés mellett, hogy új szolgáltatásokat és megoldásokat fejlesszen, amelyeket a jövő – például távközlési eszközként is használható – intelligens óráiban láthatunk majd viszont. A Citizen a burkolat és az alkatrészek, például a kijelzők és a beviteli eszközök tervezésében jeleskedik. Az IBM a vas és a rendszer felépítésével, valamint a programokkal – köztük a Linuxszal – járul hozzá a fejlesztéshez. A két cég tervei között szerepel, hogy egyetemekkel is együtt fog működni, a közös fejlesztés során megosztja velük a WatchPad-megoldást, így remélve a következő nemzedékbeli intelligens eszközök fejlesztésének felgyorsítását.

A WatchPad nagysebességű, de kis fogyasztású, 32 bites MPU-val, 16 MB flashmemóriával és egynegyed VGA-felbontású (320×240 képpont) LCD-kijelzővel bír. Vezeték nélküli kapcsolatot Bluetooth- és infravörös csatolón keresztül tud teremteni. A felhasználók az érintőképernyő, a gombok és egy módosított tekerőgomb segítségével vihetnek be adatokat és utasításokat a gépbe. Emellett gyorsulásmérő is került a készülékbe, segítségével akár a kar mozdulatait is beviteli eszközként lehet majd használni.

Az alábbiakban a WatchPad műszaki adatai közül soroltam fel néhányat, amelyek az IBM Research által közzétett adatlapról származnak:

A vas

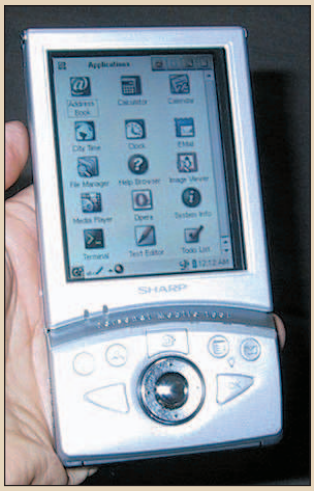
- Mérete: 65 mm×46 mm×16 mm
- Súlya: 43 g (csuklószíj nélkül)
- Processzor: nagysebességű, alacsony fogyasztású, 32 bites MPU (18–74 MHz órajellel)
- Beviteli eszközök: érintőképernyő, tekerőgomb, egyéb gombok
- Kijelző: 320×240 képpont felbontású, szürkeárnyalatos, folyadékkristályos kijelző
- Memória: 8 MB alacsony fogyasztású DRAM, 16 MB flashmemória
- Felületek: Bluetooth vezeték nélküli kapcsolat (v1.1, hangátvitelre is képes), IrDA (v1.2), RS-232C (foglalaton keresztül)
- Egyéb: hangszóró, mikrofon, rezgőmotor, ujjlenyomat-olvasó, gyorsulásmérő
- Tápellátás: Li-Ion akkumulátor
- Foglalat: RS-232C, AC-átalakító és AA-elemek

Programok

- Operációs rendszer: 2.4-es változatú Linux-rendszermag
- Grafikus felület: Microwindows
- Bluetooth-verem: IBM BlueDrekar (L2CAP, SDP, RFCOMM)

Újdonság a láthatáron

A Sharp Electronics már november óta vesz fel rendeléseket a fejlesztőktől az új Zaurus SL-5000D (fejlesztői változat) Linux-, illetve Java PDA-készülékére. A fejlesztői változat ára 399 dollár, a gép 32 MB DRAM-ot és 16 MB flashmemóriát tartalmaz.



A Zaurus SL-5000D Linux, illetve Java zsebtitkár

A készülék 206 MHz órajelű Intel StrongARM processzorra épül, amely a teljes rendszert egyetlen lapkán tartalmazza, és 3,5" méretű, 240×320 képpontos (egynegyed VGA-felbontású), 65 536 szín megjelenítésére képes TFT LCD érintőképernyővel rendelkezik, valamint – értelemszerűen – beágyazott Linux operációs rendszert futtat. A programverem a Lineo Embedix, a Trolltech Qt Palmtop Environment fejlesztésére, az Opera webböngészőjére, valamint egy PersonalJava v1.2-vel egyenértékű Java-futtatási környezetre épül. A gép két bővítőfoglalattal rendelkezik: egy biztonságos digitális (SD) kártyafoglalattal, amelyet elsősorban flashmemóriákhoz használhatunk, valamint egy CompactFlash aljzattal, amelyet főleg adatátviteli felületekhez, digitális fényképezőgépek, flashmemóriák és egyéb kiegészítők csatlakoztatásnál alkalmazhatunk.

Az SL-5000D egyedi és rendkívül értékes jellemzőinek egyike, hogy teljes méretű QWERTY billentyűzettel rendelkezik, amelyet a készülék alsó részét lefelé csúsztatva érhetünk el. Bővebb tájékoztatás a
➲ http://developer.sharpsec.com honlapon található.

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

© Kiskapu Kft. Minden jog fenntartva

Séta a LinuxWorld beágyazott oldalán

Folytatjuk sétánkat és beszámolónkat azok számára, akik nem tudtak eljutni a rendezvényre, vagy valami más kötötte le a figyelmüket.

Lineo

A Lineo a tavalyi évvel szemben – amikor több standot is megtöltött, mivel felvásárolt néhány beágyazott Linuxszal foglalkozó vállalkozást – idén nem állított ki,

hanem sajtókonferencia keretein belül tett stratégiai bejelentéseket. Ezek között szerepelt többek között, hogy a Motorola a Lineo Embedix Digital Media magját választotta DCT5000 típusú set-top-boxába, valamint bejelentette a tervezett GPL Compliance Toolsetet is. Az eszközkészlet annak eldöntésében segíti majd a fejlesztőket és vállalkozásaikat, hogy mely engedélyeztetési elvek vonatkoznak a programjaikra, és melyek a vonzataik.
➲ http://www.lineo.com

Lisa Systems

Ha valaki megkérdezi a Compaqtól, hogy mikor és hol juthat hozzá Linuxszal előtelepített iPAQ-hez, valószínűleg a Lisa Systemshez fogják irányítani. A Lisa a Compaq hatalmas LinuxWorld-pavilonjához csapódva mutatta be iPAQ–Linux párosát, amely grafikus keretrendszerként a Trolltech Qt, illetve az Embedded rendszerét használja.
➲ http://www.lisa.de

LynuxWorks

A Lineóhoz hasonlóan a LynuxWorks sem állított fel saját standot a LinuxWorldön. Az Intel pavilonjában azonban az érdeklődők rájuk bukkanhattak – itt lehetett megtekinteni BlueCat nevű Linux-rendszerüknek az Intel Internet Exchange Architecture (IXA) felületére átültetett változatát.
➲ http://www.lynuxworks.com

MontaVista

A MontaVista hatalmas standja mintha önálló életre kelt volna a rendezvényen belül: hemzsegett az új termékek és a műszaki megoldások bemutatóitól. A rendezvénycsarnok mennyezetén függő hatalmas, kalapos Tux-figura alatt a következő érdekességeket találhattuk:

A Hard Hat Linux (HHL) 2.0 keresztfelületes fejlesztőeszközeit, amelyeket elsősorban az Alchemy AU1000 nevű, a teljes rendszert egyetlen lapkán tartalmazó set-top-box hivatkozási felületére terveztek. A látogatók megismerkedhettek az új KDevelop IDE használatával, valamint a MontaVista Target Configuration Toollal, és az újonnan GPL-engedélyeztetés alá vont Library Optimizer Tool eszközzel is.
A MontaVista nemrég bejelentett High Availability Framework rendszerének bemutatóját is megtekinthettük, amelynek során egy CompactPCI-alapú Linux-rendszer folyamatos mozgóképlejátszást végzett hálózati adatfolyamról, miközben a három ethernetkábel közül egyet, illetve kettőt leoldottak.
Egy HHL-re épülő set-top-box készülék bemutatójára is sor került, valamint az új Hard Hat Graphics futtatására is az IBM Redwood nevű, PowerPC 405-alapú hivatkozási felületén.
A HHL futtatása iPAQ zsebtitkáron egyrészt a Hard Hat Graphics, másrészt a Trolltech Qt/Embedded használatával zajlott, és mindkét eszköz grafikus felhasználóifelület-építő eljárásainak ismertetése sem maradhatott el.
Két IBM-termék – a VisualAge Micro Edition, mely egy Java-jellegű virtuális gép, valamint a ViaVoice, amely mind beágyazott, mind asztali rendszereken használható beszédfelismerő program – bemutatásán is részt vehettek az érdeklődők. A ViaVoice-alapú, a felhasználótól függetlenül működő parancs- és vezérlőrendszer állítólag mindössze 200 KB memóriát foglalt.
Egy elragadó, a gépek beágyazott Linux-alapú vezérlésével kapcsolatos bemutatóról is szót kell ejtenem, ennek során ugyanis az Intrinsyc apró, StrongARM-alapú, CerfBoard névre hallgató SBC-je HHL-t futtatva egy önjáró Lego MindStorm robotot vezérelt; a robot úgy nézett ki, mintha a Csillagok háborújának valamelyik részéből lépett volna elő.
➲ http://www.mvista.com

PalmPalm

A PalmPalm a Tynux Box névre hallgató, StrongARM-alapú zsebtitkár hivatkozási felületét és a Tynux Linux operációs rendszert mutatta be. Ezenkívül megismerhettünk egy új, koreai gyártmányú mobiltelefon-, illetve zsebtitkár készüléket is.
➲ http://www.palmpalm.com

Red Hat

A Red Hat Embedded Linux Developer Kit előzetes változatát *Joe deBlaquiere* vezetőmérnök mutatta be. Az eszköz létrehozásának célja, hogy a fejlesztők számára megkönnyítsék a Red Hat Linux beágyazását. DeBlaquiere szerint az eszköz általános Red Hat SRPM-eket használ, így „a beágyazott rendszereknél is élvezhetjük a Red Hat üzembiztosságát”. Számos előre elkészített összeállítás érhető el; található köztük egészen apró, éppen csak elindítható, valamit indításra és hálózatkezelésre

© Kiskapu Kft. Minden jog fenntartva

is alkalmas rendszer és így tovább. A fejlesztők az összeállításokhoz egy könnyen használható, grafikus felületű beállítóprogram segítségével szükség szerint adhatják hozzá a további rendszerösszetevőket. A Red Hat rendszerépítő tevékenységének deBlaquiere szerint érdekes jellemzője, hogy a Lineo LIPO és a MontaVista LOT megoldásával ellentétben a Red Hat könyvtárcsökkentő, illetve egyszerűsítő folyamata négy meghatározott EL/IX profilt használ, így a létrejövő beágyazott rendszerek képesek lesznek együttműködni egy jól körülírt API-készlettel. Emellett a Red Hat RedBoot hibakereső, illetve rendszerindító programja is általános összetevőként került be az Embedded Linux eszközkészletbe. DeBlaquiere elmondta, hogy a készlet első próbaváltozatának megjelenése néhány héten belül várható. Mind a RedBoot, mind a Red Hat másik operációs rendszere, az eCOS egyaránt megtekinthető volt a Red Hat standján.

➲ http://www.redhat.com/embedded

A beágyazott Linux eszközök honlapja

REDSonic

A REDSonic négy új termékkel jelentkezett: a Secure SOHO átjáró-, illetve tűzfalmegoldással; a Window-Based Terminallal, ami olyan vékonyügyfél jellegű, terminálprogram, ami Linux-alapú hozzáférést nyújt windowsos alkalmazásokhoz; a LinuxBIOS névre keresztelt ROM-alapú Linux-rendszermaggal, amely biztonságos és gyors rendszerindítást lehetővé téve alkalmas a hagyományos BIOS-programok lecserélésére; valamint a REDSonic PowerPC kezdőkészlettel az MPC 832-höz. A REDSonic figyelemre méltó felhasználói alkalmazást is hozott: a SignSite-ot, melynek fejlesztője a Clarity Visual Systems. A SignSite fényes, színes digitális hirdetőtábla, amelyet nagy forgalmú helyeken hirdetések elhelyezésére és tájékoztatásra lehet használni. A készülék beágyazott Ampro Encore 500 Pentium-alapú SBC-t tartalmaz, amely a REDSonic REDICE-Linux operációs rendszerét futtatja.

➲ http://www.redsonic.com

RidgeRun

A RidgeRun a TI digitális fényképezőgép hivatkozási felületén futó DSPLinuxot mutatta be. A rendszer alapját alkotó, a teljes rendszert egyetlen lapkán tartalmazó TI TMS320DSC21 lapka egy RISC processzort és egy DSP-t is tartalmaz. A lapka *Rudy Prince* – a RidgeRun elnöke – szerint számos csúcskategóriás HP és Kodak fényképezőgépben megtalálható. Ugyancsak megismerhettük a DSPLinux fejlesztői környezetének rendkívül takaros „készülékszimulátorát". Várható tehát, hogy hamarosan a digitális fényképezőgépekben láthatjuk viszont a DSPLinuxot? – kérdeztük Prince-től. A válasz: igen. „A beágyazott Linux egyre növekvő szerepet fog játszani az ethernet-, a 802.11- vagy a Bluetooth-kapcsolattal, vagy egyéb csúcskategóriájú szolgáltatásokkal rendelkező eszközökben" – állította Prince.

➲ http://www.ridgerun.com

Az SH/Linux-tervezet

Miközben a Japan Linux Association standján haladtam keresztül, a nem nyereségközpontú szervezetek területén észrevettem két Linux-bemutatót, amelyek japán gyártmányú játékgépeken futottak: a Sony Playstation 2 és a Sega Dreamcast gépén. Az előbbit a Sony nemrég kiadott PS2 SDK-jának, az utóbbit a Linux-SH tervezet erőfeszítéseinek köszönhetően sikerült életre kelteni.

Tuxia

A Tuxia TASTE Embedded Linuxát mutatta be új set-top-box hivatkozási felületen, valamint néhány valóban mutatós vékonyügyfélrendszeren. Megtartották iPAQ zsebtitkáron futó TASTE operációs rendszerük első Linux-World-bemutatóját is. Az iPAQ-bemutató során megtekinthettük a Tuxia kisméretű Nanozilla böngészőjét, amely 802.11-es szabványú összeköttetésen keresztül érte el a Világhálót. A Tuxia két fontos bejelentést is tett a rendezvény kapcsán: a teljes rendszert tartalmazó lapkákat fejlesztő Rise céggel való együttműködést, amely szerint a TASTE Embedded Linux a Rise SOC termékeit is támogatni fogja, valamint hogy az Intel által a végfelhasználók számára fejlesztett médiaátjáró hivatkozási felületéhez a TASTE-et fogják beágyazott Linux operációs rendszerként használni.

➲ http://www.tuxia.com

Transvirtual Technologies

A TVT egy tévé set-top-boxot, valamint Java-alkalmazásokat mutatott be új Java-, XML-, illetve Linux-alapú XOE-felületéhez.

➲ http://www.transvirtual.com

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

… a végleges második változatban olyan fejlesztések lesznek, amelyek a zsebtitkárok mellett a következő nemzedékbeli intelligens telefonok támogatására is alkalmassá teszik.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Cégcsokor – 5. rész

(Sorozatunkban olyan cégeket gyűjtünk csokorba, amelyek régebb óta számos területen Linuxot alkalmaznak.)

Balaton Bútorgyár Rt.

Elsőként lássuk, milyen kiépítést vesznek igénybe a cégnél: egy Pentium III-as processzorral, 256 MB RAM-mal és 10 GB-os merevlemezzel felszerelt gépet használnak, amelyen operációs rendszerként Debian GNU/Linux fut. A következő feladatokra alkalmazzák: a vállalat számítógépeinek biztonságos csatlakoztatása az Internetre, levelezés, webelérés felhasználónkénti méréssel, webszolgáltatás, valamint a levelezés vírusszűrése. Ezeket a feladatokat a következő programok látják el: levelezés – Sendmail; víruskeresés – Amavis-perl-11, Kaspersky Labs AVP, F-Secure Antivírus, Virusbuster; a vírusadatbázisok frissítése egyébként naponta történik. A Web eléréséhez a Squid proxykiszolgálót alkalmazzák, amelynek naplófájljait az `sqmgrlog` nevű programmal elemzik. Webkiszolgálóként Apache-t és PHP4-et használnak, a dinamikus oldalak „háttértáraként" pedig MySQL adatbáziskezelő-kiszolgálót üzemeltek be. Ezenkívül Tripwire-t (lásd még a Linuxvilág 9. számának 34. oldalán) is futtatnak.

A felhasználók száma 20–25 között mozog. Az Internetre kábeltévés előfizetésen keresztül csatlakoznak. A rendszerről úgy készítenek biztonsági mentéseket, hogy a merevlemez lemezterületeit mentik.

A szarvasi Habar Kft.

A cég munkatársai nem mindennapi feladat megoldására vállalkoztak: egy baromfi-előnevelő klímaszabályozásának megoldását valósították meg Debian GNU/Linux alatt. Ehhez szükségük volt a kapcsolók távoli ellenőrzésére, valamint a fontosabb mérési értékek tárolására, amelyeknek alapján különböző kimutatásokat készítettek. *Kutas Ferenc*, a rendszer készítője programja nevét – Fremen – egyik kedvenc olvasmányából, *Frank Herbert* Dűnéjéből kölcsönözte.

Nem gyerekjátékról van szó, amit az is jól mutat, hogy 80 ezer pulyka élete függ a rendszer működőképességétől, rendszerleállás esetén ugyanis az állatok korától és a beállításoktól függően 40 perc és 4 óra közötti időtartam alatt elpusztulnak! Ez azonban csak akkor történhet meg, ha nincs olyan, aki észleli a leállást. Amennyiben ezt időben érzékelik, természetesen még hosszabb áram- és fűtésszünet esetén sem pusztulnak el. A hangsúly tehát azon van, hogy bármilyen jól működne is a rendszer egy esetleges üzemzavar esetén, a kezelőszemélyzet nélkül az állatok hosszú távon akkor sem élnék túl.

A régi rendszer XT-s volt, amely azonban jócskán kiöregedett, és a csere szükségessége vetette fel az új rendszer megalkotását. A rendszer két, egyenként négyólas telephelyen működik. Minden ólban egy-egy PC-t helyeztek el, az egyik telepen 486-os gépek, a másikon pedig 450 MHz-es Celeronok működnek, továbbá a telepek pihenőjében egy központi gép is található, ahonnan a telepen lévő összes ólat látják (négy virtuális konzolon keresztül). A masinák dupla kábelezéssel UTP-n és vékony ethernetkábellel is össze vannak kötve. A gépek egyelőre RSH-n tartják a kapcsolatot, amelynek kiváltása azonban már a terveik között szerepel; ezt TCP- vagy UDP-kapcsolat fölött saját protokollal kívánják megvalósítani. Ezenkívül grafikus felületet is ki szeretnének alakítani.

Az alkatrészek egy része is saját tervezés és fejlesztés eredménye, amelyhez a fejlesztő GNU-PCB, GNU/Linux `geda-gschem` és gEDA Netlist programokat alkalmazott. A pic16×84, a panelek és az rs232-rs422 átalakító saját tervezésűek, illetve fejlesztésűek. A PC-k vízmentesítésének megoldására is szükség volt, a kábelezés szintén az ő munkájukat dicséri. Saját (rs232(over)) rs422 protokoll került kialakításra. A program prototípusa Perlben készült és az RSH-t használja. Végső célkitűzésük, hogy az egész forrást C nyelvre ültessék át (így a gcc és a gnu make által lefordíthatóvá váljon), valamint a jelenlegi konzolos megoldás mellé valamilyen grafikus kezelőfelületet alakítsanak ki. A fejlesztés nyomon követésére CVS-t, és nem utolsósorban vim-et, awk-t és grepet használnak. A fejlesztő megjegyzése: „programozási munkáimat nagymértékben a Linux-kezdő és a Linuxlista (illetve archívumaik) támogatták, az alkatrészmegoldások életre hívásában pedig a Chipcad és a Codix levelezőlista volt a segítségemre".

Az állatok miatt mindenképpen folyamatos felügyelet szükségeltetik, de a tervezés során fontos szempont volt, hogy maga a rendszer minél kevesebb karbantartást igényeljen – tehát egy év elteltével is ugyanúgy kell működnie mindennek, mint az üzembe állításkor. Kutas Ferenc ígérete szerint, amennyiben a prototípus kiváltása sikerül, a rendszert GNU/GPL alatt kívánja megjelentetni.

MTA KFKI Részecske- és Magfizikai Kutatóintézet

A KFKI FTP-archívumának tárolását és frissítését Debian GNU/Linux látja el. Az intézet által „birtokolt" gép kiépítése a következő: Chieftec Jumbo ház, ASUS CUR-DLS alaplap, két 800 MHz-es Pentium III processzor, 1 GB ECC RAM, három darab 3ware Escalade 6800 IDE-RAID-vezérlő, két 30 GB-os IBM-DTLA-307030-típusú IDE merevlemez az operációs rendszer számára (RAID1-be kötve) és öt 75 GB-os IBM-DTLA-307075-típusú IDE-merevlemez az adatoknak (RAID5-be kötve). Az ár/teljesítmény arányának javítása érdekében IDE-RAID-vezérlőket alkalmaznak. Ez a típusú vezérlő hardveres RAID-et valósít meg: IDE-merevlemezekből felépített RAID-0/1/5/10-et SCSI-merevlemezként emulál az operációs rendszer felé. Az így létrehozott SCSI-merevlemezeket a BIOS és az operációs rendszer felcserélt sorrendben látja, ami egyedül az indítórésznél számít, de annál viszont kiemelkedően fontos szerepet tölt be. Először ReiserFS fájlrendszert alkalmaztak, de az archívumot tartalmazó RAID5 összeomlásakor (több merevlemez hibája okozta) a ReiserFS nagyon bonyolulttá tette a helyreállítást, ezért

© Kiskapu Kft. Minden jog fenntartva

az újratelepítés során az ext3 fájlrendszert választották. Ugyanis az indítórészt a RAID5-ön helyezték el, amely annak újraépítésekor elveszett, és így nem lehetett indítani, holott a RAID1-en lévő rendszer tökéletes volt. Megoldást jelentett volna egy olyan indítólemez készítése (vagy keresése), amelyen 3ware és ReiserFS-támogatás is létezik, ez azonban túl sok időt vett volna igénybe. Ext3 esetében az indítólemezről csak a 3ware-támogatás szükséges, ilyet pedig könnyebb volt találni. Továbbá sokat számított az ext2 fájlrendszerrel való együttműködési képessége is. Az egyetlen pluszszolgáltatás – amiért naplózó (journaling) fájlrendszert alkalmaznak – a gyors helyreállítás lehetősége, ugyanis a sebességbeli különbségek nem igazán játszanak szerepet egy FTP-kiszolgálónál. 300 GB-os lemezterület esetén meglehetősen hosszú ideig tartana, amíg az operációs rendszer ellenőrizné a fájlrendszert, hogy helyrehozza az esetleges hibákat. A hálózati kártya alaplapra integrált Intel EtherExpressPro 100 (egyébként az alaplapon integrált SCSI-vezérlő is található). A gép szünetmentes tápegységre van kötve.

Az archívum körülbelül száz forrás tükrözését jelenti a világ minden sarkából, pillanatnyilag 150 GB méretben. A jól behatárolt feladatnak köszönhetően az alkalmazott programok köre szűk. Az FTP-kiszolgálói feladatokat pure-ftpd, a webkiszolgálói teendőket pedig Roxen látja el. Az archívum rsync segítségével is elérhető.

Az archívumról nem készül mentés: ha valami elveszne, újból letöltik. Az operációs rendszer beállítási állományairól időnként a hálózaton keresztül „pillanatfelvétel" készül egy SPARC/Solarison lévő szalagos egységre. A binárisokat nem mentik, amennyiben szükséges, az egész rendszert újratelepítik, és a lényeges (mentett) beállítási állományokat visszamásolják. „Nulláról" ugyanis körülbelül egy-két óra alatt fel lehet állítani egy ilyen rendszert, amiben azonban már az is benne foglaltatik, amit *Kadlecsik József* így fogalmazott meg: „szeretek szöszmötölni, és egy-két dolgot kézzel fordítani".

A gép felügyelete igen kevés időt igényel, a legtöbbet a különböző parancsállományok megírására fordítja, amelyek a legkülönbözőbb feladatok önműködővé tételére készülnek.

Vajon miért a Linuxot választották? Mert költséghatékony és a számítástechnikai környezet gyakorlatilag egységessé vált: SPARC/Solaris és Intel/Linux-kiszolgálókat üzemeltetnek, minden más rendszer „kihalt" (korábban HP-t és SGI-t is használtak). Meglehetősen régóta alkalmaznak Linuxot, a rendszer gazdája első Linux-rendszerét még 1994 tavaszán telepítette.

További feladatok ellátására is használnak Linuxot:

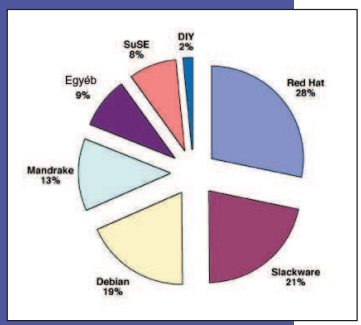
- proxykiszolgálóként
  Pentium II 400 MHz-es processzorral, 256 MB RAM-mal, AHA-294X SCSI-kártyával, egy 8 GB-os SCSI-merevlemezzel szerelt gép a rendszernek, és két 8 GB-os SCSI-merevlemezt pedig a gyorstár „vett használatba". Debian GNU/Linux és Squid látja el a proxykiszolgálói feladatot.
- nyomtatókiszolgálóként
  Pentium I 75 MHz-es processzorral, 16 MB RAM-mal szerelt gép két közvetlenül rákötött nyomtatóval (HP LaserJet 6MP és 4000), amely egy távoli nyomtatóként használt Xerox Document Centerrel, valamint egy modemkártyával működik. A gépen Slackware Linux fut, továbbá Hylafax kezeli a Számítóközpont bejövő és kimenő faxait, a Magicfilter pedig a Xerox-féle szűrőkkel működik. Fájlkiszolgálóként Samba üzemel, hogy a Windowsok is el tudják érni a számukra szükséges állományokat.
- tűzfalként
  AMD Athlon 700 MHz-es processzoros, 256 MB RAM-mal, 3Com 3c59x-es hálózati kártyákkal szerelt gép működik Debian GNU/Linux felügyelete alatt.

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

Linux-szeletek – az önkéntesek adatai alapján

A Linux Counter (➲ http://www.counter.li.org) önkéntes szerveződés. Csak azokat a felhasználókat és gépeiket tartja nyilván, akik veszik a fáradságot és jelentkeznek. 2001. október 19-ig közel kétszáz országból csaknem kétszázezer bejelentkezés érkezett a Linux Counterhez. Láttuk-hallottuk rovatunkban gyakran foglalkozunk különféle (Netcraft, Tucows, Evans Data és más forrásokból származó) Linux-kimutatások közlésével. Ebben a hónapban úgy gondoltuk, megmutatjuk, hogy a felhasználók saját bevallásuk szerint milyen Linux-változatokat alkalmaznak, beleértve a „csináld magad" fajtákat is. A kördiagramról leolvasható az eredmény. Amennyiben nem lennél vele elégedett, szavazz te is úgy, hogy bejelentkezel a Linux Counter weboldalán!

Linux Counter jelentés: 2001. negyedik negyedév



Újra a védjegyekről

A „számítógép” szót nem alkalmazhatjuk az általunk árult számítógép védjegyéül.

A védett nevek tartalmazhatnak olyan mellékneveket, mint „szuper”, „extra”, „tökéletes”, de ezek csekély értékkel bírnak.

Az ember egyaránt vehet Cadillac autót és Cadillac kutyaeledelt is.

A védjegy kiválasztása olyan a művelet, amelyet legjobb egy erre szakosodott jogász segítségével végrehajtani.

A jó védjegy kiválasztása csakúgy, mint a jó boré is, hozzáértést és körültekintést igényel. A megérzésre és a józan észre hagyatkozva gyakran születnek hatástalan védjegyek, éppen úgy, ahogyan a találomra kiválasztott bor is könnyen eredményezhet rosszemlékű vacsorát. Egyes marketingszakemberek, akik pedig igazán lehetnének tájékozottabbak is, termékeik számára jogilag hatástalan védjegyeket javasolnak. Erről természetesen nem árulhatok el túl sok részletet, ezért inkább költött példákkal szemléltetem azokat a rosszul megválasztott védjegyeket, amelyekkel találkoztam. Nem választhatunk termékünket vagy szolgáltatásunkat megnevező védjegyül például köznevet. Ez a szabály megakadályozza, hogy valaki magántulajdonná tegyen bármely hétköznapi szót, vagyis a *számítógép* szót nem alkalmazhatjuk az általunk árult számítógép védjegyéül.

Nem használhatjuk az írott vagy beszélt nyelv furfangjait sem ennek megkerülésére: például nem jegyezhető be a *Le Autó* mint a legújabb autótípus megnevezése. Attól, hogy egy védjegyet idegen csengésűvé teszünk, még nem lesz több általánosan használt köznévnél.

Nem hozhatunk létre védjegyet a termék nyilvánvaló szolgáltatásainak egyedi módú leírásával sem. A *TelefonCsengő*-t nem jegyeztethetnénk be egy olyan termék neveként, amely a bejövő telefonhívás esetén hangot ad. Továbbá nem téveszthetjük meg a vásárlókat azzal, hogy szándékosan félrevezető leírást adunk. A *Nyílt Forráskód* nem alkalmazható egy kereskedelmi, zárt forrású operációs rendszer megnevezésére. Az olyan védjegyek, amelyek „pusztán leíró jellegűek vagy félrevezető leírást adnak” tehát nem jegyezhetők be (15 U.S.C. §1052(e)).

Egy másik gyakori hiba, amikor némelyek a versenytársak védjegyeinek ismertségét próbálják kihasználni (egy új pizzázót *MacPizzá*-nak vagy egy mikroprocesszort *Pentalium*-nak nevezve), hogy a piac azonnal felfigyeljen a termékre. Ezeket a védjegyeket valószínűleg megtámadnák, mivel jó eséllyel „félreértést okoznak vagy tévedéshez vezetnek, vagy megtévesztőek” a termék eredetének vonatkozásában (15 U.S.C. §1052(d)).

A védett nevek tartalmazhatnak olyan mellékneveket, mint *szuper*, *extra*, *tökéletes*, de ezek csekély értékkel bírnak. Az ilyen kifejezések annyira elkoptatottak, hogy már nem alkalmasak termékek vagy szolgáltatások megkülönböztetésére. Ezenkívül úgy sem hozhatunk létre saját védjegyet, hogy a fenti melléknevek bármelyikét valaki más védjegyéhez fűzzük hozzá.

Védjegyünket vagy szolgáltatásnevünket meg kell óvnunk attól, hogy a termékünket vagy a szolgáltatásunkat jelentő köznévvé váljanak. Ezért próbálja például a Xerox szorgosan megelőzni, hogy az emberek azt mondják: „készítek egy xeroxot erről” ahelyett, hogy „Xerox-másolatot” mondanának. A védjegy általános értelművé válását megakadályozandó mindig használjuk melléknévként, sohasem főnévként.

A védjegy nem létezik elszigetelten, mindig meghatározott termékekkel vagy szolgáltatásokkal összefüggésben használatos, hogy ezen termékek és szolgáltatások eredetét vagy származását jelezze. Éppen ezért ugyanazt a szót több cég is használhatja védjegyként, mindaddig, amíg a termékek vagy szolgáltatások különböznek.

Az ember egyaránt vehet Cadillac autót és Cadillac kutyaeledelt, és nem fordul elő az a félreértés, hogy egy vásárló az egyik terméket keresi, így véletlenül a másikat veszi meg.

A védjegyekkel foglalkozó jogászok a lehetséges védjegyeket osztályokba sorolják, és növekvő sorrendbe állítják őket aszerint, hogy mennyire alkalmasak védjegynek, illetve milyen mértékű védelmet jelentenek. Ezek az osztályok a következők:

- Leíró – bizonyos nevek leírják a terméket, megnevezik a tulajdonost vagy a termék származási helyét. Közvetlen benyomást szolgáltatnak az áru összetevőiről, minőségéről vagy jellemzőiről. Az általános köznevek és a leíró jellegű nevek közötti különbséget világítja meg a Deep Bowl Spoon példája: *A „Deep Bowl” kifejezés a tárgy egy lényeges tulajdonságát rögzíti. Pusztán leírja a terméket, mert tájékoztat róla, hogy annak mély (deep) a fejrésze (bowl). Mindazonáltal nem a tárgyat megnevező „általánosan használt köznév”, mivel az eszköz nem egy mély tál (deep bowl), hanem egy kanál. A „Kanál (Spoon)” szó nem pusztán leírása a tárgynak – azonosítja a tárgyat –, így tehát a kifejezés köznév.* (*Fletcher*, „A leíró védjegyek megtámadhatatlanságával kapcsolatos félreértés”, 64 Trademark Rep. 252, 260 (1974).)
- Érzékletes – egy név akkor érzékletes, ha képzelőerő, gondolattársítás és az érzékelés segítségével következtethetünk belőle az áru természetére, például Orange Crush, Cuisinart és London Fog.
- Tetszőleges – amikor egy köznevet szokatlan módon használnak, például ilyen az Apple, az Apache és a Python. A tetszőlegesen kiválasztott megnevezésekből gyakran lesznek sikeres védjegyek, mert jobb eséllyel nyerik meg a vásárlók tetszését és könnyebben is jegyzik meg őket.
- Kitalált – olyan szavak, amelyeket kizárólag azért hoznak létre, hogy védjegyként szolgáljanak; ilyen például az Altoids vagy a Kodak. A kitalált kifejezések kiválóan alkalmasak védjegynek, hiszen nem lehet őket valódi szavakkal összetéveszteni.

Lawrence Rosen (www.rosenlav.com) magángyakorlatot folytató jogász a kaliforniai Redwood Cityben. A Nyílt Forrás Kezdeményezés (Open Source Initiative) ügyvezető igazgatója és jogtanácsosa (➲ www.opensource.org).

© Kiskapu Kft. Minden jog fenntartva

Új termékek

Accelerated X Summit 2.0

Már letölthetők az Accelerated X Summit 2.0 sorozatba tartozó grafikus meghajtóprogramok. Az új

meghajtóprogramok OpenGL 1.2.1-megfelelő megjelenítő csővezetéken alapulnak, és több mint harminc kártyához és hordozható géphez nyújtanak támogatást 2D-s és 3D-s üzemmódban. A Series 2.0 négy sorozatban kapható: Desktop, Laptop, Multihead és Workstation, mindegyik változatnak négy kiadása létezik. A v2.0-ban az alábbi újdonságok és továbbfejlesztett szolgáltatások jelentek meg: 3D-sztereó a legtöbb kártyán; Color Magic, amely a rendszerszínek beállítására alkalmas grafikus segédprogram; Video Window az MPEG- és más videoforrások megtekintéséhez; XiG Direct Access a grafikus kártya OpenGL-alkalmazásokból való közvetlen eléréséhez; végül DualView a kétképernyős megjelenítéshez, amely a legtöbb grafikus kártyához és hordozható géphez használható.
Adatok: Xi Graphics, Inc., 1801 Broadway, Suite 1710, Denver, Colorado 80202,
telefon: 800-946-7433,
➲ http://www.xig.com

Matisse 5.0

A Fresher Information Corporation bejelentette a Matisse 5.0 megjelenését, amely egy az objektumalapú alkalmazások és webszolgáltatások gyors fejlesztését és telepítését lehetővé tevő adatbázisprogram. A Matisse ötvözi a természetes objektumkezelést a kiszolgálóalapú SQL-lel, így kiküszöböli az objektumok relációs leképezését. Külső adatforrások, alkalmazásfejlesztő és jelentéskészítő eszközök eléréséhez az ODBC- és a JDBC-támogatás áll rendelkezésre. A Matisse 5.0 többek között az alábbiakat támogatja: Solaris, NT/2000, FreeBSD, SQL-, UDDI- és XML-szabványok, sokféle programnyelv, például Java, C, C++, Python, Perl és PHP. A fejlesztők számára ingyenesen letölthető változat az alább megadott webhelyen érhető el.
Adatok: Fresher Information Corporation, 575 Market Street, 13th Floor, San Francisco, California 94105,
telefon: 415-356-8100,
➲ http://www.fresher.com

AdminForce CGI Auto Audit

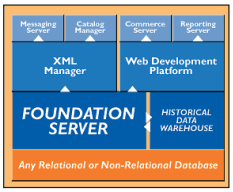
A LinuxForce Inc. bejelentette az AdminFOrce CGI Auto Auditot, azt a CGI-parancsfájlelemzőt, amely a parancsfájlok szerkezetének átvizsgálásával azonosítja a biztonsági hiányosságokat. A gépesített feldolgozással naponta több száz parancsfájlt lehet ellenőrizni a hagyományos sorról sorra történő átvizsgálás pontosságával. Ráadásul elérhető egy „metakarakter-tisztító" programkönyvtár, amelynek segítségével a parancsfájl áttekinthetőbbé tehető és a hamis riasztások elkerülhetővé válnak. Az ellenőrzés a CGI Auto Audittal távolból is elvégezhető.
Adatok: LinuxForce, Inc., 100 Glendale Road, Upper Darby, Pennsylvania 19082,
e-mail: operations@linuxforce.net,
➲ http://www.linuxforce.net

Ximian Evolution 1.0

A Ximian Evolution 1.0 egy alkalmazásban fogja össze a következő személyes és csoportos munkával kapcsolatos adatkezelési feladatokat: levelezés, naptár, névjegyalbum és feladatlista. Az Evolutiont úgy tervezték, hogy jól illeszkedjen a vállalatok sokszínű számítástechnikai környezetébe, így számos módon képes adatokat cserélni és minden fontos kapcsolattartási szabványt ismer. Ez lehetővé teszi, hogy a Linux- és a Unix-rendszereket közvetlenül csatlakoztassuk a vállalati hálózati és üzenettovábbító rendszerbe. Az Evolution támogatja az SMTP, a POP, az IMAP és más üzenetprotokollokat, valamint a naptárak megosztását az MS Outlook, Lotus Notes és más iCalendart támogató alkalmazások között. A Ximian Connector for Microsoft Exchange 2000 bővítmény szintén elérhető.

Már megjelent a Ximian Evolution 1.0.1 is, és az alkalmazás teljes egészében magyar nyelvre is le van fordítva (munkatársunk *Tímár András* jóvoltából).
Adatok: Ximian, Inc., 401 Park Drive, 3 West, Boston, Massachusetts 02215,
telefon: 617-375-3800,
➲ http://www.ximian.com

XAO 1.0



Megjelent a XAO Inc. XAO 1.0 programja. Az Apache-ra épülő webszolgáltató program lehetővé teszi az adatok a legkülönfélébb forrásokból történő szabványosított beépítését, például relációs adatbázisokból, hagyományos alkalmazásokból vagy más alkalmazáskiszolgálókból. A XAO 1.0 Foundation Server API bármilyen relációs adatbázis felett képes működni, így az objektumszintű nézet a relációs lekérdezések sebességével épülhet fel. A Foundation Server tárolási és mély keresési feladatokat is ellát más XAO-modulok és webalkalmazások számára, beleérve az e-kereskedelmet is. A XAO 1.0 példaprogramokkal együtt érkezik, ezáltal az alkalmazások anélkül testreszabhatóak, hogy a jól megalapozott kód elhagyása miatt aggódnunk kellene.
Adatok: XAO, Inc.,
221 East Walnut Street, Suite 102, Pasadena, California 91101,
telefon: 877-796-7437,
➲ http://www.xao.coms

© Kiskapu Kft. Minden jog fenntartva

A hónap szakmai tanácsai

/proc/kcore: F3ITOwn3d!

Két Red Hat 7.1-et futtató kiszolgálóm van, mindkettőn fut a Tripwire. A Tripwire kétszer is jelentette, hogy a */proc/kcore* megváltozott. Rendben lévő ez így? A két esemény között ugyanis nem indítottam újra a rendszert.

A */lib/libc-2.2.2.so* ellenőrzőösszege ugyan megváltozott, azonban minden más változatlan maradt (dátum, inode stb.). Az `rpm -V glibc` szintén kimutatta, hogy az MD5 ellenőrzőösszeg megváltozott. Mivel nyilvános webkiszolgálót üzemeltetek ezen a gépen, újratelepítettem a glibc-csomagot. Ezután a Tripwire panaszkodott a megváltozott dátumok és fájlleírók (inode) miatt, ez azonban természetes egy csomag újratelepítése után. Tudomásom szerint egy jól beállított tűzfal mögött csücsülök. Hogyan változhat meg a *glibc* ellenőrzőösszege? Miért módosul a *kcore* ellenőrzőösszege?
Magnus Sundberg, Magnus.Sundberg@dican.se

Nem ismerem a *libc* változásának okát, azonban a */proc/kcore* módosulására szolgálhatok magyarázattal. A */proc/kcore* nem valódi fájl, hanem a rendszer fizikai memóriáját tartalmazza. Természetesen a fizikai memória tartalma időnként megváltozik, máskülönben nem sok hasznát vennénk a számítógépnek. Ne aggódj emiatt!
Scott Maxwell, maxwell@ScottMaxwell.org

Hol van a /dev/sdc?

Két IDE-eszközöm van: egy 40 GB WD HDD és egy CD-RW-meghajtó, továbbá egy SIIG SCSI-kártyám is van. Úgy tudom, a Linux felismeri a SCSI-kártyámat, de valamiért nem tudom elérni a SCSI-meghajtóimat. Ha a
`/dev/MAKEDEV sdc[0,1, ,n]`
paranccsal próbálkozom, a következőt írja ki:
`don·t know how to make sdc[n].`
Az SCSI-merevlemezem azonosítója a 3. Az SCSI CD-ROM azonosítója 5, az SCSI-kártya alapértelmezés szerint 7. Szerintem minden megfelelően le lett zárva.
Derrick Blackwell, db101055@hotmail.com

Úgy tűnik, azt várod, hogy a 3 SCSI cím a */dev/sdc* lesz. A Linux nem így működik. A legalacsonyabb azonosítójú SCSI-merevlemez lesz a */dev/sda*, függetlenül a SCSI címétől. A következő legalacsonyabb azonosítójú SCSI-merevlemez lesz a */dev/sdb*, és így tovább. A SCSI CD a */dev/scd0* vagy a */dev/sr0* lesz – mindkettő ugyanolyan jól működik.
Scott Maxwell, maxwell@ScottMaxwell.org

Az egér-, monitor-, illetve billentyűzet-átkapcsoló tönkreteszi a munkaasztalt

Két gépet birtoklok, az egyiken Windows, a másikon Linux fut, és egy egér-, monitor-, illetve billentyűzet-átkapcsolóval használom őket. Amikor elindítom a munkaasztalt (ebben az esetben a Gnome-Enlightenment párost), mindkettő működik. Ha azonban a Mandrake-rendszerről átkapcsolok a másikra és vissza, az egér teljesen elvész. Ellenőriztem a vezetékeket, újraindítottam a *gpm* démont, és megnyomtam a CTLR+ALT+BACKSPACE gombokat, hogy kilépjek a munkaasztalból. Amennyiben a munkaasztalt újraindítom, az egér ismét működik, de csupán addig, amíg át nem kapcsolok a Windowsba és vissza.
A Windows oldalán minden gond nélkül át tudok kapcsolni a Linuxra, majd vissza, és a Linux is rendben működik, amíg át nem kapcsolok.
Egy régi Dell P90-es gépen Linux/Mandrake 7.1-et futtatok PS/2-es egérrel. A *gpm* a `-t ps/2` kapcsolóval fut. Lehetséges, hogy egy olyan démon okozza a gondot, amelyet biztonsági megfontolásokból nem futtatok? Az `amd`-t, az `atd`-t, az `innd`-t, az `lpd`-t és a `portmap`-et letiltottam.
Dave Dennis, dmd@speakeasy.org

Ez nagy valószínűséggel nem a Linux-telepítés hibája. A PS/2-es egerek beállításai a gép elindításakor alaphelyzetbe kerülnek. Az átkapcsoló feladata, hogy a beállításokat helyreállítsa az átkapcsolás után, és a Linux semmit nem tud az átkapcsolóról.
Christopher Wingert, cwingert@qualcomm.com

Ha arra mehetnék, nem kéne nekem arcszesz!

Gondjaim akadtak, amikor két hálózati kártyát szerettem volna használni Red Hat 7.2-t (2.4.3-12) futtató gépeimben. A rendszer mindkét hálókártyát felismeri, és be is tudom állítani őket. Két T1-es vonalam van két különböző szolgáltatóhoz, azaz két különböző IP-hálózattal kell dolgoznom. Hogyan adhatom meg külön-külön az alapértelmezett utat (default route) az egyes hálókártyákhoz?
Mike Kercher, mike@CamaroSS.net

Nem adhatsz meg két alapértelmezett utat. Meg kell adnod a Linuxnak, hogy melyik forgalom melyik hálózatra menjen. Úgy tűnik, el akarod osztani a terhelést a két T1 között. Nézd meg az EQL- vagy a Bonding-meghajtót. Az útválasztást BGP-n keresztül is végezheted. Olvasd el az Advanced Routing HOWTO-t a
➲ http://www.linuxdoc.org/HOWTO/Adv-Routing-HOWTO.html címen.
Christopher Wingert, cwingert@qualcomm.com

Igen, uram, van arcszeszünk – erre tessék!

Körülbelül 333 Linux-kiszolgálót felügyelek, és az adatközpontban minden négyzetméternyi hely számít. Karbantartási céllal el kellene érnem a kiszolgálókat konzolon keresztül, de egyiknek sincs monitora. Lehetséges-e soros terminált csatlakoztatni konzolként?
Karthik, nkk@hotmail.com

Olvasd el a Serial Console HOWTO-t a
➲ http://www.linuxdoc.org/HOWTO/Remote-Serial-Console-HOWTO címen.
Christopher Wingert, cwingert@qualcomm.com

© Kiskapu Kft. Minden jog fenntartva

Nincs meg a „mail”, de leveleznem kell

Kialakítottam otthon egy kis linuxos hálózatot, amely két asztali és egy noteszgépből áll. Az összes gépen szeretnék levelezni, de csak a régebbi asztali gépről tudok, noha mindegyiken azonos Netscape-beállításokat használok. Az újabb gépen és a noteszgépen a levelek lekérésekor azt a hibaüzenetet kapom, hogy *Netscape unable to locate server mail*, azaz a Netscape nem találja a mail nevű kiszolgálót. A „mail” az internetszolgáltatóm (Cox@home) által megadott kiszolgálónév, ami tökéletesen működik a régi gépemen. A másik két gépen az új levelek letöltésekor a Netscape mindig megkérdezi a jelszavamat, annak ellenére, hogy a beállításokban megadtam, hogy jegyezze meg őket. Vajon a Netscape rossz beállításfájlt olvas be?
Eric Smith, esmith289@home.com

Úgy tűnik, hogy a működő géped rendelkezik az internetszolgáltatódhoz tartozó teljes értékű tartománynévvel (FQDN – Fully Qualified Domain Name), a másik kettő viszont nem. Próbáld meg a gépnév további részét hozzáadni a Netscape beállításokhoz (például mail.pelda.hu, ha a pelda.hu a tartományneved).
A másik lehetőség, hogy a */etc/resolf.conf* `search` sorában megadod a helyes tartománynevet, így nem kell mindig begépelned.
Christopher Wingert, cwingert@qualcomm.com

Nem telepítettem a LILO-t

Nemrég frissítettem a rendszermagot, az újraindítás előtt azonban elfelejtettem telepíteni a LILO-t. Ha most egy másik lemezről indítom a rendszert, ez a lemez befűzhető, és az `fsck` hibátlannak mutatja – mégsem tudom róla elindítani a rendszert.
Willie Strickland, willie@istrick.com

Használd a rendszerindító hajlékonylemezt, és add meg, hogy a rendszer a merevlemezről induljon. A LILO parancssorába írj be valami ilyesmit:

```
linux root=/dev/hda1
```

Ezután írd át a *lilo.conf*-odat és úgy telepítsd a LILO-t, mintha most telepítetted volna az új rendszermagot. (Ezt a `lilo` parancs segítségével teheted meg.)
Ben Ford, ben@kalifornia.com

Az „autoupdate” nem találja a könyvtárat

Próbálom használni az `autoupdate`-et. Amikor beírom: `autoupdate`, ezt a hibaüzenetet kapom:

```
CWD failed no such directory or file
```

Amennyiben az `autoupdate -debug 2` parancsot adom ki, ugyan anonim felhasználóként be tud lépni, de az alábbiakat írja ki:

```
CWD failed.
Error: Failed to check directory at
➥ftp.redhat.com:
pub/redhat/linux/updates/7.1/en/os
no such file or directory.
```

Amikor kézzel írtam be, hogy `ftp.redhat.com`, anonim felhasználóként be tudtam lépni a `/pub/redhat/linux/updates/7.1/en/os` könyvtárba.
Adharsh Praveen R., adarsh@multitech.co.in

Az elérési út elejéről lemaradt a `/` (perjel). Így add meg az `autoupdate`-nek:

```
/pub/redhat/linux/updates/7.1/en/os.
```

Christopher Wingert, cwingert@qualcomm.com

Az „fsck” nem tudja olvasni a szuperblokkot!

Amikor kiadom az `e2fsck /dev/hda` parancsot, a következő üzenetet kapom:

```
The superblock could not be read or
does not describe a correct ext2
filesystem. If the device is valid
and it really contains an ext2
filesystem (and not swap or ufs or
something else), then the superblock
is corrupt, and you might try running
e2fsck with an alternate superblock:
e2fsck -b 8193 <device>
```

Amennyiben megteszem, amit mond (`<device>` = `dev/hda`) ugyanezt az üzenetet kapom. Éppen negyedszerre történt meg velem az eset. A többi alkalommal egyszerűen újraindítottam a Red Hat-rendszert, és semmi fontosat nem vesztettem el. Most azonban nem szeretném elveszteni azokat a programokat, amelyekkel dolgozom.
Bob Wooden, bwooden@computelnet.com

A */dev/hda* egy meghajtó. Bár lehetséges fájlrendszert telepíteni egy meghajtóra, nem túl valószínű, hogy ez a helyzet. Add ki az `fdisk -l /dev/hda` parancsot, ebből megtudod, hogy milyen lemezrészek találhatók a meghajtódon. A legvalószínűbb, hogy a lemezrészed a */dev/hda1* (hacsak nem használsz több operációs rendszert a gépeden).
Christopher Wingert, cwingert@qualcomm.com

Sikertelen biztonsági mentés

DDS4 szalagos meghajtót használok Compaq Alphán (Red Hat 7.1) több linuxos munkaállomás (PC-k Red Hat 7.0 és 7.1 rendszerekkel) fájlrendszereinek biztonsági mentésére. Telepítettem az SSH-t, ezért a DSA-hitelesítésen keresztül éjjelente egyszerű mentőparancsfájlok futnak cronból a jelszó bekérése nélkül. A fájlokat `tar` (gtar 1.13.17) és `dd` segítségével mentem a szalagra. Valahol a `tar`-, illetve `dd`-folyamatok mélyén egyszercsak az alábbi üzenetet kapom:

```
select: Bad file descriptor
```

A mentőparancsfájl itt megszakítja a `tar`-, illetve `dd`-folyamatot, és a többi utasítással folytatja a működést.
Martin Olivera, molivera@ucsd.edu

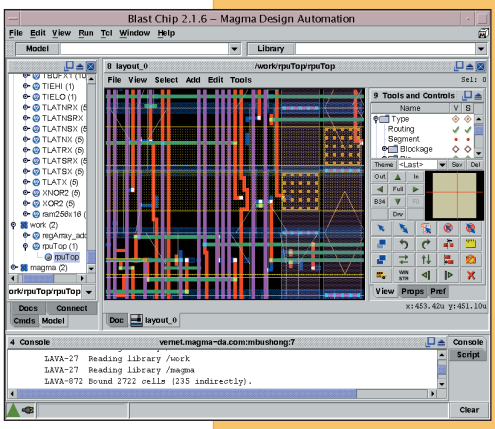
Ez a hiba az OpenSSH korai változataiban fellelhető hiányosság miatt jelentkezik. Frissítsd az SSH-t a legutóbbi üzembiztos változatra az összes érintett rendszeren.
Don Marti, dmarti@ssc.com

A Linux Journal honlapján számtalan gond megoldásához találhattok további segítséget. A Sunsite tüköroldalait, a gyakran feltett 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.

© Kiskapu Kft. Minden jog fenntartva

A Linux a számítógéppiac főáramában

Robin kalauzolásával megszemléljük, hogyan növekszik a Linux jelenléte egyre nagyobb mértékben a kiszolgálók, a munkaállomások és az asztali gépek piacán.



Linus Torvalds Just for Fun című szórakoztató életrajzi könyvében meséli, hogy amikor azt kérdezgetik tőle, vajon a Linux átveszi-e az asztali operációs rendszerek főszerepét és rést üt-e a Microsoft egyeduralmának falán, mindig igennel válaszol. Bár ez a jóslat még nem vált valóra, mégis egyre több okunk van hinni benne.

Az alábbiakban az IMB, a HP, a Compaq, a Dell, a Gateway és az SGI cégeknél nézünk körül, hogy megtudjuk, mi is a helyzet a Linuxszal az ügyféloldali operációs rendszerek vonatkozásában.

A Linux Apache-alapú webkiszolgálóként olyan sikeressé vált, hogy az IDC piackutató cég közzétett elemzése szerint tavaly a kiszolgálók operációs rendszereinek 27 százaléka Linux volt, míg a Windows részesedése ezen a téren 41 százalékot tett ki. Az asztali gépek esetében azonban a Windows tavaly 91 százalékban képviselte magát, míg a Linuxnak csak 1,4 százalék jutott. Az Apple részesedése 3,6 százalékra csökkent tovább. Az asztali gépek operációs rendszereinek piacán csak a Windows és a Linux tud növekedést felmutatni, az előbbi 11 százalékot, az utóbbi pedig 25 százalékkal növelte a részesedését.

Az IDC azonban nemcsak a számítógépekkel együtt eladott, hanem az ingyenes forrásokból feltelepített Linux-rendszerek, és a kalózmásolatokból származó Windows-példányok számát is igyekszik figyelembe venni. *Al Gillen*, az IDC elemzője megjegyzi: „A leggyakrabban használt alkatrész a Linux-rendszereknél az újrahasznosítható windowsos PC".

Talán ellentmondásosnak tűnik, hogy egy ingyenes operációs rendszer iránt ma nagyobb a kereslet a korszerű munkaállomások területén, mint az asztali PC-k piacán. A Linux nem a kisteljesítményű PC-k világából tör felfelé, hanem ellenkezőleg, a kiszolgálói részből vándorol lassan lefelé. Ennek oka egyrészt a Linux együttműködése a már régóta uralkodó különböző Unix operációs rendszerekkel a nagyteljesítményű felületeken, másrészt az, hogy nehezebb új támogatási rendszert kiépíteni a már meglévő windowsos helyett. Figyeld csak meg, hogy egyik OEM-terjesztőnél sem fogsz olyan gépeket kapni, amelyekre a Windowst és a Linuxot egyaránt telepítették, mert a Microsoft az eladókkal kötött titkos megállapodása ezt eleve kizárja. Erre a pontra a kormány sajnos nem mutatott rá a cég elleni perben.

Ma már az összes nagynevű PC-gyártó Linux operációs rendszerrel is kínálja a gépeit. Nem is olyan régen még alaposan eldugott weboldalakon tették közzé ezt a hírt, manapság azonban már a honlapokon is szerepel. Tekintsd meg a ➲ http://www.ibm.com/linux, a ➲ http://www.hp.com/linux és a ➲ http://www.compaq/linux vagy a ➲ http://www.dell.com/linux oldalakat! Nézzük, hogy a vezető PC-gyártók milyen linuxos megoldásokat kínálnak számunkra!

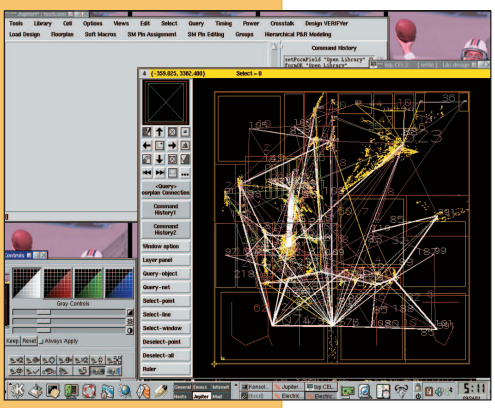
A személyi számítógép kategóriában az IBM többek között az IntelliStation munkaállomásokat, a NetVista asztali gépeket és ThinkPad hordozható gépeket kínálja. A ThinkPad A és T sorozatát OpenLinux eDesktop 2.4-es operációs rendszerrel kínálják. A ThinkPad T22 az első olyan linuxos számítógép, amihez a hivatalosan engedélyezett DVD-lejátszót, az InterVideo LinDVD nevű programot adják. A Linux alá írt nyílt forrású DVD-lejátszókat a hírhedt DeCSS-per óta a gyártók kerülik. A legtöbb DVD-film másolásvédett és az Egyesült Államokban érvényes DMCA törvény tiltja a másolásvédelmet kiiktató programok használatát.

„A munkaállomásokkal dolgozók közül egyre többen váltanak Linuxra, mivel gyors, egyszerű és megbízható" – állítja *Doug Oathout*, az IntelliStation munkaállomások marketingigazgatója. Bár a kiszolgálóknak szánt gépeken a Linux kiszorítja a Windowst, Oathout szerint a Windows-alapú munkaállomásokra ma még kevésbé veszélyes; a Linux inkább a Unix különböző változatait szorongatja meg. Az IBM sokfajta Linux-terjesztést támogat. „Valamennyi IntelliStation-modellünket kipróbáltuk a Caldera-, SuSE-, Turbolinux- és Red Hat-változatokkal is" – mondja Oathout. „Leggyakrabban a grafikus kártyák támogatottságát vizsgáljuk e próbák során. Az ATI-, nVidia- és Matrox-kártyákhoz készültek jó minőségű linuxos meghajtók, a 3DLabs kártyáihoz azonban még nincsenek ilyenek. Jelenleg ezen igyekszünk változtatni."

Az IBM a Linuxra elkülönített egymilliárd dollárból jelentős részt fordít a meghajtók fejlesztésére és kipróbálására. „Az IBM munkaállomás-piaca az elektronikus tervezést és a földtudományokat is erősen támogatja" – mondta Oathout. Az elektronikus tervezés önműködővé tételével ágazatában az élen járó cégek, mint például a Cadaence és a Mentor a termékeiket Linux alá is elérhetővé tették, ami arra bátorítja a felhasználókat, hogy a HP-UX-ról (a HP által kifejlesztett Unix) és a Solarisról Linuxra váltsanak. Az IBM úgy látja, hogy a pénzügyi területeken kevert Windows-Linux környezet van jelen. A linuxos asztali gépekre is léteznek ugyan kereskedelmi programok, de a Reutershez és a Bloomberghez való hozzáférés a Windows használatát kívánja meg. Népszerű kereskedelmi linuxos mozgóképkészítő programok közé tartozik a Maya, a Lightwave és a Softimage. A Linux alatti legfontosabb CAD-alkalmazások az ANSYS, Nastran és a Patran, valamennyi véges számú elemből álló rendszerek elemzésére képes. „A kőolaj-feldolgozó iparban használható alkalmazások fényes jövő előtt állnak"

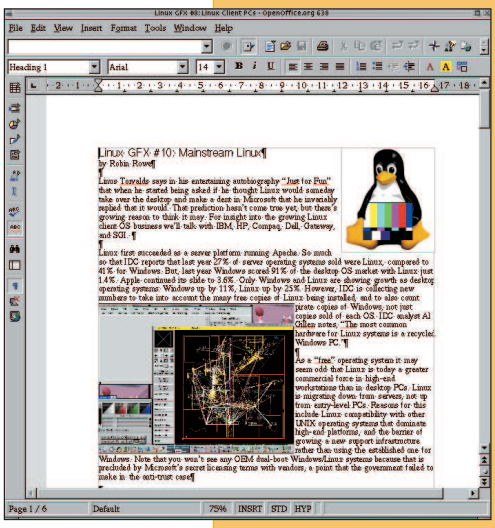
© Kiskapu Kft. Minden jog fenntartva

– állítja Oathout. „Jövőre ezek a cégek is Linuxra váltanak.”
A CATIA Windows operációs rendszerre írt CAD-alkalmazás, amit a francia Dassault Sysetemes cég fejleszt és az IBM értékesít. „A CATIA V5-öt sikerrel futtattuk Linuxon is” – jelentette be *Anthony Marechal* a cég médiakapcsolatokért felelős munkatársa. „Bár piacra vitelével kapcsolatban még nem hoztunk döntést, nyitottak vagyunk arra, hogy támogassuk a CATIA linuxos változatát, ha a piaci nyomás erre késztet bennünket.”
2001 szeptemberében a HP bejelentette, hogy 25 milliárd dollárért felvásárolja a Compaqot, valamikor 2002 folyamán. Az egyesítés után létrejövő új cég összesített 87 milliárd dolláros jövedelmével (HP – 47 milliárd, Compaq – 40 milliárd) közel akkora piaci részesedésre tesz szert, mint amekkorát a kilencvenmilliárd dolláros jövedelmű IBM birtokol. A HP és a Compaq gépei együttesen az Egyesült Államokban a kiskereskedelemben értékesített PC-k 75 százalékát teszik ki.
„Már mintegy másfél éve szállítunk Linux-alapú 3D-s munkaállomásokat” – mondja *Mike Balma* marketing-igazgató. „A Linux komoly fejlődést mutat a digitális animációk létrehozása terén, sőt már Hollywoodban is megvetette a lábát annak köszönhetően, hogy támogatja az olyan, ma már ipari szabványnak számító programokat, mint a Maya, a Houdini és a Shake. Sőt, a HP munkaállomásain a Maya csak Linux operációs rendszer alatt van hitelesítve”. A HP szoros együttműködésben segítette a Dreamworks céget, amikor az Linuxra váltott át (lásd a Linuxvilág 2001. augusztusi számának 8. oldalán megjelent cikket). A HP-munkaállomásoknak egyre nagyobb piacuk van az elektronikus tervezés önműködővé tételében, a telekommunikációs programok fejlesztése és a nagyképernyős pénzügyi-kereskedelmi alkalmazások területén. A HP-munkaállomások vásárlói jórészt a korábbi Solaris-vásárlók közül kerülnek ki. Ahogy az IBM-nél, úgy a Linux a HP-nél sem rengette még meg a windowsos munkaállomások piacát.
Valamennyi HP-munkaállomást a Linuxcare hitelesített a Turbolinux-, SuSE-, Red Hat-, Caldera-, Mandrake- és Debian-terjesztésekre. „Alapos minőségellenőrzésnek vetünk alá minden általunk használt Linux-változatot, valamint mélyrehatóan megvizsgáljuk, hogy a rendszermag milyen teljesítményt nyújt az eszközvezérlők, például egy IDE-meghajtó kezelésében” – magyarázza *Arthur Tyde*, a Linuxcare CEO munkatársa. A telepítés során esetleg felbukkanó akadályok megoldására találunk javaslatokat a HP, a Compaq, az IBM és a Dell gépeire a ➲ http://www.linuxcare.com/labs/certs weboldalon.
Míg a HP-munkaállomások mindenhol Linux operációs rendszerrel is megvásárolhatók, addig asztali és hordozható gépeket csak külön megrendelésre szállítanak Linuxszal. Kivételt Ázsia és Európa képez. „Kelet-Európa jóval nyitottabb a Linux asztali gépeken történő felhasználására” – mondja Balma. „A gépeket általában nem előretelepített Windows-rendszerrel veszik, és az árakra is érzékenyebbek.” A Winmodem körüli gondok miatt a HP otthoni felhasználóknak szánt Pavilion nevű asztali és hordozható gépei még nem támogatják teljes mértékben a Linuxot, de a cég szakemberei már dolgoznak a megoldáson.
A Debian Project volt vezetője, *Bruce Perens* 2000 decemberében csatlakozott a HP-hoz mint a linuxos stratégiák kidolgozásáért felelős szakember. A HP által fejlesztett meghajtóprogramok jó része, főként a nyomtató-meghajtók forráskódban is elérhetők. A szintén nyílt forrású Gnome grafikus felület lett a HP-UX gépek szabványos munkaasztala. A Gnome-t a HP és a Sun egymástól függetlenül, de körülbelül egy időben karolta fel.
A HP cáfolja, hogy saját Linux-terjesztést adnának ki. „Biztonsági szempontból egy kicsit csiszoltunk a Red Haten, és sokan ezt keverik össze egy önálló Linux-változattal. Valójában csak annyi történik, hogy egy már meglévő operációs rendszert kapcsolunk össze fejlett biztonsági megoldásokkal” – mondotta Balma.



„Három éve foglalkozom Linuxszal a Comaqnál” – mondja *Judy Chavis*, a Compaq cég Linux-programjának igazgatója. A Compaq a Red Hat-, a SuSE-, a Caldera- és a Turbolinux-terjesztéseket használja.
„Nincs túl nagy kereslet a linuxos asztali gépek iránt, de a munkaállomások nagyon keresettek az elektronikus tervezés területén” – mondja Chavis. „A munkaállomások iránti kereslet a múlt félévben futott fel, előtte nem sok minden történt ezen a téren.” Az elektronikus tervezés önműködővé tételének területe, az olaj- és a benzinkereskedelem, valamint a digitális animációk készítésével foglalkozó cégek képezik a Compaq linuxos munkaállomásainak felvevő piacát.
Munkaállomásaikat hitelesített és előre telepített Linuxszal szállítják, de az asztali és hordozható gépek területén a Compaq nem kínál linuxos változatokat.
Az elektronikus tervezésekben használt egyik lapkatervező programjának készítője, a Magma Design Automation 2001 májusában a Blast Chip és Blast Fusion nevű termékeit Linux alatt is elérhetővé tette. „Az elmúlt évben erőteljesen növekedett a kereslet termékeinknek Linux operációs rendszerre készített változatai iránt” – mondja *Bob Smith*, a marketing és üzleti fejlesztések részlegének igazgatóhelyettese. Az elektronikus tervezésben használt alkalmazások eladásában egy másik vezető cég, az Avanti is készül arra, hogy teljes lapkatervező rendszert tegyen elérhetővé Linux alatt a negyedik negyedévben. „Vásárlóink igényeire válaszolva az elmúlt években számos Avanti-terméknek készítettük el a linuxos változatát” – mondja *Dr. Paul Lo*, vezető operátor.
A Dell Computer cég, amely jelen pillanatban 33 milliárd dolláros jövedelmével a negyedik helyen áll a piacon,

© Kiskapu Kft. Minden jog fenntartva



a HP-Compaq egyesítés után a harmadik helyet foglalhatja majd el. A Dell Precision gépeinek 28 százalékos világpiaci részesedésével a Dell jelenleg az első számú munkaállomás-szállító. 1999 végén, mielőtt a Linux felé nyitottak volna, az összes Dell-munkaállomás Windows-alapú volt, bár kiszolgálóikhoz a Novell NetWare és a SCO UnixWare operációs rendszereket is kínálták. „Élénk érdeklődést tapasztalunk a filmpiac felől" – mondja *David Graves*, a cég szóvivője. A közelmúltban a cég az Egyesült Államokban felhagyott a végfelhasználóknak szánt linuxos PC-k támogatásával. Ez a szolgáltatás a cég weboldalán is elérhető volt, bár sohasem hirdették igazán. „Cégünk profilját a vásárlói igények alakítják" – mondja Graves. „Azon testületi fogyasztóink számára, akik testreszabott, gyárilag összeállított, ötven vagy annál több egységből álló rendszert szeretnének, természetesen feltelepítjük a Linuxot az OptiPlex asztali és a Latitude hordozható gépeinkre." A Dell az ausztráliai piacon is beszüntette a végfelhasználóknak szánt linuxos kínálatot.

A kilencmilliárdos jövedelemmel rendelkező, személyi számítógépeket gyártó Gateway cég 2001 augusztusában jelentette be, hogy visszavonul az Egyesült Államokon kívüli piacokról, és lehetséges, hogy 25 százalékos munkaerő-leépítésre is sort kell kerítenie. 2001 februárjában a Gateway bejelentette, hogy vásárlóik hatékonyabb kiszolgálása érdekében a náluk jelenleg elérhető mintegy 23 millió lehetséges számítógép-kiépítés számát néhány százra csökkentik. „Az alapkiépítésekhez nem kínálunk Linuxot – mondja *Lisa Emard* szóvivő –, csak az egyéni összeállításokat igénylő, nagy tételben vásárló fogyasztóink számára." A Gateway arra hivatkozik, hogy nincs akkora igény a Linux iránt, hogy a kiskereskedelemben történő értékesítése kifizetődő legyen.

Az SGI cég veretlen piacvezető volt a kiszolgálók és a nagy teljesítményt fejlett grafikus képességekkel kombináló munkaállomások területén. A maga másfélmilliárdos jövedelmével ezen a piacon a cég már csak 11 százalékos részesedéssel bír, aminek legfőbb oka, hogy a piac a Windows-alapú munkaállomások felé mozdult el. A Linux a cég azon üzletpolitikájának része, amelynek segítségével a windowsos munkaállomások és a Cray szuperszámítógép kínálata terén tett nem éppen kifizetődő kirándulásából fel szeretne épülni. Bár az SGI felvásárolta, majd később eladta a Crayt, a szuperszámítógépek még mindig üzleti tevékenységük központi részét képezik, és olyan termékeik jelennek meg, mint az Origin3000 (2000-ben), az Origin2000 (1996-ban) és a Challenge (1993-ban).

„A Linux izgalmas számunkra" – mondja *Simon Hayhurst*, az SGI fejlett grafikai termékeket előállító részleg vezetője. „Célunk, hogy az O2 és az Octine képességeivel ruházzuk fel az olcsó Intel-gépeket is. Az olyan nagy teljesítményű, eredetileg IRIX-rendszerre készített termékeket, mint az XFS, az OpenGL és az Open Inventor nyílt forrásban tesszük elérhetővé Linux alatt." Amint a megfelelő felbontású televíziózás a PC-ken is életképessé válik, az SGI a HDTV felé fog fordulni. Olyan feltörekvő piaci rész ez, ahol szükséges a többletteljesítmény. „Elcsípünk valamennyit a Dell felső piaci területén bonyolított üzleteiből is" – mondja Hayhurst. „A számítógépek piacán mindig az alsó területek határozzák meg a piac felső szegmensében zajló mozgásokat". Az SGI kínálatában sem az asztali, sem a hordozható gépek nem szerepelnek…

A MicroTron2000 kínálja a legolcsóbb asztali gépeket, amelyeket a helyi (San Diego, Calafornia) számítógépes böngészdében, a ComputorEdge-ben találtunk. Míg a nagy cégek esetében az alacsony kategóriás PC-k árai 659 dollártól (Dell) 799 dollárig (Gateway) terjedtek, addig egy teljes AMD Duron 750-es MicroTron2000 PC 17 hüvelykes ViewSonic E70-es monitorral (179 dollár) már 418 dollárért kapható (ebből 239 dollárt tesznek ki az alkatrészek: 256 MB PC133-as RAM, 10 GB HDD, CD-ROM, modem, 100 Mb-es hálózati csatoló). A Microsoft programjai viszont nincsenek benne ebben az árban. A Win98 95 dollárba, a Win2K 139 dollárba kerül, de a Microsoft Office programcsomag nem is szerepel a cég kínálatában. *Charles Tran* üzletvezető elmondta: „Gépeinket az ingyenes Linux operációs rendszerrel is szeretnénk kínálni, de még nem sikerült egyetlen Linux-szakértőt sem találnunk, aki csatlakozna hozzánk. Számítógépeinkhez a Red Hat-terjesztést választanánk".

Ahogy a személyi számítógépek árai tovább csökkennek, úgy egyre inkább a Microsoft-programok válnak a rendszer legdrágább elemeivé. Az alacsony kategóriás termékek piacán ezáltal megvan rá a lehetőség, hogy a Microsoft operációs rendszerei nélkül gyártsanak használható PC-ket .

A Világbank becslése szerint a Földön élő hatmilliárd emberből több mint 1,2 milliárd él kevesebb mint napi egy dollárból (főként Dél-Ázsiában és Afrikának a Szaharától délre eső területein). További kétmilliárd csak egy hajszálnyival él jobban. Arányaiban nézve a világon csak kevés ember rendelkezik számítógéppel. Az Egyesült Államokban 285 millióan élnek, de az Internetet 102 millió ember használja tevékenyen. Idén a világméretekben eladott PC-k száma körülbelül 130 millió lesz. Ha azt tekintjük, hogy a világ népessége évente 75 millióval növekszik, és egy személyi számítógép elavulási ideje mindössze egy-két év, megállapíthatjuk, hogy csak nagyon lassan haladunk afelé, hogy mindenkit számítógéphez juttassunk. A Windows csak a könnyen elérhető

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

gyümölcsöket szakítja le. 37 százalékos árcsökkenés, vagyis PC-nként 241 dolláros megtakarítás bizony sokat számít a világ legnagyobb részén.
Mexikóváros éppen most van egy kétéves átmeneti időszakban, melynek végén a nyílt forrású programokhoz érkezik el, amelyeket a motorgyártó cégek már felfedeztek maguknak. A Gnome program elkötelezett híve, *Miquel de Icaza*, Mexikóváros egyetemének volt rendszergazdája személyesen kérte Fox elnököt, hogy az eMexico nevű széles körű informatikai programot nyílt forrású rendszerekkel valósítsa meg. A nyílt forrású rendszerek alkalmazásáról kormánykezdeményezések születtek Brazíliában, Franciaországban, Németországban, Dél-Koreában és Kínában is.
Az asztali rendszerek esetében a Microsoft Office programcsomag az a vízválasztó gát, ami a Linux-áradatot visszatartja. Az irodai csomag komoly vetélytársa a StarOffice lehet. A Sun Microsystems 47 millió dollárt költött a StarOffice megvásárlására és ingyenessé tételére, majd még inkább zavarba hozta a piaci elemzőket tavalyi bejelentésével, miszerint a 6.0-s változatot a GPL (General Public Licence) alatt fogja terjeszteni, nyílt forrásúvá téve ezáltal a kilencmillió sorból álló programkódot. Mivel a kiadást 2001 októberére tervezték, az OpenOffice 6.0 e cikk olvasásakor már valószínűleg hozzáférhető (jelenleg a StarOffice 6.0 próbaváltozata érhető el – a szerk.). A programban végrehajtott módosítások eredményeképpen a bosszantó beágyazott felület ezentúl választhatóvá válik, javul a támogatása a Microsoft Office által használt állományformátumokkal, valamint támogatni fogja a kínai, a japán és a koreai nyelvet is.
2001 júniusában a Sun bejelentette, hogy az amerikai hadsereg alá tartozó DISA (Defence Information System Agency) elkötelezte magát a StarOffice mellett. Az ügynökség a Pentagon és hatszáz egyéb katonai szervezet információs rendszerének kiépítéséért felelős. A StarOffice nem a Microsoft Office-t fogja kiütni a nyeregből, hanem az Applix unixos irodai programcsomag tízezer példányát. A Microsoft nem szállt versenybe, hogy betöltse a hiányos Solaris-támogatásból adódó űrt.
A StarOffice Linux-, Windows- és Solaris-rendszerek alatt is fut. A DISA úgy tervezi, hogy körülbelül 25 ezer példányban fogja telepíteni a programcsomagot. Ingyenes alkalmazás révén a StarOffice nem hoz közvetlen bevételt a Sunnak, azonban a cég és a DISA között kiterjedt terméktámogatási szerződés lépett életbe.
A Linux katonai, illetve kormányzati célú felhasználásában csupán annyi a gond, hogy biztonsági szempontból még nem hitelesítették (természetesen a Windows XP-t sem, egyedül csak a Windows NT 3.51-et). Azonban az NSA (Nation Security Agency) kifejlesztett egy saját Linux-változatot, a SELinuxot, amelyben az operációsrendszer-alapú biztonságot már szinte művészi szintre fejlesztették. Az NSA, DARPA és egyéb amerikai ügynökségek milliókkal támogatják a nyílt forrású fejlesztéseket.
A Microsoft állítja, hogy a Windows XP kiadásával a Windows DOS-öröksége már végleg a múlté. Az XP a Windows 2000/NT-magra épül, amely a VMS-rendszerekkel mutat hasonlóságot. A Windows 2000 volt az első megbízható Windows operációs rendszer, de az NT-vel osztozik abban a bosszantó tulajdonságban, hogy időről időre – amíg a rendszerkarbantartást végzi – másodpercekre lemerevedik. Az XP további kellemetlen vonásokat (szolgáltatásokat) vezet be: a Netscape-bővítők (plugin) támogatásának hiánya, a Java-támogatás hiánya (a Sunnal szembeni pervesztés következménye), a különleges betörés elleni védelem bevezetése (regisztráció zárolása egy adott szignatúrával rendelkező gépen), a .Net portál (vagy kapuőr), és a harmadik fél programjaival való összeférhetetlenségi hibák miatti megszokott állandó frissítési gondok.
A Linux szilárd helyzetet tudhat magáénak a kiszolgálók piacán, a munkaállomások piacára viszont most robban be. A kormányzati területeken is kiépítette már a hadállásait. A következő leküzdendő akadály az asztali rendszerek meghódítása. Jelenleg még nincsen olyan cég, amely komoly méretekben kínál Linux-alapú megoldásokat az asztali rendszerekhez. De az XP növekvő ára, és az egyre szélesebb körű Linux-támogatás a cégeket megfontoltságra készteti. A Ford európai képviselete (33 ezer asztali gép felhasználója) például sűrűn kacsintgat a Linux felé.
A Windows-felhasználók többségének álma egy olyan Windows 98, amelyik nem fagy le, valamint nem szolgál mozgó célpontként.
Mindössze egy év alatt a Linux a semmiből minden igényt kielégítő választássá nőtte ki magát a filmiparban. A filmstúdiók mind Linuxra kezdenek váltani. Az élenjáró ebben a DreamWorks; ők Shrek című sikerfilmjük elkészítéséhez főként Linux-alapú rendszereket használtak.
A „niche"-piacon beállt fejlődés megerősíti a Linuxot, ennek eredménye pedig az eszközmeghajtók fejlődése és a jobb grafikai támogatás. Minden egyes fejlődési lépés újabb akadályt mozdít el az asztali rendszerek felé való terjeszkedés útjából. Széles körű ipari támogatással a háta mögött a Linux megállíthatatlan.
Ahogy Linus véli: „A program olyan, mint a szex – ingyen a legjobb".

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

Linux-index 2002. január–február

1. Ennyi millióan nézték a 2001. szeptember 11-i terrortámadás eseményeinek folyamatos közvetítését: 21
2. A New York város által kért szövetségi támogatás (milliárd dollár): 54
3. Peru bruttó nemzeti összterméke (GDP) (milliárd dollár) 54
4. Elektronikus postafiókok várható száma a 2001. év végén: egymilliárd
5. A televízió ennyi év alatt érte el az egymilliárdos nézőszámot: 50
6. A telefónia ennyi év alatt érte el az egymilliárdos felhasználószámot: 100
7. Németország helyezése azon országok között, amelyekben a leginkább elterjedt a Netscape böngészők használata: 1
8. Netscape böngésző részesedése Németországban (százalék): 20,26
9. Netscape böngésző részesedése az Egyesült Államokban (százalék): 15,79
10. Netscape böngésző részesedése a világon (százalék): 13,17
11. Ennyi országból jegyez felhasználókat a Linux Counter: 188
12. A Linux Counter által jegyzett személyek száma 2001. október 19-én: 195 900
13. A Linux Counter által jegyzett gépek száma 2001. október 19-én: 111 942
14. Linux-felhasználók száma a Linux Counter becslése szerint legkevesebb: 3 918 000
15. Linux-felhasználók száma a Linux Counter becslése szerint legfeljebb: 97 950 000
16. „Névtelen” gépek helyezése a PC-„márkák” eladási listáján: 1
17. „Névtelen” PC-k piaci részesedési tartománya: 50–70
18. Ennyi millió gépzongorát adtak el csak az USA-ban 1930-ig: 2,5
19. 1930-ban az USA népessége (millió fő): 123,2
20. A Linux-oktatás bevétele 1999-ben (millió dollár): 10,3
21. A Linux-oktatás bevétele 2001-ben (millió dollár): 56
22. A Linux-oktatás tervezett bevétele 2004-ben (millió dollár): 285
23. A rendszerszintű oktatási piac mérete 2004-ben (milliárd dollár): 2
24. A rendszerszintű oktatási piacon a Linux tervezett részesedési aránya 2004-re legkevesebb (százalék): 5,8
25. Ugyanezen a piacon a Linux tervezett százalékos részesedési arányának felső határa: 15,3
26. A Linux-támogatási szolgáltatások bevételének tervezett, összetett éves növekedési aránya 1999-től 2004-ig (százalék): 86,9
27. Az Amazon.com technológiai költségei a Linuxra való áttérés előtt (millió dollár): 71
28. Az Amazon.com technológiai költségei a Linuxra való áttérés után (millió dollár): 54
29. Az Amazon technológiai kiadásainak százalékos csökkenése a Linuxra való áttérés eredményeként: 25
30. Egy 1000 felhasználót ellátó Linux-kiszolgáló Unix-hoz viszonyított költsége: 1/3-1/2
31. Ekkora értéket jelentenek az IBM-nek azok az eszközök, amelyeket közcélra adományozott az Eclipse.org projekt keretében (millió dollár): 40
32. Ennyi programeszköz-beszállító dolgozott az Eclipse-en az Eclipse.org megalakulásakor, 2001. november 6-án: 150
33. Az Eclipse készítésébe bevont egyedi fejlesztők száma ugyanekkor: 1200
34. A Koreai Légitársaság ennyi ezer pilótája és légikísérője használja az IBM linuxos eServerét menetrend-ellenőrzés céljából: 3
35. A befogadott projektek száma a SourceForge.net-en 2001. november 11-én: 29 253
36. A SourceForge.net bejegyzett felhasználóinak száma ugyanekkor: 290 500

Források

1.:	USA Today-, Nielsen-, illetve NetRatings-idézet
2–3.:	Time Magazine
4–7.:	Strategic Policy Research, Inc.
7–11.:	StatMarket (➲ http://www.statmarket.com)
11–15.:	Linux Counter (➲ http://wwwcounter.li.org)
16–18.:	Jon „Maddog” Hall of Linux International
19.:	US Census Bureau
20–26.:	International Data Corp. (IDC)
27–29.:	CNET
30.:	Dan Kusnetzky az IDC-től (a CNET történetében)
31–34.:	IBM
35–36.:	SourceForge.net

Három e-failure gyönygyszem

1. Az ingyen és a nyereség nem ugyanaz.
2. Kezdd kicsiben, és azután válj naggyá!
3. A fafejek nem is konyítanak az üzlethez.

(Michael Jardeen)

© Kiskapu Kft. Minden jog fenntartva

Eredeti és beszédes

Gondolatok a Jabber fiatal és lelkes fejlesztőcsapatáról.

A Jabber-tanácskozás nyitónapján az első szünetben a fejlesztőcsapat négy tagja tiszteletét tette nálunk. Elsőként ***Eliot Landrum*** érkezett, akinek köszönettel vettem dicsérő szavait az új ***Linux Journal*** szerkesztési változtatásait illetően. Ezután ***Julian Missig***, ***Justin Mecham*** és ***Schuyler Heath*** jött meg. Úgy néztek ki, mint egy középiskolai sakkcsapat, de az idők folyamán már megtanultam, hogy ne ítéljek első látásra. Fiatalok ugyan, de elismerésre méltó programozói képességgel rendelkeznek, és a Jabber nekik köszönheti, hogy eljutott az első tanácskozásáig. Megkérdeztem, hogyan mutatnák be a Jabbert azok számára, akik még sosem hallottak róla. Eliot szerint „osztott, nyílt forrású csevegőprogram". A többiek egyetértően bólogattak, de az arckifejezésükből azt olvastam ki, hogy „vagy amit akarsz" – ami tulajdonképpen szintén helytálló.

Valójában sokkal inkább szokatlanok voltak, mint nagyképűek. Eliot valami olyasmit mondott, hogy „üzeneteket küldünk ki az űrbe és várjuk, mit válaszolnak az idegen civilizációk". Ezek között a civilizációk között vannak azok a vállalkozások, amelyek élénk figyelemmel kísérik, mit is hoznak létre az ifjú titánok. A résztvevők listáján láttam, hogy képviseltette magát az Intel, az Earthlink, a Disney, a PeoplePC, a Swisscom és még egy sereg kevésbé ismert cég, amelyek állítólag nagyon érdekes dolgokkal foglalkoznak. Egyikük a sokat emlegetett Nuance, amely azon munkálkodik, hogy a mobiltelefonokat csevegőügyfélként lehessen használni valós idejű, beszéd-szöveg-beszéd átalakításos beszélgetésekben a Jabber is jelen volt adatátvitel segítségével. Az Oracom, ahol pedig a Jabbert valós idejű világméretű hálózatkezelésben használják fel. A fenti két példát „beágyazott csevegésnek" nevezhetnénk, ha ez a címke nem zárná ki az XML-útválasztást és a jelenlétkezelést – hogy csak néhány olyan erényt említsünk, amelyek a Jabbert könnyen beágyazhatóvá teszik. Ennek köszönhetően annyival meghaladja a csevegés AOL-féle szintjét, hogy teljesen új elnevezésért kiált.

Lehetne esetleg „felület"? A beágyazott Linuxhoz hasonlóan a Jabber is mélyen strukturált, de mivel inkább rejtett alkalmazásként szolgál, nem nevezhető igazán felületnek. Ezt a kifejezést mostanában nem is az operációs rendszerek, hanem jobbára a Világháló kapcsán használják. A Jabber feladata nem saját jelenlétének fitogtatása (vagy saját „védjegyesítése" a dotcom-zsargon szóhasználatával élve), hanem hogy segítse és (XML-alapú) útválasztással lássa el a szaporodó és mindenütt jelenlévő alapvető internetszolgáltatásokat.

Ránéztem a srácokra, és nekik szegeztem az udvariatlan kérdést: hány évesek vagytok? „Tizenkilenc" – mondta egyikük. „Tizenhét" – egy másik. „Tizenkilenc." „Huszonegy."

„Jeremie mennyi idős?" – kérdeztem. ***Jeremie Miller*** a Jabber számára azt jelenti, amit ***Linus Torwalds*** a Linuxnak: ő az a programozó, akinek a fejében motoszkálni kezdett a Jabber ötlete. „Már két gyereke van. Hány éves is, huszonhat?" „Azt hiszem, huszonnégy."

Rá kellett jönnöm, hogy idősebb vagyok, mint közülük bármelyik három együttvéve (ami új értelmet ad rangidős vezetőszerkesztői címemnek), de erről inkább hallgatok. Annak a megfigyelésnek sem adok hangot, hogy a Jabber az alkotóival együtt még sokkal közelebb van a nullához, mint amivé végül válni fognak. „Nézd csak, miket tudunk művelni!" hozzáállásuk a tanácskozáson tartott bemutatóik során felidézte bennem a húsz évvel ezelőtti időket, amikor a PC még vadonatúj volt, s minden vele kapcsolatos dolog újnak és izgalmasnak számított, noha még csak a prototípusai voltak a majdan nélkülözhetetlenné váló eszközöknek.

1999 elején még jóformán senki sem tudott a Jabberről – én sem. Ekkor beszélt róla a barátom, ***Perry Evans*** (legtöbben talán a MapQuest alapítójaként ismerik), és kíváncsi volt, mit gondolok róla. Elmondása szerint a Jabber egy tapasztalatlan 22 éves, Iowa állam vidékéről való programozó agyszüleménye. Akkoriban a hálózati csevegés ugyanúgy viszonyult az AOL rendszeréhez, mint az Internet: nyílt forrású és XML-alapú volt. Segítségével bárki elképzelhette saját csevegőkiszolgálóját, éppen úgy, mint egy nyílt forrású hálózati vagy levelezőkiszolgáló-rendszert. S mivel lényegében XML-útválasztó volt, csaknem minden alkalmazási terület felé érdekes kapcsolatokat „sejtetett".

Perry ötleteket várt tőlem arra nézvést, hogyan támogathatnánk a Jabber nyílt forrású fejlesztését egy olyan vállalkozás révén, amely Jabber-alapú termékek és szolgáltatások eladásából teremt elő pénzt. Akadtak is ötleteim, de tudtam, hogy ***Eric Raymondnak*** még több lenne, így meghívtam őt is, hogy csatlakozzon a beszélgetésünkhöz. Végül Perry és vállalata, a Webb Interactive Services megalakította a Jabber, Inc. társaságot, én pedig a tanácsadó testületben (ami azóta már az Open Board nevet viseli) helyet foglaltam Erickel és több ismerőssel a nyílt forrást támogatók (Open Source) közösségéből.

Craig Burton következő megjegyzésében felcsillant a remény: „A piac folyamatosan a moduláris szerkezeti felépítés felé halad. Ez a folyamat a szerkezeti rend és a szakértelem újraelosztásának kiegyenlítéséhez vezet." Mondanom sem kell, hogy ugyanez volt a Linux mint beágyazott operációs rendszer sikerének története is.

Véleményem szerint azonban a „szakértelem újraelosztása" kifejezés túlzottan leegyszerűsíti azt, ami valójában történt. Nem tartom véletlen egybeesésnek, hogy ***Linus Torvalds*** és ***Jeremie Miller*** is körülbelül 21 évesen mutatta be találmányát a világnak, sem azt, hogy lehetővé tették, hogy mi, a többiek végezzük el a kezdeti munka oroszlánrészét. Az a tény, hogy a Linux és a Jabber is olyan fiatal még, annál inkább nélkülözhetetlenné teszi őket.

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

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Távoli Linux-indítás

Áttekintjük a távoli Linux-indítás alapjait, a hálózati indítást támogató rendszermaggal kapcsolatos követelményeket és feladatokat.

A „távoli Linux-felgyület” kifejezés olyan Linux-munkaállomásokra vagy hálózati csomópontokra vonatkozik, melyek nem helyi adathordozóról töltik be a rendszermagot, hanem hálózaton keresztül (leggyakrabban ethernet) kapják meg az indításhoz szükséges utasításokat. Mivel a Linux-rendszermagot és magát az operációs rendszert is könnyű az igényeinknek megfelelően beállítani, a Linux-alapú rendszerek egymástól nagyon eltérő feladatokra is testreszabhatók, kezdve a webkiszolgálóktól a fürtbe kötött rendszerekig és az X-kiszolgálókig.

Miért hasznos a távoli felügyelet?

Az első kérdés, ami eszünkbe juthat, hogy miért van szükség egyáltalán arra, hogy Linux-rendszerünket távolról indítsuk el. Végtére is egy Linux operációs rendszer helyben történő telepítése nem kíván többet, mint hogy behelyezzük a telepítőkorongot, válaszoljunk néhány kérdésre, majd igyunk meg egy hosszúkávét, amíg a rendszer települ a gépünkre. Az egyedülálló munkaállomások esetében legalábbis ez a helyzet; mihelyt azonban megnő a fenntartandó és telepítendő munkaállomások száma, főként fürtelrendezésű vagy kiszolgálófarmok esetén, a helyi adathordozóról történő telepítés már kevésbé ésszerű megoldás. Az olyan tömörített kiszolgálók megjelenésével, mint amilyeneket az RLX Technologies cég is gyárt, a távolról történő rendszerindítás szinte már szükségszerű, mivel ezek a kiszolgálók nem tartalmaznak hajlékonylemezes vagy CD-ROM-meghajtókat. Úgy készítették el őket, hogy operációs rendszerük a helyi, nagy sávszélességű ethernethálózaton keresztül töltődjön be, és távolról lehessen őket felügyelni.

A távoli Linux-rendszerek előnyei

- Központosított, jelenlétet nem igénylő felügyelet: bár sok telepítésnél valaki fizikailag is jelen van, hogy a CD-ket és a hajlékonylemezeket (24/7) kezelje, számos olyan rendszer is létezik, amely mellett hosszabb ideig senki sem tartózkodik. Ha egy programozó egy ilyen rendszeren például távolról szeretne dolgozni, és olyan betöltőállományra lenne szüksége, amely valamelyik helyi adathordozón található, balszerencséjére ezt csak akkor teheti meg, ha valaki a távoli gép mellett tartózkodik, aki az adott adathordozó behelyezéséről gondoskodik.
- Tömör kiszolgálói megoldások: mivel a fürtbe fűzött és a kiszolgálófarmok esetén a változás egyértelműen a központosított, távoli felügyelet felé mutat, a CD- és a hajlékonylemezes meghajtók kezdenek elavulttá válni. Ezen helyi adathordozóknak már a puszta jelenléte is bizonyos formai követelményeket támaszt a gépekkel szemben, növelve ezáltal az elérhető legkisebb fizikai méretet. A hálózati csomópontban lévő gépek nagyobb tömörsége érdekében egyes gyártók a CD- és hajlékonylemezes meghajtókat kivonják a forgalomból.
- Sokoldalúság: lehetőség nyílik kijavítani az olyan állományrendszerrel kapcsolatos hibákat, amelyek megakadályozzák, hogy helyi adathordozóról indítsuk a rendszert. Például futtathatjuk az `fsck` parancsot egy távolról indított gép helyi meghajtóján és kijavíthatjuk az állományrendszer esetleges hibáit.
- Költségek: miért fizessünk olyan adathordozóért, amire nincs is szükségünk? Vannak cégek, amelyek merevlemez és egyéb helyi adathordozó nélküli munkaállomásokat árulnak biztonságos terminálkiszolgálói feladatokra. A biztonság itt abban nyilvánul meg, hogy mivel az alkalmazottak a helyi adathordozókhoz nem férnek hozzá, a kényes adatokhoz sokkal nehezebb hozzájutni.
- Segít kiküszöbölni a különböző változatokból adódó buktatókat: ha az összes munkaállomást távolról indítjuk egy közös rendszermag-képállomány segítségével, ezzel kiküszöbölhetjük, hogy a helyi adathordozón a rendszermagot állandóan frissíteni kelljen. Elégséges a közös rendszermag-képállományt egyszer frissíteni, ahelyett, hogy a munkaállomások merevlemezein, vagy ami még rosszabb, a rendszerbetöltő hajlékonylemezein kellene a változtatásokat elvégezni.

A távoli rendszerindítás folyamata

A távoli rendszerindítás utánozza a helyi indítás folyamatát, de a kettő között akad néhány fontos különbség. Anélkül, hogy az egyes feladatokat végrehajtó szolgáltatásokat részleteznénk, az alábbiakban áttekintjük a hálózatról történő rendszerindítás alapvető lépéseit:

1. A hálózati gép elindul vagy újraindul, és felkészül a hálózatról történő rendszerbetöltésre.
2. A hálózati gép szétsugározza a rá jellemző egyedi ethernet MAC-címét, keresve a kiszolgálót.
3. A kiszolgáló, melyet már korábban felkészítettek arra, hogy bizonyos MAC-címeket figyeljen, válaszként az ügyfélgép IP-címét adja vissza. A kiszolgáló egy előre meghatározott IP-tartományból vett IP-címmel válaszol minden olyan kérésre, amely a hálózat fizikai szintjén érkezik.
4. A hálózati csomópontban lévő gép megkapja az IP-címét és felkészíti saját ethernetcsatolóját a TCP-, illetve IP-kapcsolatra.
5. Az imént beállított csatoló segítségével a gép most már elküldheti a rendszermagra vonatkozó kérelmet.
6. A kiszolgáló válaszul az ügyfélnek egy hálózati betöltőt küld, amely a hálózati behúzást támogató rendszermagot tölti majd be.
7. A hálózati betöltő – csak olvasható módban – befűzi a saját állományrendszert.
8. A betöltő beolvassa a kiszolgálótól kapott rendszermagot a helyi gép memóriájába és átadja neki a vezérlést.
9. A rendszermag írható, illetve olvasható módban befűzi a saját könyvtárat, a további állományrendszereket és elindítja az `init`-folyamatot.
10. Az `init` elindítja az adott gépre szabott szolgáltatásokért felelős démonokat, ezzel a rendszer teljesen feláll.

A leírásból látjuk, hogy az ügyfél hálózati gépe több ponton is függ a kiszolgáló válaszaitól. A lényeges pontok: a hálózati indítást támogató rendszermag, a saját állományrendszer, és az a mód, ahogyan a rendszermag és az IP-adat eljut a kiszolgálótól az ügyfélhez.

Az alapok

Ahhoz, hogy egy hálózati gép távolról indítható legyen, a kiszolgálónak és az ügyfélnek jól meghatározott módon együtt kell működnie. Számos alapvető követelménynek szükséges teljesülnie a távoli rendszerindításkor:

- Sokoldalú kiszolgáló: a kiszolgálónak a megfelelő szolgáltatásokat kell futtatnia (ezekről később esik szó), melyek a szükséges tudnivalókkal látják el az ügyfelet.
- Távoli gépindítás: a rendszerindítás megkezdéséhez képesnek kell lennünk, hogy fizikailag elindítsuk vagy újraindítsuk a távoli gépet. Ekkor még nem hagyatkozhatunk az ügyfélen lévő operációs rendszerre, annál is inkább, minthogy olykor éppen az operációs rendszer okozta lefagyás miatt kényszerülünk újraindítani a gépet.
- A hálózat: az összes hálózati csomópontnak el kell tudni érnie a kiszolgálót, akár közvetlenül, akár közvetve egy átjárón keresztül. Ebben a cikkben csak az ethernethálózatokra térek ki.
- Hálózati indítást támogató rendszermag: a rendszermag lehet tömörített vagy tömörítetlen, jelölt vagy jelöletlen. A jelölt rendszermagokra, amelyekről az alábbiakban még esik szó, az ***Etherboot*** nevű eljárás használatakor van szükség. További tudnivalókhoz a forrásokban megadott Etherboot weboldalt is meglátogathatjuk.
- Hálózati betöltő: ez teszi lehetővé a hálózati indítást támogató rendszermagnak a kiszolgálóról történő beolvasását az ügyfél memóriájába.
- Az állományrendszer: a régi szép időkben az állományrendszert a hálózaton keresztül, NFS-alapon szolgáltatták. Az olyan eljárásokkal, mint a SYSLINUX, a teljes állományrendszert egy ramlemezen keresztül szolgáltathatjuk, amit a rendszermaggal együtt küldhetünk el az ügyfélnek.

Most, hogy már nagyvonalakban látjuk, miként zajlik le a távoli rendszerindítás, és tisztában vagyunk vele, hogy az ügyfél és kiszolgáló kapcsolattartásában milyen követelményeknek kell teljesülnie, nézzük meg részleteiben is az egyes szolgáltatásokat!

A gép bekapcsolása távolról

Egy számítógép távolról történő bekapcsolása csak egy kis szelete a távoli alkatrészszintű irányítás lehetőségeinek. Több gyártó is kínál termékei mellé olyan különleges alkatrészeket és programokat, amelyek lehetővé teszik a távolról történő megbízható gépkezelést. Ilyenek például a hőmérsékletfigyelés, a ventilátor- és áramellátás-kezelés, BIOS-frissítés, figyelmeztető jelzések stb. Egy hálózati gép távolról történő indításához ezen alkatrészszintű szolgáltatások közül csupán arra van szükségünk, amely a gép ki- és bekapcsolását lehetővé teszi, bár egy újraindító (reset) szolgáltatás is jól jöhet.
A távoli rendszerindításhoz a munkaállomáson valószínűleg meg kell majd változtatnunk a behúzó adathordozók sorrendjét. A jellemző sorrend általában: hajlékonylemez, CD-ROM, merevlemez. Az esetek nagy többségében a hálózatról való rendszerbehúzás lehetősége vagy nem is szerepel a listában, vagy az összes helyi adathordozó után, a lista legalján található. Mivel a legtöbb munkaállomást már a merevlemezre előre telepített működő operációs rendszerrel szállítják, a gép újraindításakor vagy bekapcsolásakor a rendszer a helyi merevlemezről betöltődik.

Ethernet

A valódi hálózati rendszerindításhoz egy PXE-megfelelő ethernetcsatolóra lesz szükségünk. A PXE a Preboot eXecution Environment nevű eljárásra utal, ami része Intel Wired for Management (WfM) elnevezésű kezdeményezésének. Egy PXE-megfelelő ethernetcsatoló képes arra, hogy betöltse és futtassa a kiszolgálótól kapott hálózati

➲ syslinux.zytor.com

betöltőprogramot, mielőtt még áthozná magát a rendszermagot. Ehhez a művelethez a hálózati gép és az ethernetcsatoló BIOS-ainak együttműködése szükséges. Egy PXE-megfelelő rendszer képes arra, hogy szétsugározza a csatoló MAC-címét, fogadja a kiszolgáló válaszát, beállítsa a megfelelő TCP-értékeket, fogadja a hálózaton keresztül érkező betöltőprogramot és átadja neki a vezérlést.

Hajlékonylemez

Bár a cikk a távoli rendszerindításról szól, a hajlékonylemez pedig helyi adathordozó, mégis említést kell tennünk róla, mert a távoli rendszerindításnak létezik egy nagyon fontos kevert típusa. Mivel sok esetben az ethernetcsatoló és a gép BIOS-a együttesen nem PXE-megfelelő, létrejött egy nyílt forrású kezdeményezés: az Etherboot. Az Etherboot segítségével rendszerindító hajlékonylemezt készíthetünk, amely egy egyszerű betöltőprogramot és egy ethernet-eszközmeghajtót tartalmaz. Állítsuk be a rendszerindító adathordozók sorrendjét úgy, hogy a hajlékonylemez legyen az első helyen, majd így indítsuk el a gépet. Az Etherboot-lemezről a meghajtóprogram betöltődik a memóriába, és elkezdődik a MAC-cím szétsugárzása, valamint a kiszolgáló válaszának figyelése. Egy PXE-megfelelő rendszernek a kiszolgáló egy hálózati betöltőprogramot küld válaszként, míg az Etherboot-eljárás esetén egy hálózati indítást támogató jelölt rendszermagot. A rendszermagot a `mknbi` parancs futtatásával jelölhetjük meg (az `mknbi` parancsról bővebb adatok a ➲ http://www.etherboot.sourceforge.net/doc/html/mknbi.html weboldalon találhatók). A távoli rendszerindítás lényege, hogy a rendszermagot betöltsük a gép memóriájába. Akár a PXE-, akár az Etherboot-megoldást választjuk, az eredmény ugyanaz: a hálózati útvonal gyorsan összeáll, a rendszermag betöltődik a memóriába és megkapja a vezérlést.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Melyek az IP-adatok?

Amikor egy ügyfélgép a hálózatról indul el, akár a PXE-megoldást használva, akár hajlékonylemezről, megkezdi MAC-címének sugárzását olyan kiszolgálót keresve a helyi hálózaton, amelyik a szükséges IP-adatokat majd a rendelkezésére bocsátja. Erre azért van szükség, hogy az ügyfél a megfelelő IP-értékeket az ethernetcsatoló rendelkezésére bocsáthassa, és aztán a további párbeszédet már TCP-, illetve IP-alapon folytathassa. Több módja is létezik, hogy a hálózati gépet ellássuk a megfelelő IP-adatokkal: ezek a RARP, a BOOTP és a DHCP nevű módszerek.

RARP

A RARP-eljárás (Reverse Address Resolution Protocol) a hálózati csatolókártya egyedi 48-bites ethernetcíméhez (a MAC-címhez) rendel hozzá egy IP-címet. Amikor az ügyfél elindul a hálózatról, MAC-címét szétsugározza a vele azonos fizikai hálózaton lévő összes gépnek. Ezen gépek egyikén futnia kell a `rarpd` démonnak, mely a */etc/ethers* állományból kiolvassa, hogy az adott 48-bites címhez melyik IP-címet kell hozzárendelni, majd válaszul ezt a vadiúj IP-címet elküldi az ügyfélnek. Miután megkapta az IP-címét, az ügyfél egy TFTP-kérést (Trivial File Transfer Protocol) indít el, hogy megkapja a megfelelő képállományt (erről később még szót ejtünk). Ennek a megoldásnak legnagyobb hátránya, hogy csak a helyi fizikai hálózaton működik, és csak egyetlen adatot szolgáltat: az ügyfél IP-címét.

BOOTP

A BOOTP (Bootstrap Protocol) -eljárás határozott fejlődés a RARP-hoz képest, mivel tartalmaz hálózati átjárótámogatást (a távoli rendszerindítás útvonalválasztón keresztül is működik), és sokkal több tudnivalót szolgáltat az ügyfél számára. A BOOTP az IP-címen kívül olyan adatokat is elküld az ügyfélnek, mint az átjáró (az útvonalválasztó) címe, a kiszolgáló címe, az alhálózati maszk, valamint a rendszerbehúzó állomány (a tulajdonképpeni képállomány). Tartsuk szem előtt, hogy egy adott alkatrészcímhez egy, és csakis egy IP-cím rendelhető hozzá.
A BOOTP legnagyobb hátulütője, hogy az IP-címeket egy az egy kapcsolatban felelteti meg a MAC-címeknek, vagyis egy adott MAC-címet mindig ugyanahhoz az IP-címhez rendeli hozzá. Ha elgondolkodunk azon, hogy milyen követelmények lépnek fel mondjuk olyan esetekben, ha valaki gyakran dolgozik más irodákban, vagy hordozható számítógépével sokat utazik, akkor látjuk, hogy ez az „egy az egyhez” típusú kapcsolat némileg korlátozó. Az ilyen cégek esetében a felhasználók a számítógépeikkel utaznak, és csak néha van szükségük arra, hogy a központi kiszolgálóhoz kapcsolódjanak, például amikor letöltik a leveleiket. Az idő nagyobbik részében IP-címük kihasználatlanul hever – hiszen nem adható ki másnak –, ami nagy pazarlás. Erre a gondra kínál elegáns megoldást a DHCP-eljárás.

DHCP

A DHCP (Dynamic Host Configuration Protocol) a BOOTP ésszerű utódja. A BOOTP valójában már elavultnak számít, a legtöbb helyen felváltotta a DHCP. Az egyik ok, amiért a DHCP népszerűségében felülmúlta a BOOTP-t, az, hogy az előbbi támogatja a dinamikus IP-cím kiosztást, míg az utóbbi csak állandó IP-címeket tud kiosztani (egy MAC-címet mindig ugyanahhoz az IP-címhez rendeli hozzá). A DHCP által támogatott dinamikus címkiosztás lehetővé teszi, hogy egy adott cím több hálózati gépnek is kioszthető legyen (természetesen nem egy időben). A „mozgékony” cégek alkalmazottjainak példájánál maradva ilyenkor a következők történnének: a gép kapcsolódik a hálózathoz és szétsugározza a MAC-címét. A kiszolgáló, amelyen a `dhcpd` démon fut, már előre elkülönített egy IP-címtartományt e gépek részére, és a tartományban található következő szabad címet egyszerűen hozzárendeli az ügyfélhez. A DHCP az IP-cím kiosztás hosszú élettartamáért is felelős a DHCP-állományon keresztül.
A DHCP számtalan beállítási lehetőséggel bír; részletezésük sajnos túlmutat e cikk keretein. További tudnivalók találhatók a ***Ralph Droms*** és ***Ted Lemon*** által írt The DHCP Handbook című könyvben (Pearson Higher Education, 1999).

A rendszermag és a hálózati betöltő átküldése

Miután a hálózati gép megkapta az IP-címét és alkalmassá tette a csatolót a TCP-, illetve IP-alapú párbeszédre, a gép egy BIOS-eljáráson keresztül elküldi a behúzó képállományra vonatkozó kérelmet. Az IP-cím hozzárendelés és a képállomány elküldésének ez a szétválasztása szándékos; így válik lehetővé, hogy különböző kiszolgálók teljesíthessék ezeket a kéréseket. A TFTP (Trivial File Transfer Protocol) pont megfelelő eszköz a képállomány átvitelének lebonyolítására, mivel nagyobb testvérével, az FTP-vel (File Transfer Protocol) ellentétben az állományok letöltéséhez nem igényel felhasználói bejelentkezést. A TFTP kezdetleges biztonsági szolgáltatása abban merül ki, hogy alapértelmezetten csak a kiszolgáló */tftpboot* könyvtárából küldhet állományokat. Mivel ez a biztonsági megoldás meglehetősen ismert a rendszergazdák körében, ezért a */tftpboot* könyvtárba csak a nyilvánosságnak szánt állományokat rakják. A `tftp-hda` legfrissebb kiadásába már állományszintű biztonsági eljárást is beépítettek.
Figyeld meg, hogy egy képállomány átviteléről beszélünk – ez lehet jelölt rendszermag (Etherboot-eljárás) vagy hálózati betöltőprogram (PXE-eljárás). Ha az Etherboot-, vagyis a hajlékonylemezes rendszerbehúzási megoldást használjuk, akkor a BOOTP vagy a DHCP egy jelölt rendszermagra mutat. Ha igazi PXE-megoldással élünk, akkor a BOOTP vagy DHCP egy hálózati betöltőprogramra mutat. Ez utóbbi esetben a hálózati betöltő a gép memóriájába töltődik és TFTP-n keresztül egy jelöletlen rendszermag letöltését kezdeményezi. A PXE használatához a TFTP-kiszolgálónak támogatnia kell a ***tsize*** nevű beállítási lehetőséget (RFC 1784, RFC 2349). A ***H. Peter Anvin*** által írt tftp-hpa program támogatja ezt a lehetőséget. A program letölthető a ➲ http://www.kernel.org/pub/software/network/tftp címről.

Monolitikus rendszermag

Akár a két lépésben történő PXE-rendszerindítást választjuk (először a hálózati betöltő, majd a rendszermag), akár az egy lépésben lezajló Etherboot-eljárással történő behúzást (csak rendszermag), a végeredmény az lesz, hogy a rendszermag betöltődik az ügyfél memóriájába és megkapja a vezérlést. Talán már felmerült benned, hogy mi is a különbség egy hálózati rendszerindításra készített rendszermag, és egy átlagos, helyi merevlemezről történő indításhoz készített rendszermag között. Az első, amit el kell döntenünk, hogy moduláris vagy monolitikus rendszermagot építünk-e – az utóbbi esetben nem használhatunk betölthető modulokat. Ha már fordítottál rendszermagot, akkor tudod, hogy a beállítás során a különböző szolgáltatásokat különbözőképpen építhetjük a magba: az `Y` beállítással a szolgáltatás a rendszermag része lesz, `N` beállítással a szolgáltatás nem kerül be a rendszermagba, az `M` beállítással a szolgáltatás csak szükség esetén épül be a rendszermagba. Ha monolitikus rendszermagot építünk, az `M` lehetőséggel nem élhetünk.

A rendszermag jelzőbitjei

Számos jelzőbitet kell bekapcsolnunk, ha monolitikus, hálózati indítást támogató rendszermagot szeretnénk építeni. Először a monolitikus rendszermagot építjük meg, amihez a modultámogatást ki kell kapcsolnunk. A rendszermag ***.config*** állományában a következőket látjuk:

```
#CONFIG_MODULES is not set
```

ha monolitikus rendszermagot készítünk, ezt a lehetőséget ki kell kapcsolnunk:

```
CONFIG_MODULES=n
```

Az ext2 állományrendszer támogatására szükségünk lesz – ha ezt az állományrendszert fogjuk használni a helyi ügyfélgépen, vagy befűzni egy távoli kiszolgálóról, mint például az NFS saját állományrendszer esetén:

```
CONFIG_EXT2_FS=y
```

Amennyiben távoli saját állományrendszert kívánunk használni, amit NFS-en keresztül csatolunk, akkor az NFS-támogatásra is szükségünk lesz:

```
CONFIG_NFS_FS=y
CONFIG_ROOT_FS=y
```

Mivel az ügyfél indítására az Etherboot-eljárást fogjuk használni, be kell kapcsolnunk a hálózati beállítások lehetőséget, valamint legalább annak a hálózati csatoló kártyának (NIC) a támogatását, ami az ügyfélgépben található:

```
CONFIG_NETDEVICES=y
CONFIG_EEXPRESS_PRO100=y
```

Végül be kell állítani a RARP-, BOOTP- és DHCP-eljárásokon keresztüli IP-címhozzárendelés lehetőségét:

```
CONFIG_IP_PNP_RARP=y

CONFIG_IP_PNP_BOOTP=y

CONFIG_IP_PNP_DHCP=y
```

Ha a saját állományrendszert ramlemezen keresztül szolgáltatjuk, nem pedig NFS-en keresztül (lásd a következő bekezdést), akkor a RAM-állományrendszer lehetőséget engedélyezni kell:

```
CONFIG_BLK_DEV_RAM=y

CONFIG_BLK_DEV_INITRD=y
```

Moduláris rendszermag

Moduláris rendszermag építésekor az `M` betűvel választhatjuk ki azokat a modulokat, amelyeket a rendszermag csak szükség esetén tölt be. Mivel a modulok nem állandóan beépítettek a magba, kell valamilyen tárolóhely, ahonnan betöltődnek, ha szükség van rájuk. A Linux rendelkezésünkre bocsát ilyen területet, ez a ramlemez. A ramlemez csak a rendszerbehúzás folyamán él, és csak a modulokat tartalmazza. Használhatnánk távoli saját állományrendszer szolgáltatására is, azonban ennek részletezése kívül esik cikkünk keretein.

A moduláris rendszermag készítésének bemutatásához a MINIX állományrendszer-támogatást modulként adjuk a rendszermaghoz. Így a rendszer kezelni képes a MINIX saját állományrendszert, amit hajlékonylemezről töltünk be. A modularitás támogatásához legelőször is a modulok támogatását be kell kapcsolnunk, majd a rendszermagot újra kell fordítanunk:

```
CONFIG_MODULES=y
```

Példánkban a MINIX-támogatást dinamikus modulként valósítjuk meg, így ez a beállítás `M` betűt kap:

```
CONFIG_MINIX_FS=M
```

Miután moduláris rendszermagunkat (`bzImage`) újrafordítottuk és átmásoltuk a */**tftpboot*** könyvtárba, keresnünk kell egy módszert a betöltésre. Egy olyan eszköz, mint például a nyílt forrású SYSLINUX képes rá, hogy betöltse a rendszermagot és létrehozza a ramlemezt. Egy ***pxelinux.cfg*** mintaállomány alapbeállításban körülbelül a következőképpen néz ki:

```
DEFAULT bzImage

APPEND vga=extended initrd=minix.gz
➥root=/dev/fd0 ro

PROMPT 1

TIMEOUT 50
```

A beállitóállományban láthatjuk, hogy a rendszermag neve `bzImage`, a ramlemezé ***minix.gz***, és az ügyfél saját állományrendszer a hajlékonylemezről lesz betöltve (`/dev/fd0`). Létre kell hoznunk a ramlemezt, ami a MINIX-modult (`minix.o`), az `insmod` parancsot (ennek segítségével tölthetjük be a MINIX-modult), a konzoleszközt (***/dev/console***) és a `linuxrc` parancsot tartalmazza, utóbbi a ramlemez használatakor lesz meghívva. (A ramlemezek Linux alatti működéséről mindenre kiterjedő leírást olvashatunk a ***Werner Almesberger*** és ***Hans Lerman*** által karbantartott ***initrd.txt*** állományban, mely része a rpm-formátumban terjesztett rendszermagoknak.) Egyéni ramlemezünk létrehozásához, amely a MINIX-modult majd dinamikusan befűzi a rendszermaghoz, illetve a hajlékonylemezről betölti a MINIX saját állományrendszert, egy állományt kell létrehoznunk, és a `dd` parancs segítségével ki kell nulláznunk:

```
dd if=/dev/zero of=minixroot bs=1k count=4096
```

Ezután az állományt egy hurokeszközhöz (loopback) rendeljük `/dev/loop0`:

```
losetup /dev/loop0 minixroot
```

Létrehozzuk az ext2 állományrendszert:

```
mkfs.ext2 /dev/loop0
```

Az eszközt egy szabadon választott csatolási ponthoz fűzzük, például az */**mnt***-hez:

```
mount /dev/loop0 /mnt
```

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Létrehozzuk a legszükségesebb könyvtárakat:

```
mkdir /mnt/dev /mnt/lib /mnt/sbin
```

Létrehozzuk a konzoleszközt:

```
mkdev /mnt/dev/console c 5 1
```

A `minix.o` modult a */lib* könyvtárba, a ***sash*** és ***insmod*** állományokat pedig a */sbin* könyvtárba másoljuk:

```
cp /usr/src/linux/fs/minix/minix.o /mnt/
➥lib/minix.o

cp /sbin/sash /mnt/sbin/sash

cp /sbin/insmod /mnt/sbin/insmod/
```

Nézzük meg, hogy az `insmod` parancsnak mely rendszerkönyvtárakra lesz szüksége:

```
ldd /sbin/insmod
```

és ezeket másoljuk be a */mnt/lib* könyvtárba:

```
cp /lib/libc.so.6 /mnt/lib

cp /lib/ld-linux.so.2 /mnt/lib
```

Ezután hozzuk létre a `/mnt/linuxrc` parancsállományt, amely a MINIX-modult fogja betölteni:

```
#!/sbin/sash

/sbin/insmod /lib/minix.o
```

Tegyük végrehajthatóvá az állományt:

```
chmod 777 linuxrc
```

Fűzzük ki a */mnt* könyvtárat, és válasszuk le a hurokeszközt:

```
umount /mnt

losetup -d /dev/loop0
```

a `gzip` paranccsal tömörítsük a ***minixroot*** állományt, és ezzel fel is készültünk a gép indítására:

```
gzip minixroot
```

A fenti példával addig a pontig jutottunk el, amikor a rendszermag a saját állományrendszert a `/dev/fd0` eszköztől megpróbálja betölteni. Ahhoz, hogy ez a példa teljesen működőképes legyen, létre kell hoznunk egy MINIX-állományrendszert a hajlékonylemezen, és az eszközt be kell fűzni a rendszerbe. Legalább az `init`, az `sh` parancsokat, illetve az ezekhez szükséges függvénykönyvtárakat át kell másolnunk a lemezre, és létre kell hoznunk a konzoleszközt.

A távoli NFS saját állomány

Amint az köztudott, az élet a saját állományrendszer (/) nélkül értelmetlen. Helyi adathordozóról történő rendszerindításnál a saját állományrendszer majdnem mindig a helyi merevlemezen található. De honnan kapja meg egy távolról indított gép a saját állományrendszerét? Két választásunk adódik: egy távoli kiszolgálótól NFS-en keresztül befűzzük a sajátot vagy ramlemezt használunk. Amennyiben a saját állományrendszert NFS-en keresztül szolgáltatjuk, akkor alapértelmezés szerint a rendszermag a */tftpboot/ip* elérési úton keresi a sajátot, ahol az IP helyén az ügyfélgép IP-címe áll. Ehhez a kiszolgálón kell elindítanunk az NFS-t és exportálnunk kell a */tftpboot* könyvtárat (vagy a */tftpboot/ip*-t minden egyes ügyfélgép számára). Ahhoz, hogy az ügyfélgépen elérjünk a bejelentkezési eljárásig (login prompt), a saját állománynak tartalmaznia kell a következőket: az `init` parancsot, a parancshéj végrehajtható állományait, az eszközállományokat (legalább a konzoleszközt), és azokat a dinamikus függvénykönyvtárakat, amelyek az `init` és a héj parancsainak működéséhez szükségesek. Gyors és nem túl elegáns módja a távoli saját állományrendszer létrehozásának az `init`, `sh`, a szükséges függvénykönyvtárak és a konzoleszköz átmásolása:

```
cp /sbin/init /tftpboot/192.168.64.1/sbin/init

cp /bin/sh /tftpboot/192.168.64.1/bin/sh
```

Az `ldd` parancs megmondja nekünk, hogy az `init`-nek mely dinamikus könyvárakra van szüksége:

```
ldd /sbin/init

ldd /bin/sh
```

Ezeket a könyvtárakat másoljuk be a */tftpboot/ip/lib* könyvtárba. Az eszközök elkészítéséhez használhatjuk az ügyes `MAKEDEV` programot, amely a `MAKEDEV`-csomag része:

```
/dev/MAKEDEV -d /tftpboot/129.168.64.1/dev
➥console
```

Ha a kiszolgálón a többi szolgáltatás megfelelően működik, akkor az ügyfelet távolról indítva az lefuttatja a távoli saját állományrendszer `init` parancsát, ehhez a szintén távoli konzolt használja; elindítja a héjat és várja, hogy megadjunk egy futási szintet (erre azért van szükség, mivel a */etc/inittab* állomány a távoli saját állományrendszerben nem létezik). Az *s* billentyűt leütve egyfelhasználós üzemmódba jutunk, és a héj készenléti jel (prompt) már a rendelkezésünkre is áll. Egy különleges héjat, a `sash`-t (`standalone shell`) is használhatjuk, amely nagyon hasznos távoli környezetben. Ez annak köszönhető, hogy nem igényel dinamikusan befűzött

Könyvek

Internetworking with TCP/IP, Volume 1, 4th Edition, szerzője *Douglas E. Comer*. Prentice-Hall, 2000. IBSN 0-13-018380-6.
Linux Core Kernel Commentary, szerzője *Scott Maxwell*. Coriolis Open Press, 1999. ISBN 1-57610-469-9.
Linux in a Nutshell, 3rd Edition, szerzője *Ellen Siever et. al*. O'Reilly & Associates, Inc., 2000. ISBN 0-596-00025-1.
TCP/IP Illustrated, első rész, szerzője *W. Richard Stevens*. Addison-Wesley, 1994. ISBN 0-201-63346-9.

függvénykönyvtárakat, valamint beépített parancsokat tartalmaz állományrendszerek kezelésére (`mount`, `umount`, `sync`), állományok hozzáférésének megváltoztatására (`chmod`, `chgrp`, `chown`), archiválásra (`ar`, `tar`) és egyéb műveletekhez. Az `sh`-héj használata helyett a ***sbin/sash*** állományt átmásolhatjuk a ***/tftpboot/ip/sbin/sash*** állományba, a rendszermag így a `sash`-héjat fogja elindítani. Egy lecsupaszított ***innitab*** állományt is felhasználhatunk a `sash` indítására, ahogy ezt az alábbi példa mutatja:

```
id:1:initdefault:
1:1:respawn:/sbin/sash
```

Összegzés

A cikkben szó esett néhány olyan szolgáltatásról és módszerről, melyek segítségével távolról is elindíthatunk egy linuxos gépet. A távoli Linux-felügyelet rendkívül termékeny talaj a további kutatások számára. Ahogy a hálózatok egyre gyorsabbak, egyre több távoli ügyfelet támogatnak, és ahogy a számítógépfürtök mérete egyre nagyobb lesz, és egyre inkább központi felügyeletet igényelnek, úgy fognak a távoli Linux karbantartási módszerek egyre nagyobb szerephez jutni az iparban. A tömörített kiszolgálóeljárások megjelenésével a távoli Linux-felügyelet nemcsak kényelmi szempontokat szolgál, hanem egyenesen nélkülözhetetlenné válik.

Köszönetnyilvánítás

Hálás vagyok a ***Vasilios Hoffman (Wesleyan)*** kutatásaiért. „V”, ahogy ő szereti hívatni magát, bemutatta, hogy a hurokeszközök hogyan használhatók ramlemezek létrehozására, és hogy távoli rendszerindításokhoz miként hozhatunk létre jól működő moduláris rendszermagot. „V” a Linuxszal kapcsolatos tudnivalók két lábon járó lexikona.

Richard Ferri
az IBM Linux Technológiai Központjának vezető programozója. Olyan Linux-alapú számítógépes fürtök megvalósításán dolgozik, mint a LUI (➲ http://www.oss.software.ibm.com/lui) és az OSCAR. New York északi részén él feleségével, Pattel, három tizenéves fiával, és kétes fajtájú kutyáival.

Kapcsolódó címek

Etherboot-honlap ➲ etherboot.sourceforge.net
Az Etherboot-eljárás a hálózati csatoló beállítására hajlékonylemezt használ, így lehetővé válik a hálózaton keresztüli rendszerindítás.
Távoli Linux-indítás mini-HOWTO
➲ cui.unige.ch/info/pc/remote-boot/howto.html
SYSLINUX-honlap
➲ syslinux.zytor.com – *H. Peter Anvin* kiváló munkája.
A tftp-hpa oldala
➲ http://www.kernel.org/pub/software/network/tftp – egy újabb kiváló program, szintén H. Peter Anvintól.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

VPN-átjáró telepítése

Hogyan kell telepíteni és üzemeltetni IPSec-alapú tűzfallal ellátott VPN-átjárót mindössze egyetlen rendszerindításra alkalmas hajlékonylemezre telepített Linux-változattal?

A VPN olyan eszköz, amely lehetővé teszi az adatok biztonságos átvitelét még az olyan megbízhatatlan hálózatokon is, mint amilyen az Internet. A VPN-t gyakran arra használják, hogy helyi hálózatokat (LAN) az Internet révén széles körű, vagyis nagy kiterjedésű hálózattá (WAN) egyesítsenek. Lehetséges, hogy két iroda között VPN-t kell kiépítenie, de arról már nincs meggyőződve, hogy a vállalati szintű VPN-megoldásokkal együtt járó magas kialakítási költségek is indokoltak-e.

A LAN-ok számára tervezett alkalmazások – amelyek például a hálózati állománymegosztást használják – működése a WAN-csatlakozást követően alapvetően leértékelődik. Hasonlóképp a WAN-kapcsolatok sávszélessége és hosszabb válaszideje kedvezőtlenül befolyásolja a megbízhatóságot, a csoportkezelést és a vékonyügyfélprogram működését. Ezenkívül lehet, hogy Ön otthonról, a dolgozószobájából szeretne nagy sebességű internetelérése révén zökkenőmentesen és biztonságosan kapcsolódni cége belső hálózatához, IPSec-útválasztón keresztül. De lehet, hogy éppen csak érdeklődik a VPN és az IPSec iránt és szeretné őket kipróbálni – cikkünk Önnek szól.

Írásunkban bemutatott VPN-tűzfalak képesek bármilyen 486-os vagy annál erősebb számítógépen működni, amelybe 16 MB vagy annál több memóriát, valamint kettő, Linuxszal együttműködő ethernet hálózati kártyát építettek.

Alapgondolatom az volt, hogy a felhasználók kezébe olyan kiindulópontot adjunk, amellyel egyedüli, önálló csomagból mindenki készíthet magának nagyméretű, biztonságos, méretezhető és pontosan beállítható VPN-eket, amelyek ráadásul más közönséges, kereskedelmi forgalomban kapható VPN-ekkel is képesek együttműködni. Ha kevés karbantartást igénylő tűzfal–VPN-átjáróval kíván kísérletezni, az itt bemutatásra kerülő programcsomag eszményi lehet az Ön számára. Jelen írásunkban azt mutatjuk be, hogyan telepítsünk a legkevesebb költséggel VPN-átjárót, amely az IETF (Internet Engineering Task Force) IPSec ajánlását használja.

Az IPSec olyan nyílt szabvány, amelyet szinte minden ismertebb tűzfalprogram és alkatrészgyártó, mint például a Lucent, a Cisco, a Nortel és a Check Point támogat. Ez a csomag széles körű együttműködésre képes IPSec-et tartalmaz, amely valódi 3DES-titkosítást és MD5-hitelesítést használ a végpont-webhely, illetve a végpont-végpont jellegű VPN-ekben. Mindezt jó lenne úgy megvalósítani, hogy ne kelljen teljes Linux-változatot igénybe venni, vagy az IPSec-modult befordítani a rendszermagba.

Az általunk itt megvizsgált VPN a FreeS/WAN-en alapul (➲ http://www.freeswan.org), amely az IPSec-ajánlás nyílt forrású, hordozható megvalósítása. Bemutatták, hogy a FreeS/WAN különböző mértékű együttműködésre képes a Cisco IOS 12.0 és az annál fejlettebb útválasztókkal, Nortel Contivity kapcsolókkal, OpenBSD, Raptor Firewall tűzfallal, Check Point FW-1, SSH Sentinel VPN 1.1, F-Secure VPN, Xedia Access Point, PGP 6.5/PGPnet és későbbi változatokkal, IRE SafeNet/SoftPK, Freegate 1.3, Borderware 6.0, TimeStep PERMIT/Gate 2520, Intel Shiva LanRover, Sun Solaris és Windows 2000 rendszerekkel.

A FreeS/WAN hivatalos honlapján megfelelési listáját a Világhálón elérhető leírással együtt rendszeresen frissíti. A letölthető csomagban a FreeS/WAN 1.5 változata szerepel. A Linux Router Project (➲ http://www.linuxrouter.org) alapján összeállítottam egy egyetlen hajlékonylemezből álló változatot, amely a tűzfal alapbeállításait telepíti. A tömör Linux-változat egyetlen rendszerindításra alkalmas hajlékonylemezen elfér. Ez a lemez tulajdonképpen ***Charles Steinkuehler*** Eiger-lemez képállománya, Steinkuehler által készített IPSec-megfelelő rendszermaggal és az LRP IPSec programjával együtt.

A tűzfal kiépítéséhez a Linux-változatokban megtalálható IP Chains programot használtam. A felhasznált változat a 2.2.16 Linux-rendszermagra épül és a DUCLING mozaikszavas elnevezést kapta, tehát Diskette-based Ultra Compact IPSec Gateway (hajlékonylemez-alapú ultratömör IPSec-átjáró).

A tömör Linux-változatok kanyargós utat jártak be. Szakmai szempontból az LRP ***Dave Cinege*** tömör változatára hivatkozik. Több változat is forgalomban van, beleértve ***Matthew Grant*** immár „halott" Eiger-változatának Charles Steinkuehler által készített újabb változatát (EigerStein). További hasonló változat a ***David Douthitt*** által készített Oxygen (➲ http://leaf.sourceforge.net/content.php?menu=900&page_in=1). A fentieken kívül létezik még a LEAF-vállalkozás (Linux Embedded Appliance Firewall), amely ernyő módjára fogja össze a programfejlesztőket, és megkísérli összehangolni a programkibocsátásokat és -leírásokat, vagyis afféle „mindent egy helyen beszerzési hely" a tömör Linux-változatok számára (➲ http://leaf.sourceforge.net). A továbbiakban az LRP mozaikszót használom az alább bemutatott tömör Linux-változatra való hivatkozáshoz, annak ellenére, hogy sokan talán helytelennek találják az elnevezést.

Amennyiben MS Windows 9x rendszert használ, az önkibontó állomány a kicsomagolás után egy szabványos, nagy sűrűségű hajlékonylemezre telepíti magát. De Linux alatt a képállományt rendszerindításra alkalmas hajlékonylemezre is rá lehet másolni. Amint a lemez elkészült, indítsuk el róla a rendszert, majd végezzük el a beállítási állományok szerkesztését. Ezzel a munka el is van végezve: nincs szükség a lemezfelosztások formázására vagy a merevlemezre telepített betöltéskezelő programokkal való bogarászásra. Amennyiben mégsem lenne elégedett az eredménnyel, távolítsa el a hajlékonylemezt a meghajtóból és indítsa újra a gépet.

Keresse fel az alábbi helyet a Világhálón:
➲ http://leaf.sourceforge.net/devel/thc, ahol bővebben olvashat a lehetőségekről.

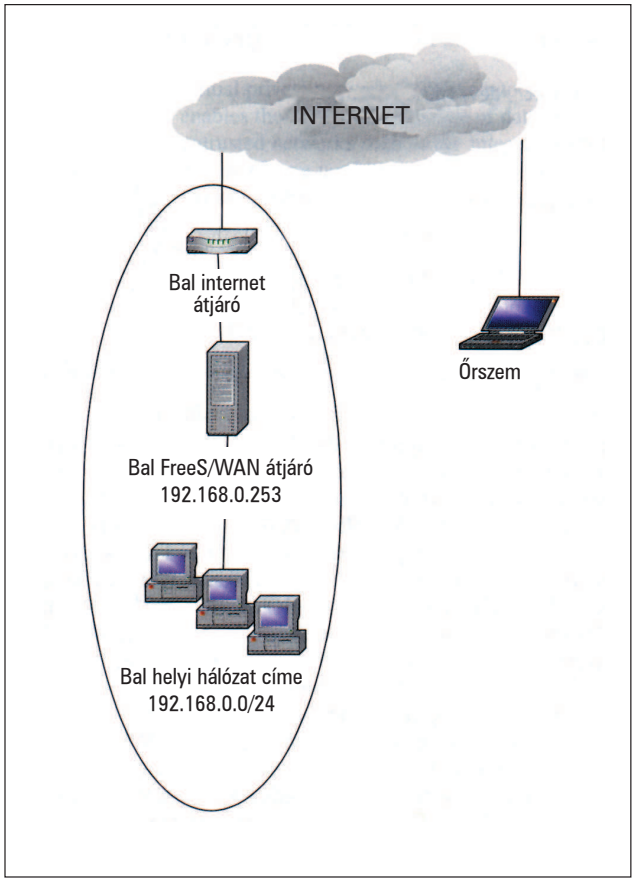
Háttérműveletek a tűzfalon és a VPN-ben

Az LRP itt bemutatott változata szabványos IP Chains-alapú tűzfalra épül. Az IP Chains (a 2.4.x sorozatú rendszermagban már IP Tables) szabadon terjeszthető csomagszűrő Linux-rendszerekhez (lásd még ***David A. Bandel*** A Netfilter

megszelídítése című írását a Linuxvilág 2001. októberi számának. Sokat lehet tanulni az IP Chains HOGYAN-oldalairól is, amennyiben valaki még járatlan e tűzfalkezelő eszköz beállításában. Ez utóbbi leírást a
➲ http://www.Linuxdoc.org/HOWTO/ipchains-howto/html
címen olvashatja el). A VPN IPSec-megvalósítását a FreeS/WAN készítette el, amely megfelel az IETF IPSec-ajánlásának. Az IPSec tulajdonképpen az Internet Protocol (IP) kiterjesztése, amely gondoskodik a hitelesítésről és a titkosításról. Ez utóbbi két feladatot három protokoll látja el, nevezetesen az ESP (Encapsulated Security Payload), az AH (Authentication Header) és az IKE (the Internet Key Exchange). Valamennyi rendszeralkotó szerepel a FreeS/WAN IPSec-megvalósításában és általában átlátható a végfelhasználók számára. Az ESP és az AH protokoll kezeli a titkosítási és hitelesítési feladatokat, az IKE pedig a kapcsolati jellemzőket közvetíti, többek között a kezdeti jellemzők beállítását, a titkosítási kulcsok kezelését és megújítását. Jelenleg a FreeS/WAN egyetlen titkosítási sémát támogat, a „triple Data Encryption Standard"-szabványt, azaz rövidítve a 3DES-szabványt – amely jelenleg *a* Szabvány az IPSec-titkosításban.
A hitelesítés az úgynevezett megosztott titkokból (megosztott kulcs) készült MD5-kivonat alapján történik. A megosztott kulcsok lehetnek kölcsönösen elfogadott jelsorozatok, RSA titkosítási kulcspárok vagy X.509 igazolások, illetve nyugták. A FreeS/WAN KLIPS (kernel ipsec, vagyis a rendszermagba fordított IPSec-modul) üzembe helyezi az AH és ESP protokollokat, valamint a csomagok kezelését. Az IKE-folyamatok kezelik a kulcsegyeztetéseket, a kulcsfrissítéseket pedig a FreeS/WAN önállóan működni képes `pluto` démonja végzi.

Rendszerigények és telepítés

A telepítés megkezdéséhez mindenek előtt egy hajlékonylemez-meghajtóval – jómagam csak 3,5 hüvelykes hajlékonylemez-meghajtókat használtam – ellátott számítógépre lesz szükség, amelybe két hálózati kártyát is építettek. Az LRP Linux-változat telepítési igényei csekélyek, nem szükséges hozzá erőteljes gép. A célnak bármilyen Intel 486-os vagy annál erősebb gép megfelel, amelybe 8 MB vagy annál nagyobb memóriát építettek.
A munkához szükséges két hajlékonylemez, amelyeknek megbízható, nagy sűrűségű lemezeknek kell lenniük, de akár lehetnek például az AOL reklámhordozó lemezei is. Soha semmilyen gondom nem akadt a megszokott (generic) hajlékonylemez-meghajtókkal, viszont az Imation USB U2 SuperDisk-meghajtójával a hajlékonylemez formázása során előfordultak írási hibák.
Töltsük le a megfelelő ***DUCLING.tgz/zip***-változatot az
➲ ftp://ftp.cinemage.com/pub webhelyről, majd csomagoljuk ki az állomány tartalmát. Ha a gépe állandó IP-címmel van ellátva, töltse le az állandó változatot, ellenben dinamikus IP-cím esetén a DHCP-változatra lesz szüksége. Ha a gépén Windows 9x rendszert használ, töltse le a ***ducling-stat-W9x-1-0.zip*** vagy a ***ducling-dyn-W9x-1-0.zip*** állományt. A tömörített állomány WinZippel való kibontását követően létre fog jönni a ***ducling-stat-1-0.exe***, illetve a ***ducling-dyn-1-0.exe*** állomány, valamint a könyvtári modulok. Az ***.exe*** önkicsomagoló állomány, amely a hajlékonylemezt formázza, majd felírja rá a képállományt (image). Futtassa a ***ducling-stat-1-0.exe*** vagy a ***ducling-dyn-1-0.exe*** programot és helyezzen egy hajlékonylemezt a meghajtóba. Ne felejtse el, hogy a lemezen lévő összes adat el fog veszni.
Abban az esetben, hogy ha Ön MS-DOS-t vagy Windows 3.1-et használ, először az állandóan a memóriában tartózkodó `FDREAD.EXE` segédprogramot kell a DOS-ba betölteni, ha 1722 KB formátumú hajlékonylemezre kíván írni, illetve ilyenről olvasni. Az `FDREAD.EXE` ***Christopher H. Hochstätter*** ingyenes segédprogramja. Ha számítógépén Linuxot használ, töltse le a fent említett állományokat, majd a `tar` programmal végezze el az állományok kicsomagolását. Az alábbi



A webhely gyalogos beállítása

példában DHCP-megfelelő, dinamikus címfeloldást támogató programot mutatunk be:

```
tar xvfz ducling-dyn-1-0.tgz
```

Ezután írjuk fel a ***ducling-dyn-1-0.img*** képállományt a formázott hajlékonylemezre a Linux `fdformat`, illetve `dd` parancsai segítségével:

```
fdformat /dev/fd0u1722
dd if= ducling-dyn-1-0.ima of=/dev/fd0u1722
```

Ha a Linux a hajlékonylemez létrehozását a fent leírt módon elvégezte, máris rendelkezik egy rendszerindításra használható lemezzel. A ***zipfile/mappa*** nevű modulok tartalmazzák a hálózati kártyák meghajtómoduljait a tűzfalmaszkoláshoz szükséges kiegészítő modulokkal együtt. Másolja a ***zipfile/mappa*** tartalmát egy további MS-DOS formátumúra formázott hajlékonylemezre mostani bemutatónk beállításához (lásd alább). Linuxban ezt az alábbi két paranccsal tehetjük meg:

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

```
fdformat /dev/fd0
```

ezután pedig

```
mkdosfs /dev/fd0
```

A `mount` paranccsal tegyük a hajlékonylemezt a rendszer számára hozzáférhetővé, és másoljuk át a modulokat. Olvassa el a ***README*** állományban mellékelt leírást, amely a tűzfal, illetve útválasztó beállításának részleteit tartalmazza. Amennyiben az összes kívánt csomagot kapacitáshiány miatt nem lehet egyetlen hajlékonylemezre összesűríteni, meg kell vizsgálni a két hajlékonylemezes telepítés lehetőségét – a DUCLING-változat README-állományából erről is kaphatunk tájékoztatást – azonban a betölthető CD-ROM avagy kicsi merevlemez elkészítésének leírását is megtalálhatjuk. Bővebb tájékoztatás végett keresse fel az LRP-leírásnak a Világhálóra feltett oldalait.

Az LRP rendszerindító hajlékonylemezek – a meglepő igazság

Talán meglepődik azon, hogy az LRP DOS-formátumú hajlékonylemezeket használ. Valószínűleg még jobban meglepődik, amikor felfedezi, hogy DUCLING-változat 1722 KB-os képállományként telepíti magát a lemezre. A 3,5 hüvelykes, nagy írássűrűségű hajlékonylemez műszakilag 2 MB kapacitású adathordozó, s ezt adatot a lemezen is feltüntetik: 2 MB „nyers" vagyis formázatlan kapacitás. Az 1440 KB-os formázott tárolókapacitás csupán a hagyományos lemezformátum eredménye, amelyben az adathordozóra 80 sávot írnak fel és 18 szektort sávonként. Megfelelő eszközökkel olyan hajlékonylemezeket hozhatunk létre, amelyeken 80 szektor és szektoronként 24 sáv van kijelölve, ez összesen 1920 KB-os kapacitást jelent. Az LRP-változatokhoz általában 1680 KB-os formátumú hajlékonylemezeket használnak és a sávfelírás biztonságosnak tűnik. A fentebb említett formátumokon kívül az 1722 KB (82, 21), 1743 (83, 21), és 1760 (80, 22) lemezformátumokat is használják. Az 1722 KB-os lemezformátumot a próbák során elég megbízhatónak találtam, és mindeddig nem tapasztaltam olyan hibákat, amelyekről beszámolhatnék. Egészen 1920 KB-os méretig hoztam létre és használtam nagy kapacitásúra formázott hajlékonylemezeket. A szokásosnál nagyobb kapacitásúra formázott hajlékonylemezek hajlamosak alkalmatlanná válni a rendszerindításra, nyilvánvalóan a számítógépek BIOS-a és a lemezen lévő szabványostól eltérő szektorméret ellentétéből fakadóan. Állítólag az 1680 KB-nál nagyobb kapacitású lemezek akár alkatrészfüggőségi gondoktól is szenvedhetnek. A Windows NT és a Windows 2000 rendszerekről az a hír járja, hogy az 1680 KB-nál nagyobb kapacitású lemezeknél a lemezre történő írás során megbízhatósági gondok jelentkeznek. Az MS Windows 9x operációs rendszerek az alapértelmezésnek megfelelő beállítások megváltoztatása nélkül képesek a hajlékonylemezt a szokásosnál nagyobb kapacitásúra formázni. Linux-rendszerekben gyakran szükséges, hogy a `mount` parancs kiadásakor a tényleges formátumot is megjelöljük, vagyis például `/dev/fd0u1722`, ahol az `fd0` a 0 (nullás) hajlékonylemez-meghajtót jelöli, míg az `u1722` az 1722 KB-os lemezformátumot. A Linux szabványos hajlékonylemez-meghajtója az alapértelmezés szerint `/dev/fd0u1440`, tehát 1440 KB-os formátumú. A nagy kapacitású lemezek készítésével és ilyenek használatával kapcsolatos tanácsok végett tanulmányozza a ***Paul Batozech*** készítette LRP-rendszerindító lemez HOGYAN-leírását. Szóban forgó témánkon túl számos

A FreeS/WAN telepítési kapcsolat listája

```
conn Listing for the Setup Shown in Figure 1
conn sentinel-vpn
       type=tunnel
#BiztonsÆgi Ætjær , amely m g tt hÆl zat van
#a k vetkezı ugrÆs errefelØ van.
   type=tunnel
   left=1.2.3.4
   leftnexthop=1.2.3.5
   leftsubnet=192.168.0.0/24
   right=0.0.0.0
        rightnexthop=
        rightsubnet=
        keyexchange=ike
         keylife=8h
         keyingtries=0
         pfs=no
         authby=secret
         auto=add
```

más érdekességre is bukkanhat a ➲ http://leaf.sourceforge.net/devel/thc címen. Az MS Windowshoz ***Gilles Vollant*** készítette el a WinImage-et (➲ http://www.winimage.com), amit különösen hasznosnak és felhasználóbarátnak találtam.

Az olyan Linux-eszközökhöz képest azonban, mint amilyen az `fdformat`, `mkdosfs` és a frissebb szuperformázó-alkalmazások, ez több szempontból is korlátozott program. Az itt vizsgált, MS Windows számára létrehozott önkicsomagoló állományok a WinImage program segítségével készültek.

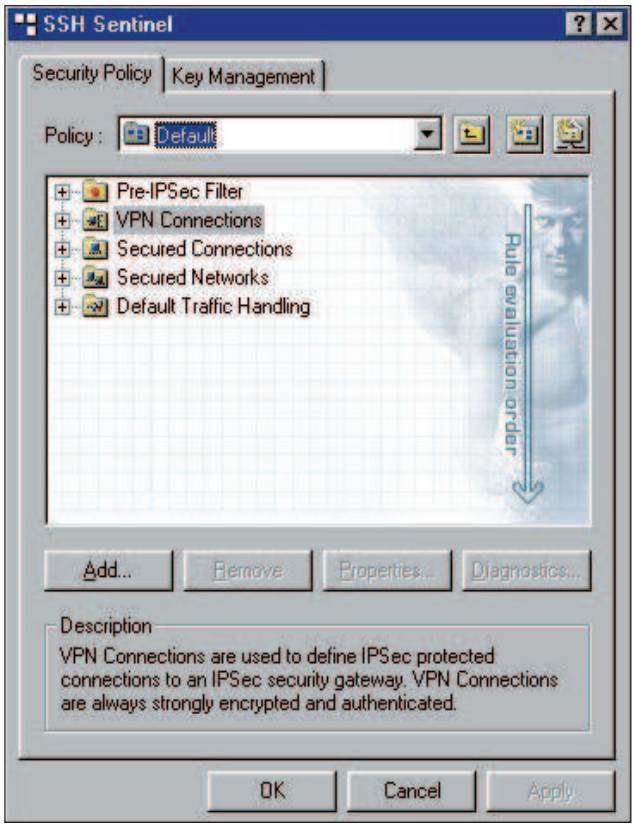
Hogyan történik az LRP-változat betöltése?

Mielőtt elkezdene dolgozni az LRP-vel, hasznos megfigyelni, hogyan is működik a rendszer. Ha a rendszerindító lemezt veszi szemügyre, egy sor állományt fog látni: ***ldLinux.sys***, ***Linux***, ***sysLinux.cfg***, ***root.lrp***, ***modules.lrp***, és a ***local.lrp*** nevű állományt.

Az ***ldlinux.sys*** tölti be a csizmahúzó szerepét, vagyis ez az az állomány, amely gondoskodik a rendszermag – a Linux nevű állomány – és a kezdeti ***root.lrp*** csomag memóriába töltéséről. A rendszermag megkezdi működését, létrehoz egy ramlemezt és ide bontja ki a ***root.lrp*** állományt. A ramlemez ezúttal nem más, mint egy tárterület, amelyet a program lemezrészként foglal le. Más szóval a rendszermag ezután teszi elérhetővé a `mount` paranccsal a ***sysLinux.cfg***-ben meghatározott rendszerindító készüléket. Az indítólemezen levő maradék ***.lrp***-csomagok kibontása a ***syslinux.cfg*** állományban előírtaknak megfelelően történik, majd azok is betöltődnek a ramlemez területére.

Az ***.lrp***-csomagok szabványos Unix-típusú ***tar***-állományok, vagyis `tar` és `gzip` programmal tömörített archívumok. Ha az ***.lrp***-csomagok már telepítve vannak a ramlemez könyvtárfájába, a rendszer megkezdi a betöltési folyamatot a szabványos Linux ***rc***-állományban meghatározott betöltési sorrendnek megfelelően.

Az LRP egyszerűen szólva maga a lecsupaszított rendszermag betölthető modulokkal és egyéb ***.lrp*** formátumba csomagolt programokkal együtt. Az LRP valódi Linux; általában véve, ami képes működni közönséges Linux-rendszeren, az bizonyosan képes lesz az LRP-lemezről is elindulni.



1. kép A Sentinel Biztonsági házirend

Gyakran az LRP-alkalmazások és -szolgáltatások kiterjesztésének akadálya a hajlékonylemez szabta méretkorlát. Amennyiben Ön további szolgáltatásokat igényel, például távoli felügyeletet SSH-n keresztül, DNS-kiszolgálót és egyebeket, talán a több hajlékonylemezes, CD-ROM-alapú, vagy teljes merevlemezt igénylő változatot is meg szeretné majd tekinteni.

VPN-útválasztó, illetve tűzfal telepítése és beállítása

A rendszerindító hajlékonylemez létrehozása után győződjön meg róla, hogy a hajlékonylemez be lett-e abba a számítógépbe helyezve, amelyiken a VPN-tűzfalat üzemeltetni kívánja. Ellenőrizze, hogy a rendszerindítás hajlékonylemezről történik-e. A VPN-tűzfal indításakor a gép képernyőjén meg fogja látni az LRP üdvözlőképernyőjét, a Linux betöltőprogram üzeneteit, majd a bejelentkező parancsot.

Ha eddig eljutott, gratulálunk! Sikeresen telepített egy LRP-változatot. Most már elkezdheti beállítani az LRP-tűzfal jellemzőit úgy, ahogyan az a programhoz mellékelt leírásban is szerepel. A tűzfal beállítása után a VPN-t is be kell állítani. A programhoz járó DUCLING-leírás megadja a részhálózat-részhálózat beállításának részleteit. Ebben a leírásban található az IPSec hitelesítési módjának beállítása (***/etc/ipsec.secrets***), az IPSec-hálózat beállítása (***/etc/ipsec.conf***), és a tűzfal az 500-as kapura (UDP), illetve az 50 és 51-es kapura vonatkozó hozzáférés engedélyezési szabályokkal együtt.

Fontos megjegyezni, hogy nem szükséges állandó IP-cím a VPN-kapcsolatok használatához. A „gyalogos" beállítást a következő fejezetben mutatjuk be, ahol a VPN-ügyfél nem meghatározott állandó IP-címmel rendelkezik. Működtettem VPN-eket dinamikusan kiosztott IP-címekkel ellátott géppárok között. A DHCP által a gépekhez rendelt IP-címes VPN-ek kezelése akkor válik bonyolulttá, ha mindkét kiosztott IP-cím gyakran változik meg. A következő fejezet a „gyalogos" beállítás lehetőségeit mutatja be a DUCLING- és Microsoft-alapú IPSec-ügyfél között.

Példa az együttműködésre

Az alábbi példánk az MS Windows 9x, illetve 2000 ügyfélprogram által létesített végpont-webhely kapcsolatot mutatja be, amely az SSH Communication Sentinel 1.1 (nyilvános béta3-változat) nevű programot használja. A FreeS/WAN az IPSec-megvalósítások bő választékával képes együttműködni. A telepítési folyamat bonyolultsága és a számítási teljesítmény nagysága termékenként változik.

Számos 3DES-, illetve MD5-titkosítást támogató termék az IKÉ-n keresztül képes együttműködni a FreeS/WAN-nel. Másrészt arra a megállapításra jutottam, hogy az erős titkosítást támogató, összes tulajdonsággal rendelkező IPSec-megvalósítások jogtiszta beszerzése roppant fárasztó, különösen akkor, ha Ön, az olvasó, az Egyesült Államokon kívül él.

Számos alkatrészgyártó kínál IPSec-megoldásában korlátozott lehetőségeket. Például az egyik termék csak a gyenge titkosítást támogatja, a másik pedig esetleg a VPN-szolgáltatásokat a szállításra korlátozza. Fontos az IPSec-en keresztül biztosított két VPN-üzemmód megkülönböztetése: a továbbítási és alagút üzemmódé. Továbbítási szállítási feladatokat lát el és hitelesítést végez a két végpont között. Az alagút üzemmód inkább részhálózatok összekapcsolására használható és lehetővé teszi a részhálózatok elérését a tűzfalon és útválasztón keresztül. Alapjában véve a továbbítási mód a forgalmat a végpont-végpont közötti kapcsolattartásra korlátozza. Az alagút üzemmód megengedi a végpont-végpont, végpont-részhálózat és részhálózat-részhálózat adatcseretípusokat.

Úgy tűnik, létezik legalább egy alkatrészgyártó, amelyik nem engedi meg, hogy IPSec-megoldása állandó IP-című kapcsolatra fusson. Úgy látszik, hogy az SSH Sentinel termék (➲ http://www.ipsec.com) egyik fentebb említett gondtól sem szenved, valószínűleg abból a tényből fakadóan, hogy a cég székhelye az Egyesült Államokon kívül van.

Letöltöttem és kipróbáltam a Sentinel 1.1 beta3 harmincnapos próbaváltozatát, és a Windows 98-cal működő asztali számítógépen nagyon könnyűnek találtam a beállítását. A Sentinel leírása a FreeS/WAN VPN-átjáróval kapcsolatos beállítási példákat is tartalmaz.

Az alábbiakban olvasható a „gyalogos beállítás" összefoglalója, amely dinamikusan kiosztott IP-címmel rendelkező távoli felhasználók számára lehetővé teszi, hogy csatlakozzanak a tűzfal mögötti helyi hálózathoz.

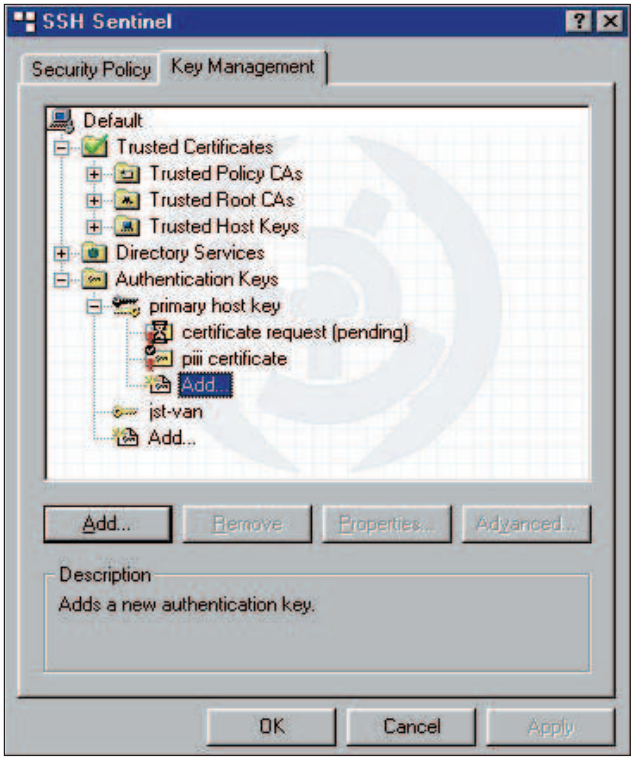
Szükség lesz az 50 és 51-es kapu (TCP) és az 500-as kapu (UDP) megnyitására a dinamikus IP-cím, illetve az internetszolgáltató DHCP-címtartománya számára. *Ábránkon* (35. oldal) az alapvető telepítést láthatjuk. Szükség lesz a DUCLING FreeS/WAN tűzfalon a ***/etc/network.conf*** állomány szerkesztésére: lépjen be az ***lrcfg***-be, válassza az egymás után következő menükben mindig az első pontot, végül a beállítást:

```
eth0_IP_SPOOF=NO/
```

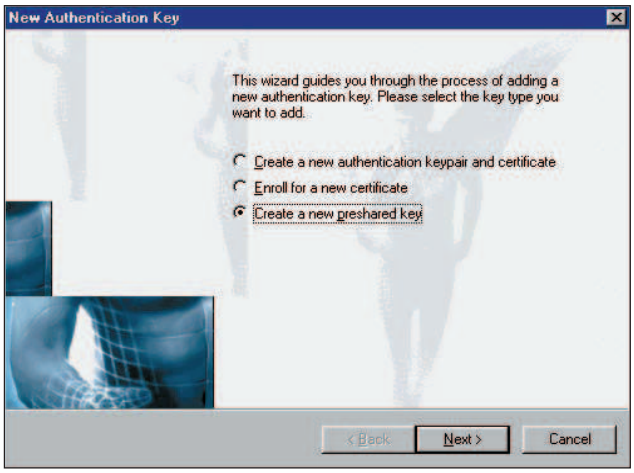
az alagút üzemmódba irányított csomagok letiltásának kikapcsolásához. A programhoz mellékelt leírás részletes útmutatással szolgál arra nézve, hogyan kell ezeket a feladatokat elvégezni. A FreeS/WAN ***ipsec.conf*** állományának tartalmát a *listánkon* tekinthetik meg. Az ide vonatkozó ***ipsec.secrets*** állományban pedig az alábbi bejegyzés szerepel:

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva



2. kép Új kulcs hozzáadása

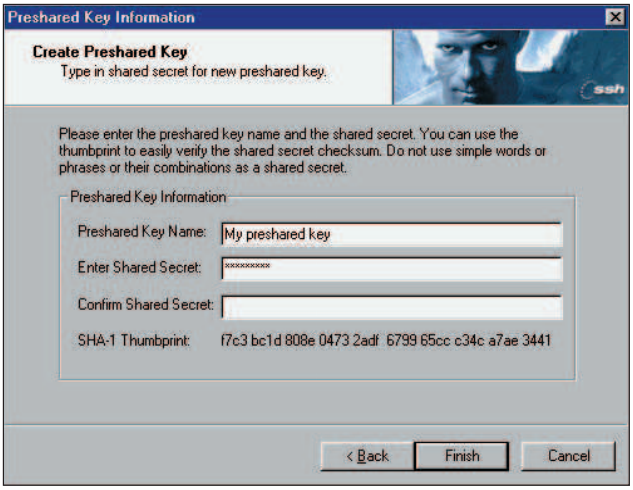


3. kép Előre megosztott kulcs beállítása

```
1.2.3.4  0.0.0.0  :PSK Ide kell  rni a
gyalogos beÆll tÆs jelszavÆt
```

ahol az idézőjelek között megadott jelsorozat a megosztott titok jelkombinációja. A 0.0.0.0 formában megadott IP-cím bármilyen IP-címet jelölhet, a `rightsubnet` és a `rightnexthop` értékek üresen hagyása végpont–részhálózat-típusú kapcsolatra utal. A Sentinel IPSec-szolgáltatás üzembe helyezéséhez az alábbiakat kell elvégezni:

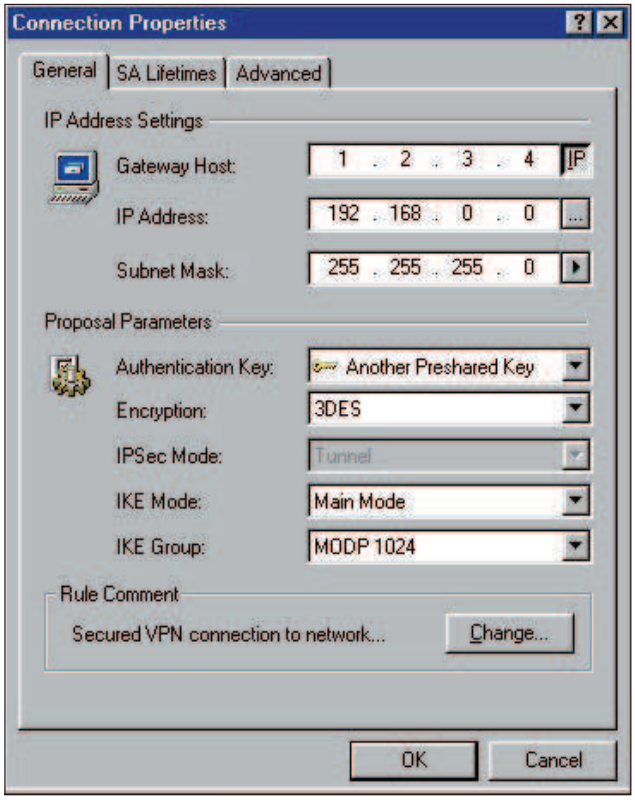
1. Töltse le az SSH Sentinel programot a ➲ http://www.ipsec.com címről, és a telepítés során kövesse az útmutatóban leírtakat.
2. Lépjen be a Sentinel program ***Sentinel Policy Manager***-be (házirend-kezelőjébe): lásd a *1. képet*.



4. kép „Megosztott titok" begépelése

5. kép Kulcs begépelése

3. Válassza a ***Key management*** fület (Kulcskezelés) és a ***Authentication*** keys (Hitelesítő kulcsokat), majd nyomja meg az ***Add*** (Hozzáadás gombot): a miként *2. kép* mutatja.
4. Hozzon létre új, előre megosztott kulcsot (***Create***), majd bökjön rá a ***Next*** (Következő) gombra (*3. kép*).
5. Gépelje be saját előre megosztott kulcsát – idézőjelek nélkül. Ennek meg kell egyeznie a megosztott titok jelsorozattal, amelyet a */etc/ipsec.conf*-ban adott meg (*4. kép*).
6. Nyomja meg a kész gombot.
7. Az SSH Sentinel házirend-kezelője főkonzolának Biztonsági házirend menüjében válassza a VPN-kapcsolatokat, azon belül pedig az ***Add***-ot (Hozzáadás).
8. Gépelje be a távoli átjáró IP-címét és nevét: példánkban ez 1.2.3.4, és válassza az előzetesen megosztott „titkot", amelyet hitelesítési kulcsként az ötödik lépésben hozott létre (*5. kép*).
9. Válassza a 3DES-titkosítást: a ***Main mode*** (Fő üzemmódot) és a ***MODP 1024***-et az IKE-üzemmód és az IKE-csoport számára.
10. Állítsa be az IKE élettartamát, vagyis az ismételt kulcskiosztások közötti időszak hosszát ugyanarra az értékre, amely az ***ipsec.conf*** állományban szerepel; ez általában 480 perc, azaz 8 óra.



6. kép A „Properties" (Tulajdonságok) menüfül a VPN-kapcsolatoknál

Mentsen minden beállítást és próbáljon meg pingelni egy tűzfal mögötti gépet, vagyis próbálja ki a belső csatlakozófelületet, a 192.168.x.254-es címet. Ekkor a kapcsolatnak létre kell jönnie. Próbálja futtatni a Sentinel hibakereső programját, hogy azzal ellenőrizze, a kapcsolat valóban létrejött-e. Meg kellett állapítanom, hogy a hibakereső program néha előidézheti a FeeeS/WAN–Windows-kapcsolat meghibásodását. Amennyiben ez történik, a FreeS/WAN-átjárón indítsa újra az IPSec-et és élessze fel az egyes kapcsolatokat. Itt hadd hívjuk fel ismét a figyelmet arra, hogyha a kapcsolatot újra kell indítania, az LRP-t futtató gépbe jelentkezzen be és az IPSec-elemek újraindításához írja be:

```
#etc/initd.d/ipsec restart
```

Úgy találtam, hogy a Windows 2000 Professionalban – de a Windows 98-ban nem – az útválasztó táblát a DOS-konzolból a megosztott részhálózatra kézzel kell módosítani:

```
route ADD 192.168.0.0 MASK 255.255.255.0 1.2.3.4
```

Ellenőrizze a Microsoft `route` parancsát a leírásban.

Összefoglalás

Jelen cikkünk egyetlen 3,5 hüvelykes hajlékonylemezről indított tűzfalas VPN-átjáró kialakításának eszközeit vázolja fel. Mindössze egyetlen hajlékonylemez révén adott a lehetőség, hogy számítógépeket és változatos elrendezésű hálózatokat az Internet segítségével biztonságosan kapcsoljunk össze.
A DUCLING-változat a vázig lecsupaszított Linux-változat. Amennyiben meggyőződött arról, hogy a FreeS/WAN VPN valóban képes az igényeinek megfelelni, akkor vagy egy

A leaf honlapja

teljesebb LRP-változatot választ, vagy egy teljes Linux-változatot választ olyan feladatok megoldására, mint amilyen például a távoli elérés, a biztonságos héjprogram, az SSH vagy a DNS-kiszolgáló.

Duncan Napier
vezeti a kanadai North Vancouverben (British Columbia) a Napier System Research vállalkozást, amely hálózati és informatikai tanácsadással foglalkozik.

Kapcsolódó címek a hibakereséshez

Az LRP beállításával kapcsolatos gondok megoldása végett keresse fel a SourceForge LRP webhelyét ➲ http://leaf.sourceforge.net. *Richard Onanian* honlapja a ➲ http://leaf.sourceforge.net/devel/thc a hibakereső adatok és HOGYAN-ok szempontjából hasznos. Ha a hibakeresést a FreeS/WAN-nál kell végezni, akkor tanulmányozza a termék leírását a ➲ http://www.freeswan.org címen.
Amennyiben a kérdéseire nem kapna kielégítő választ a fentebb említett forrásokból, vizsgálja át a LEAF ➲ http://www.geocrawler.com/lists/3/SourceForge/7325/0 és az LRP ➲ http://www.geocrawler.com/lists/ 3/Linux/303/0 levelezési listák archívumait.
Ha a gondja közvetlenül a FreeS/WAN VPN-nel kapcsolatos, akkor a FreeS/WAN archívumait olvasgassa ➲ http://lists.freeswan.org/mailman/listinfo.
Amennyiben az említett gond még mindig makacsul ellenáll a megoldási kísérleteknek, küldje el őket a megfelelő levelezési listákra.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Mondd, te kit választanál?

Minden bizonnyal számos felhasználóban felmerült a vágy, hogy egyszer ő is telepítse a Linuxot ezt a méltán népszerű operációs rendszert.

Most induló sorozatunkban nekik kívánunk segíteni annak bemutatásával, hogy miben is különbözik a Linux a Windowstól, és miként férhetnek el mégis egymás mellett egy számítógépen. A kezdő felhasználókat továbbá megismertetjük a legfontosabb műveletekkel, beleértve a Linux telepítését is.

Amikor a Linuxot és a Windowst összehasonlítjuk egymással, az első dolog, amit mindenképpen meg kell említenünk: míg a Linux mindenki számára hozzáférhető, ingyenes, nyílt forráskódú operációs rendszer, addig a másikat csak borsos áron szerezhetjük be „jogtisztán". A Linux nemcsak magán-, hanem üzleti célokra is szabadon használható. Ez a mai világban nagyon csábító, mivel ha végzünk egy kis fejszámolást, rájöhetünk: a gépeinkhez szükséges programokra fordított összeg csaknem ugyanannyi, mint amennyit a vasra költöttünk. Nem szabad elfeledkeznünk arról sem, hogy általában nem csak az operációs rendszert kell megvennünk. Igaz, ez a legfontosabb program a számítógépen, de a különböző felhasználói programok (például az irodai csomagok, a grafikai alkalmazások stb.) nélkül nem sok mindenre használhatjuk. Ezeknek az alkalmazásoknak az ára nem ritkán a többszöröse is lehet az adott operációs rendszerének.

A hálózati programok terén a helyzet még ennél is siralmasabb. Attól függően, hány ügyfelet szeretnénk hálózatba kapcsolni, a hálózati operációs rendszerért (amely a kiszolgálógépen fog futni) akár egy-másfélmillió forintot sem szégyellnek elkérni.

Ha azonban a Linuxot (vagy valamelyik társát) választjuk, számos felesleges költségtől kímélhetjük meg magunkat, és még csak át sem hágjuk a különböző jogszabályokat.

Mint említettük, az operációs rendszeren kívül felhasználói alkalmazásokra is szükségünk van. Nos, ma már kevés kivételtől eltekintve Linux alatt is mindenféle program a rendelkezésünkre áll, amelyek nagy része szabadon felhasználható és ingyenes. A korlátozottan felhasználhatók – amelyek pénzbe kerülnek – között is akad olyan, amelyiknek magáncélú (otthoni) felhasználása engedélyezett.

A Linux nem csak otthon vagy az irodai munkában állja meg a helyét. Legnagyobb erőssége ma is hálózati támogatása. Szabadon és ingyenesen beállíthatjuk a legkülönfélébb hálózatokba – akár kiszolgálóként, akár ügyfélként. Sőt, a különböző próbák alapján megállapíthatjuk: a Linux mind üzembiztosságban, mind gyorsaságban felveszi a versenyt pénzért kapható „vetélytársaival".

Ha a Linux szabadon használható és ingyenes, miért éri meg a programozóknak a rendszer fejlesztése? Legtöbbjük tulajdonképpen teljesen ingyen dolgozik rajta, pusztán szórakozásból, amolyan hobbiként. Akadnak tehát olyanok, akik ezeknek az alkalmazásoknak a fejlesztését tűzték ki életcéljukként.

E cikk olvasása közben sokakban felmerülhet a kérdés, hogy ha a Linux mindent tud, amit egy operációs rendszernek tudnia kell, ráadásul ingyen és szabadon felhasználhatóan, akkor miért nem söpörte le idáig a fizetős rendszereket a piacról? Hogy ezt a kérdést megválaszoljuk, először rá kell világítanunk egy-két dologra.

A Linux úgynevezett Unix-alapú operációs rendszer. A Unix-rendszerek a Windowsnál és a DOS-nál is sokkal régebbiek, kialakulásuk a hetvenes évekre tehető. Abban az időben a személyi számítógép (PC) fogalma még mindenki számára ismeretlen volt, ezért az „egyszerű számítógép-felhasználó" fogalma sem létezett. Aki ilyen masina közelébe kerülhetett, az vagy programozó vagy foglalkozását tekintve hozzá hasonló személy volt. A rendszerek tervezésénél tehát a könnyebb kezelhetőség egyáltalán nem szerepelt a szempontok között, mivel a programok kezelői maguk is programozók voltak. A Unix fejlesztését még e szemlélet jegyében kezdték el.

Maga a grafikus felhasználói felületnek az ötlete csak jóval később merült fel a fejlesztőkben, amikor a PC-k megjelenésével a számítógépeket egyre szélesebb körben is használni kezdték, főképpen olyan emberek, akik már nem voltak programozók.

A grafikus felület és a Windows megjelenése számos könnyebbséget hozott a felhasználók számára, mivel a számítógépek használata jóval egyszerűbb és „áttekinthetőbb" lett. A Unix nem követte ezt a fejlődési irányt, mivel akkoriban kizárólag nagygépes rendszereken alkalmazták. Ez részben ma sem változott, a legtöbb Unix-alapú rendszert máig is szuperszámítógépeken, illetve nagyteljesítményű kiszolgálókon használják. Gondoljunk csak a Sun által gyártott kiszolgálókra, amelyeket a szintén Unix-alapú Solarisszal szállítanak. Azóta természetesen Unixra is fejlesztettek grafikus felületet, de a rendszer alapjaiban nem sokat változott.

A 80-as évek végén és a 90-es évek elején kezdték el először igazán a Unix-alapú rendszert fejleszteni, illetve a meglévőket a PC-kre is átültetni. Gondoljunk csak a Minixre, erre a kis Unix-rendszerre, amelyet ***Andrew. S. Tannenbaum*** fejlesztett a diákjai számára. Ebben az időben fogott hozzá ***Linus Torvalds*** is saját rendszerének, a Linuxnak a fejlesztéséhez a célból, hogy Unix-alapú, szabadon használható és ingyenes, teljes értékű operációs rendszert fejlesszen PC-re. Itt kezdődik tehát a Linux története.

De kanyarodjunk vissza kiinduló kérdésünkhöz: miért nem tudta megszerezni a Linux az elsőbbséget az operációs rendszerek piacán? Az egyik legfőbb ok az, hogy a Linux sokkal kevésbé „felhasználóbarát", mint a Windows. Ez utóbbi olyan rendszer, amelyet kifejezetten arra a célra fejlesztettek ki, hogy a lehető legkevesebb számítástechnikai ismerettel rendelkező felhasználó is könnyedén elboldogulhasson vele. A Linux azonban a Unix-rendszerek összes tulajdonságát örökölte, és a Unix köztudottan nem egyszerű kezelhetőségéről híres. Például az összes rendszerbeállítást szöveges állományokban tárolja, ha tehát valamit meg szeretnénk változtatni, azt szövegszerkesztő segítségével saját kezűleg kell megtennünk. Mindezt „tetézi", hogy a szövegfájlok felépítése

ugyan logikus, de néhol nehezen áttekinthető és bonyolult.
Mindezek ellenére a Linux barátságosabb külsőre való formálása gőzerővel zajlik. El kell ismerni, hogy néhány Linux-terjesztésnél (lásd később) a telepítés és a rendszer használata még a tapasztalatlanabb felhasználóknak sem okoz sok fejfájást. Ez azonban nem más, mint a „durva fapados" belsőre ráhúzott, csillogó-villogó külső héj.
A telepítés után előre beállított, kész Linuxot kapunk. Ezekben a rendszerekben számos beállítást anélkül megváltoztathatunk, hogy ki kellene lépnünk például a grafikus rendszerből, vagy akár egyetlen parancsot is be kellene írnunk. Ha azonban többre fáj a fogunk, például valamit át akarunk szervezni, sajnos mélyebben bele kell ásnunk magunkat a Linux szövegfájljainak rejtelmeibe. A rendszer hatékony használatához tehát elengedhetetlenek olyan ismeretek, amelyeket egyedül, segítség nélkül nehezen sajátíthat el az ember.
Ha egyszer eljutunk oda, hogy a felhasználók bármit meg tudjanak csinálni a Linuxukkal anélkül, hogy egyetlen parancsot is beírnának vagy bármit átszerkesztenének valamilyen szövegfájlban, akkor elképzelhető, hogy a Linux minden további nélkül lesöpri a Windowst a pályáról. Ettől azonban még messze van, és nem lesz könnyű elérnie, főleg azért, mert a Unix a Windowsétól gyökeresen eltérő filozófia alapján működő rendszer.
Most sokan biztosan azt gondolják, hogy a Unix elavult rendszer, és csodálkoznak is talán, mit keres még most is itt, a 21. században. Nincs igazuk, ugyanis a Unix igen nagy számú előnnyel bír. Kétségtelenül erős hátránya a nehézkes beállítás, de cserébe teljes testreszabhatóságot nyújt a felhasználók számára.
Mit is jelent ez pontosan? Nehéz szavakkal visszaadni: aki huzamosabb ideig használ valamilyen Unix-rendszert, előbb-utóbb rá fog érezni. A Unix-rendszerek mindenképpen lehetőséget nyújtanak a felhasználóknak ahhoz, hogy rendszerüket úgy alakítsák ki, ahogyan az számukra a legkedvezőbb.
A testreszabhatóságra a legjobb példa talán a grafikus rendszer. A Linux grafikus rendszerének nincs egységes kinézete: csupán azt adták meg, hogy a rendszer miként kezelje az ablakokat, a menüket és a többi grafikus elemet. De hogy miként nézzen ki a munkaasztal, hogyan fessenek az ablakok, milyen legyen a képernyő felépítése, nincs meghatározva. Ezt az úgynevezett ablakkezelők (WindowManagerek) mondják meg, amelyekből több száz létezik, és amelyekből a felhasználó kedvére választhatja ki a neki legjobban tetszőt. Így minden linuxos munkaállomás teljesen eltérő látványt nyújthat.

Vegyünk egy másik példát! A Linux nyílt forráskódú rendszer, tehát a rendszermag forráskódja bárki számára elérhető. Ez lehetővé teszi, hogy az egész rendszermagot újrafordíthassuk, és az általunk fordított rendszermagot használhassuk. Ennek két előnye is van: egyrészt a kész rendszermag a saját processzorunkkal lesz a leghatékonyabb, ami gyorsabb futást eredményezhet. Másrészt kihagyhatjuk belőle a felesleges támogatásokat: ha például nincs SCSI-vezérlőnk (és jó ideig nem is lesz), az egész SCSI-támogatást kivehetjük, így csökken a rendszermag mérete.
Mikor hallottunk mi a Windows esetében olyasmiről, hogy a rendszer magját kisebbre és gyorsabbra cserélhetjük le? (Nem kell megijedni, a rendszermag fordítása a Linuxban teljesen hétköznapi műveletnek számít, amelyhez szükségtelen különösebb programozási ismeret).
Hozzá kell tennünk: nem csak a Linux az egyetlen ingyenes Unix-rendszer. Számos hasonló program található a piacon, a leghíresebb talán a FreeBSD.
E két rendszer használatában eléggé sok a közös vonás (hiszen mindkettő Unix-alapú), sőt a legtöbb alkalmazás mindkét rendszerre megtalálható.
A Linux azonban egyvalamiben különbözik a FreeBSD-től: igazából csak egy rendszermag, semmi több. Ez a rendszermag együttműködik a többi segédprogrammal, alkalmazással, de azok hivatalosan nem a Linux részei. A FreeBSD-nél más a helyzet, mert létezik mögötte egy szervezet, amely a rendszer magjához „hozzáigazítja" az alapkönyvtárakat, a felhasználói alkalmazásokat stb. A Linuxnál ilyen nem létezik, csak a rendszermag fejlesztését fogják össze.
Ezért jöttek létre az úgynevezett „Linux-kiadók". Ezek üzleti vagy nem nyereségközpontú vállalkozások, amelyek a Linux-rendszermag mellé beszerzik a többi alkotórészt, majd egymáshoz „hangolják" őket, végül kibocsátják a felhasználók számára.
Minden Linux-változatnak más a célja: a SuSE és a Mandrake például a kezdő felhasználókat célozza meg, a Debiannál inkább a telepítés közbeni testreszabásra helyeződik a hangsúly. Ezért nem mindegy, hogy ki melyik Linux-változatot választja, hiába tartalmazza ugyanazokat az összetevőket.
Ezeknek a változatoknak a többségét a boltokban meg lehet vásárolni. Magyarországon a Red Hat, a SuSE, a Mandrake és a Turbolinux a legelterjedtebb. Áruk változó, 12–50 ezer forint körül mozognak (attól függően, mit tartalmaznak). Mindegyikben találunk 4–6 CD-t, amelyek linuxos alkalmazásokkal és leírásokkal vannak tele, továbbá egy vagy több nyomtatott kézikönyvet, amelyből az adott Linux-változat használatát sajátíthatjuk el. A Debian a kivétel, mivel nem nyereségközpontú vállalkozás, így saját maga nem dobozolja termékeit – a CD-ket viszont pár dollárért megrendelhetjük.
Azokat a korongokat, amelyeken maga a rendszer található és a leírásokat is tartalmazza, általában ingyen az Internetről is letölthetjük, tehát nem kell azért fizetnünk, mert nem kereskedelmi változatot használunk. Ha viszont megvesszük a dobozos változatot, gyakran olyan alkalmazásokat is találhatunk benne, amelyek nem terjeszthetők szabadon, azaz nem is másolhatók.
Kezdetnek talán ennyi elég lesz. A következő részben részletesebben is kitérünk a Linux és a Windows közötti különbségek taglalására.

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.

© Kiskapu Kft. Minden jog fenntartva

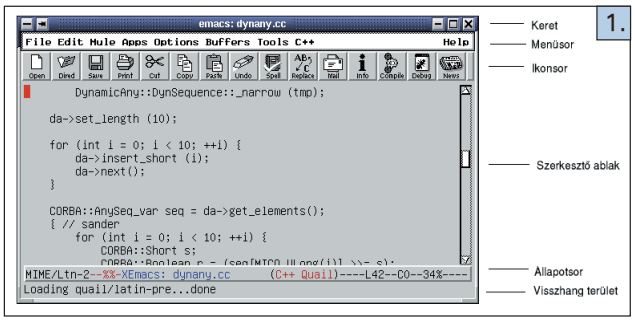
© Kiskapu Kft. Minden jog fenntartva

Az Emacs (2. rész)

Sorozatunk e részében az Emacs-munkaterületet mutatjuk be bővebben.

Elsőként egy terminálemulátorban indítsuk el az XEmacsot az `xemacs` begépelésével, majd a CTRL+ALT+F1 billentyűvel lépjünk át karakteres üzemmódba, és ott adjuk ki az `emacs` parancsot. Azonnal észrevesszük, hogy a karakteres változatot nem veszi körül keret, ugyanis ez az üzemmód az Emacs hőskorát idézi, hiszen a programot akkor kezdték el fejleszteni, amikor még nem léteztek a manapság megszokott grafikus felhasználói felületek, és a számítógépek nem tudtak egyszerre több – a mai értelemben vett – ablakot megnyitni. Nem volt még Windows, csak később lőn világosság, ezért az Emacsban az ablakok (Windows) a képernyőn elkülönített munkaterületeket jelentik. Ha az Emacsot X-kiszolgáló nélkül a parancssorról indítjuk, akkor ma sem tudunk kereteket megnyitni. A keret (frame) szintén ablak, azonban kerete van. Áthelyezhető a munkaasztalon, összecsukható, átméretezhető és ikonokat tartalmazó címsorral bír, bár e tulajdonságok megléte erősen függ az általunk kedvelt ablakkezelőtől. Minden különálló keretben újabb ablakokat nyithatunk meg, és amikor az Emacsból kilépünk, ezek a keretek is bezáródnak. A Unix világában máig erősen él ez a régi időkben gyökerező megkülönböztetés, mert például a Java Fejlesztő Készletében (JSDK) a Jwindow-összetevő a keret nélküli ablak, a JFrame pedig a fent leírt keret.

Az ablakokban és a keretekben az átmeneti tárak (buffer) tartalma jelenik meg. Egy átmeneti tárat akár több ablakban is szerkeszthetünk egyidejűleg. Amikor bezárunk egy ablakot, csak a munkaterületet tüntetjük el, az átmeneti tár tartalma nem veszik el, bármikor előhívhatjuk. Ha bármelyik ablakban vagy keretben mentjük egy átmeneti tár tartalmát, máshol nem kell ismételten mentenünk, mivel egy állományhoz csak egy átmeneti tár van hozzárendelve, bár több ablakban vagy keretben is megtekinthetjük vagy szerkeszthetjük egyidejűleg. Az átfedett részeken láthatjuk, hogy a változások azonnal megjelennek a másik ablakban is.



1.

A keret alatt, a munkaterület legfelső sorában a menüsor található (lásd az *1. képet*). Alatta az ikonsor, majd a szerkesztőablak(ok), ahol a szerkesztési munkát végezhetjük. Minden megnyitott ablak legalsó sorában egy státuszsor vagy Emacs-néven állapotsor (*Mode Line*) található, ami az ablakban lévő átmeneti tár állapotát mutatja. Például a képen látható állapotsoron legelöl az éppen érvényes nyelvi kódolást olvashatjuk („MIME/Ltn-2”), ami a közép-európai betűkészletet jelenti. Az ezután következő karakterek mutatják, hogy módosítottuk-e az állományt:

- --: az átmeneti tárat még nem változtattuk meg,
- **: az átmeneti tár módosult, azaz a szöveget szerkesztettük,
- %%: az átmeneti tárba behívott állomány csak olvasható,
- %*: az állomány csak olvasható, az átmeneti tár tartalma mégis módosítva lett.

Az „XEmacs:” kiírás után az átmeneti tár neve következik. Ezután zárójelben a főmódról (major mode) és az almódról (minor mode) kapunk adatokat, ami a példában a C++ és a Quail. Ilyen főmód lehet például az alap- (Fundamental), a szöveges (Text), a Lisp- vagy a C-mód. Többféle főmódból választhatunk, de egy átmeneti tárban egyidejűleg csak egy főmód lehet működő. Más tárakban más főmódokat adhatunk meg. A főmódok megváltoztatásakor kissé másként viselkedik a szerkesztő, amit az almódok ismét tovább módosíthatnak. Az ***L*** betű után a pillanatnyi sor, a ***C*** betű után pedig az jelenlegi oszlop száma következik. Legvégül megtudhatjuk, hogy hozzávetőlegesen hol járunk a szövegben. Az 1-es képen a szöveg 34 százalékánál lévő résznél járunk, de ha rövid állományról van szó, és a teljes szöveg belefér az ablakba, akkor nem százalékos kiírást kapunk, hanem az „All” üzenet jelenik meg. Hosszú állományok esetén a „Top” szó azt jelzi, hogy az átmeneti tár elején, a „Bot” (azaz bottom) pedig azt, hogy a végén vagyunk.

A legalsó soron a visszhangterület (echo area) osztozik a kisméretű átmeneti tárral (minibuffer). Ez a terület alapértelmezetten egysoros, de ha az egeret a felső elválasztóvonalra visszük, akkor felfelé megnagyobbíthatjuk a méretét. Ebben a sorban jelennek meg a különböző üzenetek, és ide gépelhetjük be az Emacs-parancsokat. A hibaüzeneteket és figyelmeztetéseket kellemetlen hang kíséri, ami a tájékoztató üzenetek esetében elmarad. Az itt felbukkanó szövegeket a program egy *Messages* nevű átmeneti tárba menti, hogy később tanulmányozhassuk őket, ha szükséges. A hosszú ideig dolgozó parancsok az üzenet végére három pontot tesznek, majd a „done” (elkészült) kiírással jelzik, hogy végeztek.

A billentyűkombinációk

A C+X+C+F megnyomásával hívjunk be egy fájlt. A továbbiakban minden a cikkben említett parancs- vagy billentyűkombináció az XEmacs 21.1 változatra vonatkozik, de mindig érdemes kipróbálni azokat az `emacs`-paranccsal indítható másik változattal is, ami nálam a GNU Emacs 20.7.1 volt. A ***C*** betű helyett a CTRL szót is írhattam volna, de minden Emacs-leírásban a nagy ***C*** betűt használják a CTRL, az ***M*** (Meta) betűt pedig az ALT helyett. Ha az ALT nem úgy viselkedik, ahogy elvárjuk, próbálkozzunk helyette az ESC vagy a Windows zászló billentyűkkel. A CD-melléklet Magazin/Emacs/emacs/refcard könyvtárában található egy

refcard.dvi fájl, ami referenciakártya az Emacsban használatos 0billentyűkombinációkról. A kártyán láthatjuk, hogy a két billentyű megnyomását igénylő szolgáltatások általában a szerkesztéssel kapcsolatosak, a négy mozdulatot igénylő, C+X előtagúak valamilyen állománymű veletre, a C+H előtagúak pedig a súgóra vonatkoznak. A „négy mozdulat” kifejezés nem egészen pontos, hiszen a fájl megnyitására használatos C+X C+F billentyűkombináció valójában csak három mozdulatot jelent, hiszen a ***C*** azaz CTRL billentyűt nem feltétlenül szükséges közben felengedni, tehát elég lett volna annyit írni, hogy CTRL+X+F. Mégis célszerű ragaszkodni az Emacsban megszokott jelölésmódhoz, hiszen a program megkülönbözteti az olyan beviteli eseménysorozatokat, amelyek képesek előhívni egy parancsot, és az olyanokat, amelyek előtagként szerepelnek. Az előbbieket teljes billentyűknek, az utóbbiakat pedig előtagbillentyűknek nevezzük. Ha csak a C+X előtag billentyűt ütjük le, az Emacs a visszhangterületen kiírja a „C+x+” üzenetet, és türelmesen vár a folytatásra. A második kötőjel jelzi, hogy nem teljes billentyűről van szó – a C+X előtagot tehát ki kell egészíteni. Egy teljes billentyű lehet egyetlen, de akár négy vagy öt beviteli esemény kombinációja is: például a HOME billentyű megnyomása önmagában lefuttatja a sor elejére ugrás parancsot, de a C+X 4 előtag összekapcsolása a C+O beviteli eseménnyel a C+X 4 C+O teljes billentyűt eredményezi. Az Emacs mindig a beírt karaktereket értékeli ki, és nem érdekli, hogy miképpen vittük be azokat, például az angol billentyűzeten a C+X = billentyűkombinációt kell megnyomnunk, hogy megkapjuk egy kurzor alatti karakter adatait, de a magyar billentyűzeten a C+X SHIFT+7 szolgál ugyanerre.

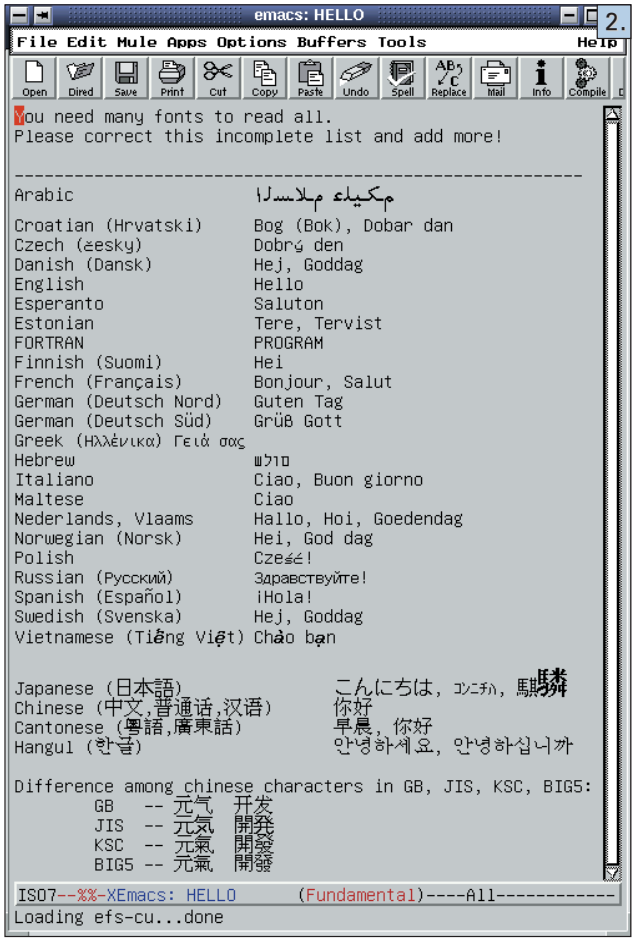
A parancsok

A parancsnevek begépelését mindig az ***M+x*** előtaggal kezdjük, majd az ENTER beviteli billentyűt is meg kell nyomni, hogy a parancsok lefussanak. Például az ***M+x font-lock-mode RET*** parancs működésbe hozza az adott nyelvre jellemző kiemelési módot, vagy éppenséggel megszünteti, ha működő volt. A ***RET*** a ***return*** rövidítése, lényegében ugyanaz, mint az ENTER. Ha elfelejtettük, hogy a szerkesztett állomány melyik könyvtárban van, az érvényes munkakönyvtárat az ***M+x pwd RET*** parancscsal írattathatjuk ki, és az ***M+x cd RET*** paranccsal léphetünk át egy másikba. A teljes billentyűk parancsnevekkel helyettesíthetők, hiszen minden teljes billentyűhöz hozzá van kötve egy parancs, azonban fordítva ez nem feltétlenül igaz. Nyilvánvaló, hogy nem a parancsnevek hossza határozza meg azt, hogy egy parancshoz tartozik-e teljes billentyű, hanem az, hogy milyen gyakran van szükség a parancs használatára. Például a kijelölt terület első betűjét nagybetűssé alakító ***M+x capitalize-region*** parancsot ritkán használjuk, ezért nincsen hozzárendelt teljes billentyű. Ha kíváncsiak vagyunk Emacs-változatunk érvényben lévő kötéseire, a ***C+h b*** kilistázza az egymáshoz rendeléseket. Az Emacs fejlesztői minden esetben a teljes billentyűt részesítik előnyben a parancsbegépeléssel szemben. Kiadhatjuk éppenséggel az ***M+x find-file-read-only-other-frame RET*** parancsot, ami az állományt csak olvasásra egy új keretben nyitja meg, de ilyenkor az Emacs megdorgál bennünket, és figyelmeztet, hogy ez a parancs a C++X 5 R billentyűkombinációhoz van kötve.

Az angol nyelvű Emacs

Amikor a fájlmegnyitási parancsot előhívó C+X+C+F teljes billentyűt gépeljük, az állapotsor alatti kisméretű átmeneti tárban megjelenik a beütött billentyűkombináció visszhangja, majd a „Find file: ~/” szöveg. Látjuk, hogy saját ***Home*** könyvtárunk az alapértelmezett (ennek rövidítése a ~ (hullám – tilde) karakter). Ide kell begépelni a keresett állomány nevét. Ha nemlétező fájl nevét adjuk meg, az Emacs létrehozza azt. Ha a könyvtár is új, megkérdezi, hogy készítsen-e ilyet:

```
The directory containing
/home/ratio/rde/exam.cc does not exist.
Create? (yes or no)
```

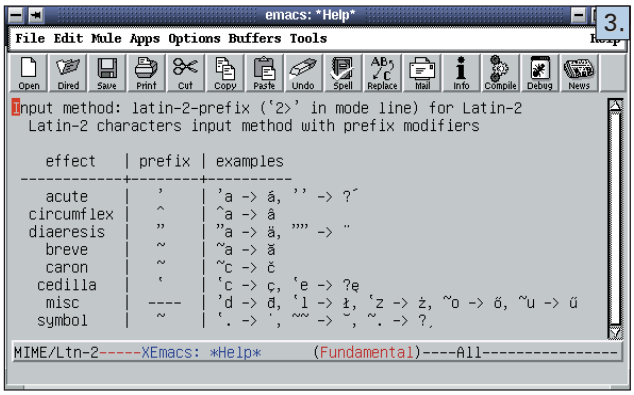


Látjuk, hogy az Emacs nyelve az angol. Én nem találtam hozzá magyar nyelvű segédletet, bár lehetséges, hogy létezik. A magyar nyelvű billentyűzettel is nehézségek jelentkeztek, alapértelmezetten nem lehet beírni a hosszú ***ő*** és ***ű*** betűket, bár az Emacs jól ismeri fel a gondot, hiszen a mini átmeneti tárolóban rögtön kiírja, hogy „odoubleacute” vagy „udoubleacute not defined”. A magyar billentyűzetkiosztáson bajos az olyan Emacs-karakterek használata, amelyek az ALT+GR megnyomásával íródnak be. Például a C+X [és C+X] lapozó utasítások két zárójelére a magyar billentyűzeten ráül az ő és az ú, emiatt a CTRL+X ALT+GR+[, valamint a CTRL+X ALT GR+] kombinációkat kell begyakorolni. Ez több mozdulatot igényel, és zavart okozhat, ha a gombokat nem jó ritmusban nyomjuk. Vannak tehát, akik szerint az Emacs használatakor célszerű angol billentyűzetre váltani. Ilyenkor viszont úgy érezhetjük magunkat, mintha visszamentünk volna a múltba. Hajdan az angol billentyűzetre kellett ráírni a magyar betűket, most pedig megfordítva kellene megtennünk ugyanezt, hogy megtaláljuk őket. Ha viszont valaki a saját képére formálja az Emacs billentyűit, és jelentősen átalakítja őket, akkor kiteszi magát

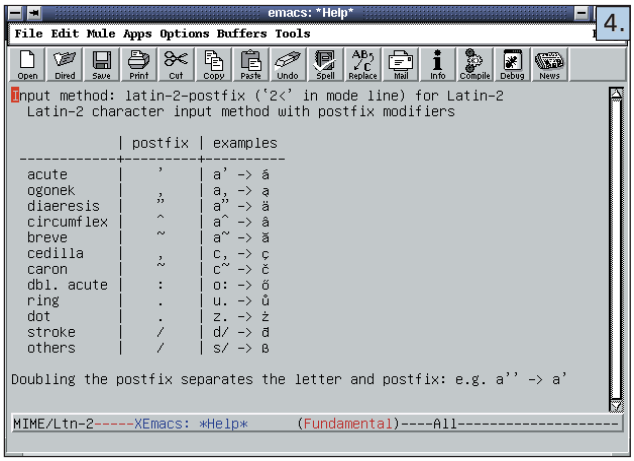
© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

3.



4.



annak, hogy megszokott környezetét máshol nem fogja megtalálni. Ha a magyar billentyűzet megnehezíti a dolgunkat, ne ragaszkodjunk feltétlenül az alapértelmezett billentyűkombinációkhoz, hanem használjuk az egeret vagy a menüket ugyanarra a feladatra! Ez lehet, hogy lassabb egy picit, de nem igényel nagyobb változtatásokat az Emacs eredeti viselkedésében.
Ha a kisméretű átmeneti tárban véletlenül elgépelünk valamit, nyomjuk meg a C+G teljes billentyűt, akár többször is. A C+X ESC+ESC megismétli a minitárba legutóbb beírt parancsot, az ***M+x list-command-history*** pedig kilistázza az összes korábban ide begépelt parancsot.

Az Öszvér

Ismét mozaikszóval állunk szemben, ami a Multi+lingual Enhancement to GNU Emacs (azaz a GNU Emacs többnyelvű bővítménye) szavakból lett létrehozva. A ***Mule*** (azaz Öszvér) találó név, mert segítségével egyetlen fájlban több nyelvet is keverhetünk, nyelvi szempontból „öszvérfájlt" hozva létre.
Vessünk egy pillantást a *2. kép*-re, ahol többek közt latin, orosz, görög, kínai, arab és héber írású szavak láthatók egymás mellett! Töltsük be most a ***Magazin/Emacs/emacs/src/RMain.pas*** forrásfájlt a CD-mellékletről, hogy a szerkesztőutasításokat gyakoroljuk! Az Emacs a fájlkiterjesztés alapján önműködően felismeri a programozási nyelvet, de a kiemelési módot nekünk kell működőképessé tenni a kisméretű átmeneti tárban, ahová nemcsak betűkombinációkat, hanem parancsokat is beírhatunk. Ha az ***RMain.pas*** fájlban jobban megnézzük a magyar nyelvű szövegrészeket, észrevehetjük, hogy az ***ő*** és ***ű*** betűkön kalapos ékezetek vannak. Át kell tehát váltanunk a Latin 2 betűkészletre. Az C+X+K teljes billentyűvel zárjuk be az ***RMain.pas*** fájlt tartalmazó átmeneti tárat, majd adjuk ki az

```
(defalias 'press-udoubleacute
  (read-kbd-macro "C-q 573"))
(defalias 'press-Udoubleacute
  (read-kbd-macro "C-q 533"))
(defalias 'press-odoubleacute
  (read-kbd-macro "C-q 565"))
(defalias 'press-Odoubleacute
  (read-kbd-macro "C-q 525"))

(defalias 'set-doubleacutes (read-kbd-macro
"M-x global-set-key RET <udoubleacute>
➥press-udoubleacute RET
 M-x global-set-key RET <Udoubleacute>
            ➥press-Udoubleacute RET
 M-x global-set-key RET <odoubleacute>
            ➥press-odoubleacute RET
 M-x global-set-key RET <Odoubleacute>
            ➥press-Odoubleacute RET"))

(defalias 'latin-2-unix (read-kbd-macro
"C-x C-m f iso-8859-2-unix RET
 M-x set-doubleacutes RET"))

(global-set-key [f2] 'latin-2-unix)
```

M+x set+language+mode RET parancsot, és írjuk be a ***Latin-2*** utasítást. Ezzel a nyelvet a közép-európai ISO-8859-2 karaktertáblára állítjuk. Alul az állománysoron megjelenik a „MIME/Ltn-2" üzenet. Most ismét hívjuk be az ***RMain.pas*** fájlt, és nézzük meg az ***ő*** és ***ű*** betűket! A ***Mule/Set language environment/Latin-2*** menüt is használhattuk volna, vagy a C+X C+M+L vagy a C+X ENTER+L billentyűkombinációt. Általában igaz az Emacsra, hogy ugyanannak a dolognak a végrehajtása többféleképpen is lehetséges, és mi magunk választhatjuk ki a nekünk tetszőt. Most lépjünk ki az Emacsból a C+X C+C billentyűk megnyomásával. Indítsuk el ismét az XEmacsot, és a nyelvet állítsuk Latin-2-re, majd hívjuk be a ***Magazin/Emacs/emacs/src/mule.txt*** fájlt!
A ***File/Save as*** vagy C+X C+W parancsokkal ezt a próbafájlt másoljuk át a merevlemezünkre, és válasszuk a ***Mule/Toggle Input Method*** menüpontot vagy nyomjuk meg a C+\ teljes billentyűt! Alul az állapotsoron megjelenik a „Quail" (fürj) szó – az Öszvér alkotói láthatóan szeretik az állatokat. A beviteli mód a ***latin+2+prefix*** lehet, amelynek lényege, hogy elsőként beírjuk az ékezetet létrehozó vesszőt, idézőjelet, kettőspontot stb., majd begépeljük a módosítandó karaktert, ahogy azt a *3. kép* táblázatában látjuk.
Próbálgassuk a fenti billentyűkombinációkat! Most válasszuk a ***Mule/Select Input Method*** menüpontot vagy nyomjuk meg a C+X C+M C+\ gombokat! Alul a kisméretű átmeneti tárolóban a TAB megnyomásakor megjelenő kiegészítő listából válasszuk a ***latin-2-postfix*** kapcsolót, vagy rögtön gépeljük be ezt a TAB megnyomása nélkül. Most előbb a módosítandó karaktert kell beütni, utána a módosítót. Gyakoroljuk a *4. kép* táblázatában látható kombinációit!
Állítsuk át az ***Options/Customize/Emacs/Environment/I18n/Mule*** menüpontban a ***verbose*** (szószátyár) és a ***highlight*** (kiemelés) kapcsolókat! Szószátyár módban segítséget kapunk a visszhangterületen, kiemelés módban pedig a program az éppen megváltoztatandó karaktert vagy módosítót aláhúzza. Tudjuk, hogy az ***I18n*** rövidítés az Internationalization szóból ered, mivel a kezdő ***I*** betű és az utolsó ***n*** betű között ***18*** karaktert

számlálhatunk össze. Az ***I18n*** munkafolyamattal készítjük fel a programunkat arra, hogy más nyelvi környezetben is a helyi sajátosságoknak megfelelően működjön.

Makrók használata

Az Emacs indításkor az ***.emacs*** nevű állományt keresi meg, ami alapértelmezetten ***Home*** könyvtárunkban van. Ide Lisp-függvényhívásokat kell beírni, amelyek megváltoztatják az Emacs alapértelmezett tulajdonságait és viselkedését. Ha mi magunk nem ismerjük a Lisp nyelvet, akkor sem vagyunk elveszve, mivel makrók használatával létrehozhatunk ilyen kódot. Mint tudjuk, a makrók rögzítik az általunk végrehajtott munkafolyamatot, és billentyűleütéshez kötődnek. Elvileg az egérkattintások is rögzíthetők, de a gyakorlatban ennek nincs sok értelme, mivel a makró az egér helyzetét jegyzi meg, és semmi sem garantálja, hogy legközelebb a kattintás helye ugyanaz lesz, mint a makró rögzítésekor. A billentyűmakrókat gyakorta ismétlődő hosszabb gépelések kiváltására használhatjuk. Példaként nézzük meg, hogyan lehet az ***ő*** és ***ű*** betűket makrókkal hozzákapcsolni a billentyűkhöz. Ehhez érdemes tudni, hogy a karakterek nemcsak a fenti ismertetett módon vihetők be a szövegbe, hanem a ***quoted-insert*** paranccsal is, ami a C+Q teljes billentyűhöz van kötve. Ha ilyenkor beírunk még egy három oktális számjegyből álló karakterkódot, a kívánt betű meg fog jelenni a szövegben. Az ***ű*** kódja 573, az ***Ű*** 533, az ***ő*** 565 és az ***Ő*** 525. Ha egy karakter kódjára esetleg nem emlékszünk, a C+X = teljes billentyűvel megtudhatjuk. Próbáljuk ki az ***ű*** betűvel, hogy az alábbihoz hasonló tájékoztatást kapjuk a visszhangterületen!

char: ű (0573, 379, 0x17b) point=1150 of 53167(2%) column 33

A ***char:*** kiírás után a lekérdezett ***ű*** betűt, majd a karakterkódot olvashatjuk oktális, decimális és hexadecimális formában. A ***point=*** után láthatjuk, hogy a mi ***ű*** betűnk a szövegben jelen lévő 53 167 karakter 1150. helyén van méghozzá a 33 oszlopban. A betű helyzete a teljes méret két százalékán van. Miután kipróbáltuk a fenti parancsokat, elkezdhetjük a makrók rögzítését. Elsőként az ***ű*** betűt kössük az ***udoubleacute*** billentyűhöz. A makrók rögzítését a C+X (teljes billentyűvel kezdjük, és talán nem meglepő, hogy a C+X) billentyűvel fejezzük be. Tehát most a következőket kell tennünk:

1. ***C-x (***
2. ***C-q 573***
3. ***C-x)***

Miután rögzítettük, a makrót az ***M+x call-last-kbd-macro*** paranccsal vagy a C+X E teljes billentyűvel többször egymás után előhívhatjuk. Ennek a parancsnak számkapcsolót is megadhatunk, azaz például a C+U 20 C+X E parancs húsz darab ***ű*** betűt fog beszúrni a szövegbe. Ha hosszabb ideig akarunk egy makrót használni, célszerű nevet adni neki. Ez az ***M+x name-last-kbd-macro RET*** a makró neve ***RET*** paranccsal tehetjük meg:

4. ***M+x name-last-kbd-macro RET press+udoubleacute RET***

Miután elneveztük a makrót, a programmal megírattathatjuk a hozzá tartozó Lisp-kódot az ***M+x insert-kbd-macro RET a makró neve RET*** paranccsal:

5. ***M+x insert-kbd-macro RET press+udoubleacute RET***

Ez a parancs a következő sorokat szúrja be a működő átmeneti tárba:

```
(defalias 'press-udoubleacute
  (read-kbd-macro "C-q 573"))
```

Ez már érvényes Lisp-kód, amit bemásolhatunk az ***.emacs*** fájlba, hogy az Emacs minden indításakor elérhető legyen. Tegyük meg, majd indítsuk újra az Emacsot! Most van egy saját parancsunk, és az ***M+x press-udoubleacute*** begépelgetésével ű betűket írhatunk a szövegbe. Gyakorlásképpen ismételjük meg a fenti lépéseket a többi három betűvel is!
Természetesen rögtön látjuk, hogy az ilyen hosszú parancsok beírása nem a leghatékonyabb módja egyetlen betű beszúrásának, ezért parancsunkat célszerű billentyűhöz kötni. Ehhez az ***M+x global-set-key RET billentyű parancsnév RET*** parancsot kell használnunk:

6. ***M+x global-set-key RET ű press+udoubleacute RET***

Ha mindent utasítást jól hajtottunk végre, most már működni fog az ű billentyűnk, és egyetlen mozdulattal beírhatjuk az ***ű*** betűket. Megemlítem még a ***local-set-key*** parancsot is, aminek ugyanaz a feladata, mint a ***global-set-key*** parancsnak, de hatóköre jóval szűkebb: csak az adott átmeneti tárra és ugyanazon főmódra korlátozódik.
Ha az alábbi, makrók által létrehozott kódot bemásoljuk az ***.emacs*** fájlba, az F2 billentyű megnyomásával beállíthatjuk a ***Latin-2***-kódolást, és az ***ő***, ***Ő***, ***ű***, ***Ű*** betűk bevitelét a megfelelő billentyűkhöz köthetjük hozzá, mint az *listánkon* is látható.
A kóddal kapcsolatban megjegyzem, hogy a saját dolgainkat nem célszerű az F2 vagy más billentyűhöz kötni, még akkor sem, ha az adott pillanatban úgy tűnik, hogy az Emacs azokat semmire sem használja. Lehet olyan fő- vagy almód, ami átértelmezi az addig látszólag semmire sem használt billentyűt. Ezért a fejlesztők azt ajánlják, hogy inkább a ***C+c*** előtaggal kössük saját parancsainkat, ami kifejezetten a felhasználók számára van fenntartva. Azért hagytam mégis a példában, mert az ***.emacs*** fájlban másként kell kötni a funkcióbillentyűket és a közönségeseket. Figyeljük meg, hogy zárójelek közé kell tenni az ***F2*** nevet, míg a normál billentyűknél idézőjeleket használunk:

```
(global-set-key "\C-cl" 'latin-2-unix)
```

Ha ezt a sort másoljuk be a fenti helyett az ***.emacs fájlba***, a ***latin-2-unix*** saját parancsot a C+C+L teljes billentyűhöz kötjük, amit nagy valószínűséggel egyetlen Emacs-fejlesztő sem használt még fel. Meg kell azonban jegyeznem, hogy a Héj, a TeX és más módok gyakran használnak ***C-c*** előtagú parancsokat!
Ha egy kötést érvényteleníteni akarunk, az ***M+x global-unset-key RET*** billentyű és az ***M+x local-unset-key RET*** billentyűparancsokat használjuk!
Következő számunkban innen folytatjuk.

Szaló István
(ratiosoft@freemail.hu) tanár, immár több mint másfél évtizede foglalkozik programozással, de csak a Java és a Linux megismerése után tudta meg, hogy mi is az igazi programozás. Több írása megjelent már a hazai számítástechnikai lapokban. Ha néha feláll számítógépe mellől, rendszerint művészettörténész feleségével és kisiskolás lányával „találja szemben” magát.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Adatvesztés ellen védekezz Mondo Rescue-val

Hugo elmondja, hogyan állíthatjuk helyre adatainkat, ha visszafordíthatatlan adatvesztés következne be.

Minden számítógép-felhasználó vesztett már el adatokat valamikor élete folyamán. Ez az esemény sajnos a legritkább esetben tanítja meg őket arra, hogy rendszeresen biztonsági mentést készítsenek az adataikról. A Mondo Rescue egy vagy több rendszerindításhoz használható CD-t készít a fájlrendszer alapján. Így akkor is vissza tudjuk állítani a teljes rendszert, ha visszafordíthatatlan adatvesztés következne be. A Mondo támogatja a Linux, illetve Windows kettős rendszereket és a RAID, ReiserFS, ext2, ext3, XFS, JFS és VFAT fájlrendszereket is.

Háttér

A Mondo virágzó nyílt forrású program: képes az operációs rendszer és az adatok helyreállítására a puszta vastól kezdve, és 2000 májusa óta egyetlenegyszer sem számoltak be adatvesztésről. A program fejlesztése jó ütemben halad előre.
Eredetileg a Mondo létrehozásának célja az volt, hogy csak a Windowst futtató rendszerekről legyen képes biztonsági mentést készíteni, és a Windows-felhasználók számára nyújtson helyreállítási lehetőségeket balesetek esetén. Van egy kis számítógépes boltom a Tennessee állambeli Nashville közelében, és észrevettem, hogy sokan, akik számítógépet vásároltak tőlünk, később visszahozták őket, hogy a merevlemezt formázzuk újra és a rendszert telepítsük újra. Vajon ennek csupán az volt az oka, hogy nem tudták, miként kellene karbantartaniuk saját számítógépüket? Vajon az alkatrészeink okozták-e a bajokat? Mindenesetre szükségünk volt rá, hogy felügyelet nélkül, 10–15 perc alatt fel tudjuk telepíteni a Windows új példányát egy átlagos számítógépre. Akkoriban a Norton Ghostot használtuk, de a Ghost nem tette lehetővé, hogy csak válogatott részeket rakjunk fel újra; vagy mindent kellett vagy semmit sem lehetett felraknunk. Ezenkívül a Ghost sokszor összeomlott, ha Linux-lemezrészeket kellett másolnia.
Azt találtuk ki, hogy egy CD-n Linuxot futtatunk, és a Windowst ezen a CD-n nagy ***tar***-állományokba csomagoljuk. Röviden összefoglalva ezt a hosszú és kínkeserves történetet, elmondhatjuk, hogy sikerült. Sőt, megírtam egy olyan kis programot is, amely képes létrehozni Windows által indítható VFAT lemezrészeket.
2001 márciusában programunk felkeltette a Hewlett-Packard egyik grenoble-i alkalmazottjának az érdeklődését. ***Bruno Corvec***, a kiszolgálókkal foglalkozó részleg egyik ügyes programozója alaposabban is szemügyre vette a kódot. Játszadozott vele, és számos szempontból tovább is fejlesztette. 2000. szeptembere (a programírás felfüggesztése) és 2001 márciusa között a Mondo két újabb változata jelent meg. Mindkettő csaknem teljes egészében Bruno Corvec, ***Maciej Kulasa*** és más közreműködők munkájának volt köszönhető. 2001 júniusában összeraktam egy új számítógépet és munkához láttam. Július 4-ére kijavítottam az összes nagyobb hibát, és megjelent a Mondo 1.00 változata (kérem, mellőzzétek a függetlenségnapi viccelődést).

A működés módja

A Mondo egyszerűen egy kis program, amely két másik eszközön, a Mindin és az afion alapul.
A Mindi, más néven a Mindi-Linux egy miniterjesztés, mely a rendszermag, a modulok, segédprogramok és programkönyvtárak alapján készít rendszerindításra alkalmas lemezeket. Ennek egy általános rendszerindító lemeznél nagyobb esélye van rá, hogy binárisan is összeillő legyen a már meglevő programjainkkal, ugyanis segédprogramjait közvetlenül a saját lemezünkről másolja le. Képes El Torito formátumú, 2,88 MB méretű rendszerindító lemezképek létrehozására is. A Mondo a Mindi segítségével elkészít egy 2,88 MB méretű rendszerindító „lemezt”, és az ehhez tartozó adatokat tartalmazó „lemezeket”, melyek minden Mondo CD-re felkerülnek. Amikor újraindítjuk rendszerünket, ugyanazok a modulok lesznek betöltve, mint a biztonsági másolat elkészítésének időpontjában. Elméletileg tehát ugyanolyan környezetben indul el a gépünk, mint amilyenben a biztonsági másolat elkészítésének időpontjában volt.
A Mondo-Archive nagyjából a következőképpen működik:

1. Állományaidat több, egyenként 5–10 MB méretű ***tar***-állományba tömöríti.
2. Ezeket a ***tar***-állományokat összegyűjti egy könyvtárba.
3. Ugyanebben a könyvtárban elhelyez egy El Torito formátumú rendszerindító hajlékonylemezről készült képet.
4. A `mkisofs`-t ebben a könyvtárban elindítja, a kimenetet pedig továbbadja a `cdrecord`-nak, így elkészül egy CD, mely ennek a könyvtárnak és tartalmának pontos mását tartalmazza.
5. A fenti lépéseket ***N*** számú CD-nél megismétli, ahol ***N*** azoknak a CD-knek a számát jelenti, melyekre összes fájlunk elhelyezéséhez szükség van.

A rendszermag beállítása

A Mindi 0.38 kiadása a rendszermag 2.4.7-es változatát tartalmazza a biztonság érdekében, arra az esetre, ha saját rendszermagunk nem támogatja a rendszerindító lemez elkészítéséhez szükséges összes szolgáltatást. Ez nem azért van, mert a Mondo annyira válogatós; hanem a Linux furcsaságai miatt. Léteznek olyan rendszermagok is, melyek egyszerűen nem alkalmasak rendszerindításra. A kezdő felhasználók maradjanak meg az alapértelmezett rendszermagnál, és a Mindivel használtassák a saját rendszermagját azáltal, hogy nemmel felelnek arra a kérdésre, miszerint saját rendszermagjukat akarják-e használni a rendszerindító lemezen.
A haladó felhasználóknak meg kell győződniük róla, hogy rendszermagjuk támogatja a következő eszközöket és szolgáltatásokat: CD-ROM, IDE CD-k, IDE, `initrd` és ramlemezek, hajlékonylemezes meghajtók, üzembiztos loopfs-támogatás (ami azt jelenti, hogy a rendszermagnak a 2.2.17-nek vagy későbbinek, illetve 2.4.5-ac10 változatúnak vagy frissebbnek kell lennie), valamint az ISO9660 támogatása.
Amennyiben rendszermagunk mindezeket nem támogatja, valószínűleg jobb, ha a Mindi saját rendszermagját használjuk

mindaddig, amíg meg nem bízunk magunkban annyira, hogy elkészítsük saját rendszermagunkat. Ha úgy látod, hogy a Mindi rendszermagja valamit nem támogat, amire neked szükséged van (például XFS), akkor kérlek, szólj nekem. A Mindi következő változatába valószínűleg e tulajdonságnak a támogatását is belefoglalom.

A Mondo beszerzése és telepítése

Ha szeretnéd felrakni a programot, látogass el a ➲ http://www.microwerks.net/hugo címre, és töltsd le onnan a Mondót és a Mindit. Az utóbbi program részét alkotja az elsőnek, de leválasztottam róla, mert a Mindi önállóan is képes rendszerindító lemezek készítésére a rendszermagod, moduljaid, segédprogramjaid és programkönyvtáraid alapján. A letöltést lehetővé levő weblapon mindkét program telepítéséhez szükséges útmutató megtalálható.
Az RPM-eket használóknak könnyű dolguk van: egyszerűen letöltik a Mindi RPM-jét a */tmp* könyvtárba, majd letöltik a Mondo RPM-jét a */tmp* könyvtárba, és kiadják a következő parancsot:

```
rpm -Uvh /tmp/mondo-1.13-1.i386.rpm
➥/tmp/mindi-0.39-1.i386.rpm
```

A *tar*-állományok használóinak kicsivel nehezebb a dolguk: le kell tölteniük a Mindi tar-tömörítvényét a */tmp* könyvtárba, majd a Mondo tar-tömörítvényét is ugyanoda, ezután pedig ki kell adniuk a következő parancsokat:

```
cd /tmp
tar -zxvf midi-0.39.tgz
cd mindi-0.39
./install.sh
cd ..
tar -zxvf mondo-1.13.tgz
cd mondo-1.13
./install.sh
```

A Mondo által használt egyéb eszközök

A próbakorong készítése jó ötlet, ugyanis az új felhasználók így kipróbálhatják a programot anélkül, hogy tönkretennék saját rendszerüket. Először is meg kell győződnöd róla, hogy a Linux tudja használni a CD-íródat. Utána elindíthatod a *mondo-archive* programot. CD-meghajtódat így keresheted meg:

```
dmesg | grep CD
```

Ha IDE-csatolójú CD-íród van, akkor a `/dev/hdX` sort látod majd, ahol `X` egy *a* és *h* közötti betű. Ha a SCSI-emuláció helyesen lett beállítva, a `cdrecord -scanbus` parancs kiadásával látni fogod a listában CD-íródat.
Amennyiben a CD-íród megfelelően be lett állítva, valami ehhez hasonlót látsz majd:

```
0,0,0 --- JoeCamel 4x CD writer
```

Az eszköz leírásától balra található `0,0,0` szám az a SCSI-eszköz, ahol az író található. Ezt a számot írd fel.
Ha azt szeretnéd, hogy a biztonsági mentést tartalmazó CD-re felkerüljenek bizonyos különleges programok, akkor kézzel hozzá kell adnod ezt a programot és a beállítófájljait a */usr/share/mindi/deplist.txt* fájlhoz. A Mindi megkeresi a könyvtárakat és hozzáadja a CD-hez.

Futtasd a Mindit, és hozz létre vele néhány rendszerindító lemezt, hogy megbizonyosodj róla, rendesen működik-e a gépeden. Az alábbiakat kell beírnod:

```
cd /usr/share/mindi
./mindi
```

Ha a rendszermagod túl nagy (körülbelül 900 KB-nál nagyobb), akkor nem tudsz rendszerindító hajlékonylemezeket létrehozni, bár ekkor is gyárthatsz rendszerindításra képes CD-ket. Mindkét esetben a Mindit a legegyszerűbben úgy próbálhatod ki, ha az *N* betűt nyomod meg, amikor a program megkérdezi, készítsen-e rendszerindító hajlékonylemezt („Create boot floppies?"), és az Y-t nyomod meg, amikor azt kérdezi, készítsen-e lemezképet („Create iso image?"). Ezután a `cdrecord` segítségével létrehozhatod a rendszerindításra képes egyszer írható (CD-R) vagy újraírható (CD-RW) lemezt. Ezt kell beírnod:

```
cd /root/images/mindi
```

Utána a következő lehetőségek közül választva megírhatod a CD-t, attól függően, hogy az íróban levő lemez CD-R vagy CD-RW. Az `x,x,x` helyére a saját íród SCSI-beállításait írd. Újraírható lemez esetén ezt a parancsot kell kiadnod:

```
cdrecord blank=fast dev=x,x,x speed=2
➥mindi.iso
```

Egyszer írható CD esetén pedig ezt:

```
cdrecord dev=x,x,x speed=2 mindi.iso
```

Zárd be az összes alkalmazást, és indítsd újra a rendszert a CD-ről, ne pedig a merevlemezről. (Lehetséges, hogy a BIOS beállításait meg kell változtatni ahhoz, hogy a számítógép a merevlemez helyett CD-ről induljon.) Ha rendszered rendben elindul a CD-ről, biztos lehetsz benne, hogy a Mondo is rendszerindításra alkalmas biztonsági mentést készít majd a CD-re. Az eszményi biztonsági CD természetesen a saját rendszerma-

A Mondo felhasználásának más lehetőségei

- Egy Linux-telepítés pontos lemásolására is alkalmazható.
- Mentés készítése nem RAID fájlrendszerről, és helyreállítás RAID-ként, beleértve a saját lemezrészt is (ha a rendszermag támogatja).
- A rendszer lementése egy bizonyos formátumban, és visszaállítása másik formátumban.
- A lemezrészek átalakíthatók, például csökkentés, illetve növelés, az eszközök átnevezése, merevlemezek hozzáadása stb. azelőtt, hogy a meghajtóidat felosztanád és megformáznád. A Mondo helyreállítja az adatokat és a változásoknak megfelelően módosítja a */etc/lilo.conf* és a */etc/fstab* fájlokat.
- Mentés készíthető Lin/Win-rendszerekről, beleértve a rendszerindító lemezrészt is. A Mondo mindent helyesen fog beállítani, amikor az adatokat visszatöltöd. Biztonság kedvéért azért futtasd le a Scandisket, amikor először indítod el a Windowst.
- A Mondóval készített biztonsági mentések révén ellenőrizheted számítógéped sértetlenségét.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

godat tartalmazza. Azt ajánlom, hogy amikor csak lehetséges, a saját rendszermagodat használd, ez ugyanis a lehető legkisebbre csökkenti annak veszélyét, hogy a rendszerindító CD nem támogatja géped alkatrészeit vagy fájlrendszereit stb. Végül a teljes biztonsági mentést a következő parancsok kiadásával készíthetjük el:

```
cd /home
mondo-archive --burn-cds 2 0,0,0 --comp-level 9
```

A `2` azt jelzi, hogy kétszeres sebességgel írjuk meg a CD-t. Ha újraírható CD-re írsz, ezt a parancsot add ki:

```
mondo-archive --burn-cds 2 0,0,0 cdrw
➥--comp-level 9
```

A parancs kiadása után helyezz a meghajtóba egy üres CD-R(W) lemezt, és hagyd magára a gépet. Ennyi az egész. Én mindig a legjobb tömörítést választom (`9`), mert munkába indulás előtt szoktam elindítani a Mondót. Amikor hazaérek, beteszem a második újraírható CD-t, és várok fél órát. Ennyit vesz igénybe a napi biztonsági mentés elkészítése.
Az alapértelmezés szerinti tömörítés a `3`. Ha nagyon sietsz, használd az `1`-es tömörítést (`--comp-level 1`), amivel felgyorsíthatod a biztonsági másolat elkészítését. Ezzel több CD-t fogyasztasz el, de elméletileg rövidebb idő alatt elkészül.
Ha a Mondo nem talál CD-t a meghajtóban, akkor megáll egy ***Retry***/***Fail***/***Abort*** (Újra/Mégsem/Kilép) üzenettel. Ha beteszünk egy CD-t, és a ***Retry*** (Újra) lehetőséget választjuk, akkor újrapróbálja, mintha mi sem történt volna. Amennyiben a ***Abort*** (Kilép) lehetőséget választod, a program leáll. Ha a ***Fail*** (Mégsem) lehetőséget választod, akkor a program azt a CD-t kihagyja, de folytatja a biztonsági másolat készítését. Legtöbb esetben a ***Retry*** a legjobb választás.
Ha vannak valóságos elérési utak, melyeket nem akarsz belefoglalni a biztonsági másolatba, ilyenkor a következő kapcsolóval zárhatod ki őket:

```
--exclude-paths /foo /bar /xanadu
```

Ha csak bizonyos elérési utakat akarsz belefoglalni, akkor a `--bkpath /home` kapcsolót kell használnod. Tehát ha csak a saját (***/home***) és rendszerindítás (***/boot***) könyvtárról akarsz biztonsági másolatot készíteni, de ki akarod hagyni a közös MP3 könyvtárt, ezt a parancsot kell kiadnod:

```
mondo-archive --burn-cds 2 0,0,0 cdrw
➥--bkpath /home /boot --exclude-paths
➥/home/MP3S /home/WAVs /home/secret
```

Ha nem azonnal akarod elkészíteni a CD-ket, hanem először CD-lemezképet akarsz készíteni, amelyekből a CD-t később írod fel, akkor a következő parancsot add ki:

```
mondo-archive --isodir /root --bkpath /home
➥/boot --exclude-paths /home/MP3S /home/WAVs
➥/home/secret
```

Ezzel létrehozod az ***1.ISO***, ***2.ISO*** stb. fájlokat, és mented őket a ***/root*** könyvtárba.
A Mondo-Archive futtatása előtt feltétlenül adj hozzá néhány fájlt a ***/usr/share/mindi/deplist.txt*** fájlhoz, futtasd le a `mount` parancsot és győződj meg róla, hogy befűzted azokat a lemezrészeket, amelyekről biztonsági másolatot szeretnél készíteni. Futtasd le a `df` parancsot is, hogy megállapítsd, mekkora lesz a biztonsági mentés, milyen tömörítésre lesz szükség, illetve hány CD-t kell felhasználnod.
Az összehasonlítás folyamata felgyorsítható azzal, ha a rendszer elindítása után terminált váltasz, és bekapcsolod az ***ide-opt***-ot. Nyomd meg az ALT és a bal nyílbillentyűt, majd pedig futtasd az `ide-opt` parancsot. Ez bekapcsolja a DMA-t és más hasznos nyalánkságokat.
Ha össze akarod hasonlítani a CD-n levő biztonsági mentést az élő rendszeredben levő fájlokkal, indítsd el rendszeredet a CD-ről, és válaszd az összehasonlítás üzemmódot (írd be a `compare` parancsot és nyomd meg az ENTER-t). Az összehasonlítási folyamat lefutása után nézd meg a ***/tmp/mondo-restore.log*** fájlt, amely megmutatja, mely fájloknál volt eltérés.
A Mondo alkalmazása során tapasztalt kezdeti gondoktól eltekintve, melyekkel a saját rendszermagod rendszerindításra való felhasználása során találkozhatsz (ugyanis néhány rendszermag nem alkalmas rendszerindító lemez készítésére, ezért újra le kell fordítani), minden bizonnyal azt állapítod majd meg, hogy a program eléggé unalmas. Azt teszi, amit ígér. Tömöríti az összes fájlodat az egyszer írható vagy újraírható CD-kre, és ha szükség van rá, helyreállítja őket. Felosztja a lemezeidet, megformázza őket, helyreállítja az adatokat, és lefuttatja a LILO-t, hogy beállítsa a rendszerindító lemezrészt.

Egyszerű helyreállítás

Képzeld el, hogy a merevlemezedről véletlenül vagy szándékosan minden adat törlődik. Vagy képzeld el azt, hogy pontosan le akarod másolni meglevő operációs rendszeredet. Mindkét esetben ***Nuke***-módban kell futtatnod a programot. Indítsd el számítógépedet a Mondo CD-ről, írd be a `nuke` parancsot, és figyelj. Ennyi az egész.
Ha pontosan látni szeretnéd, hogy a Mondo mit csinál a helyreállítás során, nyomd meg az ALT+A billentyűket, és írd be a következő parancsot:

```
tail -f /tmp/mondo-restore.log
```

Így részleteiben is figyelemmel kísérheted a helyreállítás folyamatát.

Válogatott részek helyreállítása

Ha csak néhány állományt akarsz helyreállítani, vagy a lemezeidet nem akarod előkészíteni és megformázni, akkor párbeszédes módban kell elindítanod a gépet. Amikor a gép elindul, add ki az `interactive` parancsot, és üsd le az ENTER-t. Számos kérdésre kell majd igennel vagy nemmel válaszolnod: szeretné felosztani a lemezeit? Szeretné megformázni őket? Szeretne mindent visszaállítani? Szeretne valamit visszaállítani? Szeretné futtatni a LILO-t, és beállítani a rendszerindító lemezrészt?
A párbeszédes mód azoknak való, akik elvesztették élő rendszerük egy részének az adatait, vagy esetleg a legutóbbi biztonsági mentésük adatainak egy részét, és egy korábbi biztonsági mentésből akarják visszanyerni egy részüket.

A profi mód használata

Ha elkészítetted a biztonsági másolatot rendszeredről, és a rendszert a CD-ről összehasonlító módban elindítottad, valamint ellenőrizted a lementett állományokat, akár kísérletezhetsz is rendszereddel. Mozgathatod a lemezrészeket, megváltoztathatod a méretüket, be- és kikapcsolhatod a RAID-et, játszadozhatsz más rendszerindító programokkal stb. A Mondo ebben valóban

nagyszerű. Véleményem szerint az átlagos linuxos számítógépre a gondatlan rendszergazda jelenti a legnagyobb veszélyt.
Ha ki szeretnél próbálni ezek közül a trükkök közül néhányat, akkor indítsd el a rendszert a Mondo CD-ről, és válaszd a profi módot. Ennek hatására parancssort kapsz. Szerkeszd át a befűzött kötetek listáját tartalmazó fájlt. A befűzött kötetek listáját a */tmp/mountlist.txt* nevű szövegfájl tartalmazza, mely a ramlemezen található, miután rendszeredet elindítottad a Mondo CD-ről. Felsorolja azokat a lemezrészeket, melyeket a program létre fog hozni, valamint ezeknek a méretét, csatolási pontját és a formátumukat. Ha a lemezrészek méretét vagy elhelyezkedését módosítani szeretnéd, ezt a fájlt egyszerűen csak a `pico /tmp/mountlist.txt` parancs segítségével át kell szerkesztened (vagy használhatod bármelyik kedvenc szövegszerkesztődet). Mentsd a módosításokat és zárd be a fájlt a CTRL+X billentyűk és az ENTER megnyomásával. Alább látható a befűzött kötetek listájának egy mintája. A méretek kilobájtban vannak megadva, ezért gondosan számold meg a nullákat, ha az értékeket megváltoztatod.
Ne felejtsd, az új elrendezés mindaddig nem lép életbe, amíg a `mondo-restore` parancsot nem futtatod le, amivel átváltoztatod és újra megformázod a meghajtókat:

```
/dev/hda1   /mnt/windows  vfat  4096000
/dev/hda2   swap                swap    256
/dev/hda3   /                   ext2    8192000*
```

Ha meg szeretnéd változtatni a saját lemezrész méretét és típusát, csupán az érintett mezők értékeit kell átírnod:

```
/dev/hda1   /mnt/windows  vfat  4096000
/dev/hda2   swap                swap    256
/dev/hda3   /                   reiserfs
                                ➥16384000*
```

Egy másik lehetőség, hogy több, elsődleges lemezrész használata helyett egy elsődleges lemezrészt (`hda1`), egy kiterjesztett lemezrészt (`hda2`, melyet a Mondo hozott létre és kezel), és több logikai lemezrészt használsz. Figyeld meg az új, `/dev/hdaN` bejegyzéseket:

```
/dev/hda1  /mnt/windows    vfat        4096000
/dev/hda5  swap            swap        256
/dev/hda6  /               reiserfs    8123000
/dev/hda7  /usr            reiserfs    4099000
/dev/hda8  /home           reiserfs    4099000*
```

Ha a gépedhez új merevlemezt (például elsődleges szolga) csatlakoztatsz, lemezrészeid némelyikét áthelyezheted arra a meghajtóra. Figyeld meg a változtatásokat az alábbi listában:

```
/dev/hda1  /mnt/windows    vfat        4096000
/dev/hda2  swap            swap        256
/dev/hda3  /               reiserfs    81422000
/dev/hdb1  /home           reiserfs    9481000
/dev/hdb5  /usr            reiserfs    16384000
/dev/hdb6  /tmp            reiserfs    1589000
```

Valamivel bonyolultabb a helyzet, ha RAID-re akarsz áttérni, ugyanis ekkor a */etc/raidtab* fájlt is létre kell hoznod. Ezt a profi módban belülről is megteheted. Csak írd be a `pico /etc/raidtab` parancsot, és hozz létre egy jó *raidtab* fájlt (ennek leírása meghaladja cikkünk kereteit). Ezután cseréld ki a hagyományos eszközt egy RAID-eszközre (`/dev/mdN`):

```
/dev/hda1  /mnt/windows  vfat  4096000
/dev/hda5  swap                swap     256
/dev/md0                   /       reiserfs
                                   ➥16384000
```

A befűzött fájlok listájának átszerkesztése után hajtsd végre a `mondo-restore` parancsot. Amikor a program megkérdezi, fel akarod-e osztani és meg akarod-e formázni a lemezeidet, felelj igennel. Szándékaidtól függően ekkor helyre is állíthatod az adatokat és futtathatod a LILO-t is, hogy beállítsa a rendszerindító lemezrészt is. Amennyiben csupán egy új, lemezrészekre való felosztást próbálsz ki, a többi kérdésre valószínűleg nemmel felelsz majd. Máskülönben mondj igent.

Összegzés

A jó biztonsági mentést készítő programoknál rendkívül fontos a jó leírás, az elvégzett alapos próbák sora és a könnyű használhatóság. Ha a program túl érzékeny, akkor a felhasználók nem veszik maguknak a fáradságot, hogy használják.
Ha te is közénk szeretnél tartozni, kérlek, töltsd le a programot és csatlakozz a levelezőlistánkhoz, szeretnék rólad is hallani.

Hugo Rabson
26 éves, Nashville-ben (Tennessee) lakik és dolgozik (a WebMD-nek). A Mondóval a Linux-közösségnek próbálja meg viszonozni azt, amit kapott. Szívesen olvasná az észrevételeit.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Könnyű álmok: a protokollok (11. rész)

Sorozatunk ezen részében a napjainkban használt internetes alkalmazások belső működéséről rántjuk le a leplet.

A fő célkitűzésünk, hogy ismertessük a legfontosabb protokollok működését, valamint hogy miként lehet őket csomagszűrővel elérhetővé tenni.
Mint arról már a korábbi cikkekben szó esett [1.], a Linux csomagszűrője (a 2.2-es és a 2.4-es rendszermagé is) három alapvető szűrőláncot tartalmaz. Az `input` láncra érkeznek be a hálózatról és a helyi csatolóról (lo) érkező csomagok, az átmenő forgalom a `forward` láncon halad keresztül, és a kimenő csomagok az `output` láncon keresztül hagyják el a hálózati alrendszert.
Ne felejtsük el, hogy míg a 2.2-es rendszermag csomagszűrőjénél egy átmenő csomag valóban minden láncon keresztülhaladt, addig az új 2.4-es rendszermag szűrőjénél (Netfilter, lásd Linuxvilág 2001. október, 27. oldal) az átmenő forgalom csak a `forward` láncon halad keresztül.
A cikk 2.2-es rendszermagra vonatkozó példáinál az egyszerűség kedvéért az átvivendő csomagok engedélyezését kizárólag az `input` láncon mutatjuk be. Ennek helyes működéséhez vagy a másik két láncon is engedélyezni kell e csomagok továbbítását, vagy biztosítanunk kell, hogy a másik két lánc ne akadályozza a forgalom zavartalanságát.
Írásunkban szereplő példákban feltételezzük, hogy tűzfalunknak csupán két lába van. Az `eth0` csatoló a belső (védett), míg az `eth1` csatoló a külső hálózat (Internet) felé mutat.

Ping

A `ping` a legalapvetőbb programok egyike. Gyakran használják a távoli gépek működésének ellenőrzésére vagy a hálózat késleltetésének mérésére. Először nézzük meg a `ping` által létrehozott hálózati forgalmat!
Mint az *1. ábrán* szereplő `tcpdump` kimenetből is látható, a ***cheetah*** nevű gép egy `icmp echo request` csomagot küld (az `icmp` csomagtípusa `echo request`, a kódértékek ehhez a típushoz nem tartoznak). A megcímzett gép (a példán a ***dolphin*** nevű) erre egy `icmp echo reply` csomaggal (az `icmp` csomagtípusa `echo reply`, kód értékek ehhez a típushoz sem tartoznak) válaszol. A `ping` program az `echo request` elküldése és az `echo reply` beérkezése közti időt méri, és a kettő különbségét írja a `time` mezőbe.
A `ping` parancs az `icmp` csomag adatrészében olyan adatokat helyez el, hogy a visszakapott választ azonosítani tudja (például kiszűrhesse a többszörös válaszokat, valamint felismerhesse a csomagvesztést). Amennyiben a csomagot az üzenetszórt (broadcast) címre irányítjuk, a hálózaton elhelyezkedő gépek közül több is válaszolhat – erre láthatunk példát a *2. listán*. Az üzenetszórt címre érkező `echo request` csomagok elfogadása néha hasznos lehet, ugyanakkor veszélyeket is rejt magában. Előszeretettel használják `DoS` (Denial of Service)-támadásoknál. Gondoljunk csak bele, hogy a támadó beküld egy olyan csomagot a hálózatunkba, amelynek forráscímét egy ártatlan gép címére hamisította. Ebben az esetben megcímzett gépeink válaszolnak, és ahány gépünk csak van, mind `echo reply` csomagot fog küldeni az „ártatlan” gépnek. Ezzel a támadó eléri, hogy egyetlen csomag költségével hálózatunkat erősítőként használva gépeink számával megegyező számú csomagot küldjön áldozatának. A támadás neve az angol szakirodalomban ***smurf attack***. Az üzenetszórt címre érkező csomagokra a Linux-rendszermag alapesetben válaszol, de ez le is tiltható. A letiltáshoz az alábbi parancsot használhatjuk:

```
echo 1 >
➥/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

Természetesen ezt minden indításkor meg kell tenni.
A feladatot bízzuk a `sysctl` vagy `systune` parancsokra, vagy a fenti sort tegyük be egy olyan állományba, amely minden indításkor végrehajtódik.
Visszatérve az ábrához: látható, hogy a ***cheetah*** gép `echo request` csomagjára mind a ***dolphin***, mind a ***hawk*** gép válaszol. Érdemes megfigyelni, hogy a ***cheetah*** gép magának is válaszol. Az ábra megmutatja, hogy először a gép saját válasza érkezik meg, és csak a másik kettőt minősíti másodpéldánynak.
A `ping` paranccsal kapcsolatban fontos megjegyezni, hogy az `echo request` csomagot a távoli gép rendszermagja dolgozza fel, az `echo reply` csomagot is ő küldi. Amennyiben a távoli gép az `echo request` kérésre válaszol, akkor a távoli rendszer TCP-, illetve IP-alrendszerének ezen része működőképes. Ettől azonban a gép még teljesen használhatatlan is lehet. Ugyanígy ha a `ping`-re elmarad a válasz, ez még nem feltétlenül jelenti a távoli rendszer teljes leállását. Oka lehet egy hálózati hiba vagy paranoiás tűzfal is.
Amennyiben a tűzfalon engedélyezni szeretnénk, hogy a belső gépek a külső gépeket `ping`-gel ellenőrizhessék, kifelé engedélyeznünk kell az `echo request`, befelé pedig az `echo reply` csomagokat. Ha Linux-rendszerünk a belső hálózaton saját címosztályt használ, használt rendszermagunk `CONFIG_IP_MASQUERADE_ICMP` beállítási lehetőségének engedélyezettnek kell lennie. A `ping`-et az alábbi parancsokkal engedélyezhetjük abban az esetben, ha nem használunk szűrést sem az átmenő, sem a kimenő forgalomnál:

- 2.2.x-es rendszermagsorozat esetén:

```
ipchains -A input -i eth0 -p icmp
➥--icmp-type echo-request -j ACCEPT
ipchains -A input -i eth1 -p icmp
➥--icmp-type echo-reply -j ACCEPT
```

- 2.4.x-es rendszermagsorozat esetén (Netfilter használata mellett):

```
iptables -A FORWARD -i eth0 -p icmp
➥--icmp-type echo-request -j ACCEPT
iptables -A FORWARD -i eth1 -p icmp
➥--icmp-type echo-reply -j ACCEPT
```

© Kiskapu Kft. Minden jog fenntartva

Traceroute

A `traceroute` program rendkívül hasznos segédeszköz a forgalomirányítási hibák felderítésére. A program működésének megértéséhez célszerű feleleventénünk a korábbi számokban leírtakat az IP-protokoll működéséről [2.].
A program a hálózati út meghatározásához trükkösen összeállított UDP-csomagokat használ. A trükk a csomagban az, hogy az élettartam-számlálóját (time to live) 1 értékre állítja. Így a legelső forgalomirányító berendezés, amely veszi a csomagot, rögtön megállapítja, hogy a csomag élettartama már lejárt, amit a feladóval egy `icmp time exceeded` típusú üzenetben azonnal közöl is. A `time exceeded` üzenet tartalmazza a hibát kiváltó UDP-csomag elejét, ezáltal a küldő azonosítani tudja. A csomag feladója a legelső forgalomirányító berendezés, így annak címét egyszerűen visszakaptuk. A `time exceeded` csomag vétele után a program az előző lépéseket ismétli, csak folyamatosan növelt `ttl`, valamint a célkapuértékkel, amíg csak a célszámítógépet el nem éri. Ezt szemlélteti a 3. lista (26. CD Magazin/Konnyu).
Mint láttuk, a közbenső forgalomirányító berendezések általában a `time exceeded` üzenetet küldik vissza. A célgépig eljutó csomagnál más a helyzet. A `traceroute` alkalmazást úgy írták meg, hogy általában kihasználatlan UDP-kapukat címezzen, hiszen ilyenkor `icmp port unreachable` üzenet jön vissza. Amennyiben a célrendszer nem érhető el, úgy `icmp destination unreachable` üzenet is visszaérkezhet. A művelet sikerét nagymértékben befolyásolhatják az útközben elhelyezett csomagszűrő berendezések. A `tracerote` engedélyezése a tűzfalon kissé összetettebb, azonban megoldható. Amennyiben engedélyezni szeretnénk, hogy a tűzfalunkig használhassák a `traceroute`-ot, a 33434..33523 `udp` tartományra érkező csomagokat utasítsuk el egy `icmp port unreachable` üzenet segítségével (ezt teszi az IP Chains `REJECT` művelete is). Ha azt szeretnénk, hogy belső gépeink `traceroute` segítségével vizsgálhassák a külső gépeket, tűzfalunkon a fenti tartományra engedélyezni kell a kimenő UDP-csomagokat, valamint a bejövő `icmp time exceeded` és a `icmp dest unreachable` csomagokat is. Érdemes megfigyelni, hogy kimenő csomagjaink forráskapuja 32 768 feletti. Ennek oka, hogy a unixos `traceroute` a `pid`-jének (process id – folyamatazonosító) legalsó 15 bitjét adja a 32 768-hoz, és így alakul ki a tényleges forráskapu. Az ehhez szükséges tűzfalbeállítások az alábbiak:

- 2.2.x-es rendszermagsorozatnál:

```
ipchains -A input -i eth0 -p udp
➥-s 0.0.0.0/0 32768:
➥-d 0.0.0.0/0 33434:33523 -j ACCEPT
ipchains -A input -i eth1 -p udp -d
➥0.0.0.0/0 33434:33523 -j REJECT
ipchains -A input -i eth1 -p icmp
➥--icmp-type destination-unreachable
➥-j ACCEPT
ipchains -A input -i eth1 -p icmp
➥--icmp-type time-exceeded -j ACCEPT
```

- 2.4.x-es rendszermagsorozatnál (Netfilter használata mellett):

```
iptables -A FORWARD -i eth0 -p udp -s
➥0.0.0.0/0 --sport 32768: -d 0.0.0.0/0
➥--dport 33434:33523 -j ACCEPT
iptables -A FORWARD -i eth1 -p udp -d
➥0.0.0.0/0 --dport 33434:33523 -j REJECT
iptables -A FORWARD -i eth1 -p icmp
➥--icmp-type destination-unreachable
➥-j ACCEPT
iptables -A FORWARD -i eth1 -p icmp
➥--icmp-type time-exceeded -j ACCEPT
```

1.a lista Ping parancs kimenete

```
[bozo@cheetah bozo]$ ping -c 3 dolphin
Warning: no SO_TIMESTAMP support, falling back to SIOCGSTAMP
PING dolphin.home (10.1.2.251) from 10.1.2.232 : 56(84) bytes of data.
64 bytes from dolphin.home (10.1.2.251): icmp_seq=0 ttl=255 time=1.083 msec
64 bytes from dolphin.home (10.1.2.251): icmp_seq=1 ttl=255 time=930 usec
64 bytes from dolphin.home (10.1.2.251): icmp_seq=2 ttl=255 time=916 usec

--- dolphin.home ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.916/0.976/1.083/0.079 ms
```

1.b lista Ping a vonalon

```
16:21:57.663751 < cheetah > dolphin: icmp: echo request
16:21:57.663910 > dolphin > cheetah: icmp: echo reply
16:21:58.655136 < cheetah > dolphin: icmp: echo request
16:21:58.655305 > dolphin > cheetah: icmp: echo reply
16:21:59.655105 < cheetah > dolphin: icmp: echo request
16:21:59.655272 > dolphin > cheetah: icmp: echo reply
```

Domain

A `domain` protokoll az Internet egyik legalapvetőbb szolgáltatása, nélküle nagyon nehéz lenne a kapcsolattartás. A protokoll célja, hogy lehetővé tegye az „emberi fogyasztásra alkalmas” tartománynevek leképezését IP-címekre, valamint az IP-címek visszaalakítását tartománynevekké. Hathatós támogatást nyújt az elektronikus levelezéshez is. Maga a protokoll meglehetősen összetett, ezért csak a fontosabb részeit ismertetjük a teljesség igénye nélkül.
A DNS (Domain Name System) információinak alapegysége az úgynevezett RR (Resource Record). Egy RR tartalma lehet például egy név–IP-cím összerendelés. Az RR túl kis egység ahhoz, hogy hatékonyan kezelni lehessen, ezért az RR-eket zónákba szervezték. A zóna az RR-ek hierarchikus

© Kiskapu Kft. Minden jog fenntartva

2.a lista Üzenetszórt ping, duplikátumok jelölése

```
[bozo@cheetah bozo]$ sudo ping -b -c 3 10.1.2.255
WARNING: pinging broadcast address
Warning: no SO_TIMESTAMP support, falling back to SIOCGSTAMP
PING 10.1.2.255 (10.1.2.255) from 10.1.2.232 : 56(84) bytes of data.
64 bytes from 10.1.2.232: icmp_seq=0 ttl=255 time=272 usec
64 bytes from 10.1.2.251: icmp_seq=0 ttl=255 time=1.016 msec (DUP!)
64 bytes from 10.1.2.250: icmp_seq=0 ttl=255 time=1.373 msec (DUP!)
64 bytes from 10.1.2.232: icmp_seq=1 ttl=255 time=200 usec
64 bytes from 10.1.2.251: icmp_seq=1 ttl=255 time=952 usec (DUP!)
64 bytes from 10.1.2.250: icmp_seq=1 ttl=255 time=1.068 msec (DUP!)
64 bytes from 10.1.2.232: icmp_seq=2 ttl=255 time=179 usec

--- 10.1.2.255 ping statistics ---
3 packets transmitted, 3 packets received, +4 duplicates, 0% packet loss
round-trip min/avg/max/mdev = 0.179/0.722/1.373/0.456 ms
```

2.b lista Üzenetszórt ping a vonalon

```
16:23:56.581375 B cheetah > 10.1.2.255: icmp: echo request (DF)
16:23:56.581551 > dolphin > cheetah: icmp: echo reply
16:23:56.582350 P hawk > cheetah: icmp: echo reply
16:23:57.573525 B cheetah > 10.1.2.255: icmp: echo request (DF)
16:23:57.573694 > dolphin > cheetah: icmp: echo reply
16:23:57.574256 P hawk > cheetah: icmp: echo reply
16:23:58.573493 B cheetah > 10.1.2.255: icmp: echo request (DF)
16:23:58.573661 > dolphin > cheetah: icmp: echo reply
16:23:58.574277 P hawk > cheetah: icmp: echo reply
```

halmaza egészen a fa leveleiig vagy egy másik zóna határáig. A zónák az Interneten elosztva és redundánsan kerülnek tárolásra. Minden zónához létezik egy kitüntetett kiszolgáló, ahol a zóna mesterpéldánya található. Ez a kiszolgáló a zóna elsődleges névkiszolgálója. A redundancia miatt fontos, hogy a zónára tartalék kiszolgálók is jussanak, hiszen a zónára vonatkozó adatok akkor is szükségesek, ha az elsődleges névkiszolgáló valami miatt nem érhető el (vonalhiba, túlterhelés vagy működésképtelen névkiszolgáló következtében). Minden zónához legalább egy tartalék névkiszolgálót kötelező bejegyezni. Ezeket hívjuk az adott zóna másodlagos névkiszolgálójának. Természetesen egy névkiszolgáló több zónának is az elsődleges forrása lehet, ugyanakkor más zónák másodlagos névkiszolgálójaként is működhet. A másodlagos kiszolgáló a zónában meghatározott időnként ellenőrzi az elsődleges kiszolgálót, és ha a zóna megváltozott, azt letölti onnan. A teljes zóna letöltését zónatranszfernek nevezik.

Az adott zóna elsődleges kiszolgálója és másodlagos kiszolgálói (amennyiben a nála levő másolat elég friss) teljes bizonyossággal válaszolhatnak a zónát érintő kérdésre. Ezt úgy nevezzük, hogy az adott zóna ***authoritatív***. Mit is jelent ez a fogalom és miért is kell ilyenekkel foglalkoznunk? Minden DNS-kiszolgáló (az Internet hatékony működése érdekében) a lekért adatokat bizonyos ideig tárolja, így a kérdezőknek tovább is tudja küldeni.

Most tekintsük át, hogy miként is működik mindez a gyakorlatban! Valaki például meg szeretné nézni a ***www.kiskapu.hu*** weblapot, de nem tudja a címét. Minden helyesen beállított számítógépnek rendelkezésére áll egy vagy több DNS-kiszolgáló címe, amit valamilyen módon megadtak neki. Ezek a névkiszolgálók általában a felhasználó hálózatán helyezkednek el vagy az internetszolgáltatójának névkiszolgálóiról van szó. A lényeg, hogy a felhasználó számítógépéhez közel helyezkednek el. Amikor a felhasználó számítógépe nevet vagy IP-címet szeretne feloldani, a listán szereplő legelső kiszolgálóhoz fordul. Amennyiben elérhetetlen, a soron következővel próbálkozik (természetesen a folyamat kissé bonyolultabb, de erre most nem térünk ki). Az egyszerűség kedvéért tekintsük úgy, hogy csak egy ilyen kiszolgáló létezik. A felhasználó számítógépe kérést küld a beállított névkiszolgálónak, melyben a ***www.kiskapu.hu*** névhez tartozó IP-címe(ke)t szeretné megtudni. Amennyiben a névkiszolgáló ismeri a kérdésre a választ (például az adat rendelkezésre áll a gyorstárában (cache)), visszaküldi az ügyfélnek, és a folyamat lezárul. Amennyiben a névkiszolgáló gyorstára teljesen üres, nem tehetünk mást, minthogy „elindulunk a kályhától". Ebben az esetben a kiszolgáló visszaküldi az üzenetet, miszerint a válasz nem található, és megadja az Internet `root` DNS-kiszolgálóinak a címét. Ezekután az ügyfél tőlük is megkérdezi, amire kíváncsi. Nem valószínű, hogy a `root` DNS-kiszolgálók személyesen ismernék a ***www.kiskapu.hu*** címét, így szintén a „nincs találat" választ küldik vissza – a ***.hu*** névkiszolgálóinak listájával kiegészítve. A válasz vétele után a felhasználó számítógépe a ***.hu*** névkiszolgálóihoz fordul, ahol szintén elutasítják, azonban a válaszban már a ***kiskapu.hu*** névkiszolgálóinak címét kapja meg. Az utolsó körben a felhasználó számítógépe a ***kiskapu.hu*** névkiszolgálóihoz fordul, ahol megkapja a hőn áhított adatot. Amennyiben a kívánt név nem létezik (például elgépeltük, és ***wwww.kiskapu.hu***-t írtunk), a ***kiskapu.hu*** zóna névkiszolgálója szintén elutasít minket, de azt is közli, hogy ez az adat authoritatív (azaz a gép tényleg nem létezik).

A fenti folyamat kissé vadregényesnek tűnik, és a hatékonyság csöppnyi szikrája sem lelhető fel benne. Ezen a gondon segít, hogy a névkiszolgálók a kapott adatokat bizonyos ideig megőrzik és felhasználják. Könnyen lehet, hogy szolgáltatónk DNS-kiszolgálója tisztában van a ***www.kiskapu.hu*** címével vagy legalább a ***kiskapu.hu*** zónával, ezért a folyamat jóval rövidebb. A protokoll tervezői azonban egyszerűbb megvalósításokra is gondoltak, ezért az úgynevezett rekurzív lekérdezést is megvalósították. Ebben az esetben névkiszolgálónk a fenti folyamatot maga végzi el, a felhasználó számítógépe már csak a végső választ kapja meg. (Ez jelen esetben nem

„42” – az indokért lásd ***Douglas Adams*** „Galaxis útikalauz stopposoknak” c. regénysorozatát). Az IP-cím és tartománynév összerendelést egy ügyes trükk segítségével oldották meg: az IP-címeket is beforgatták a rendes `namespace`-be. Az IP-cím bájtjait fordított sorrendben írjuk le, majd az `in-addr.arpa` tartományt tesszük mögé. Például a `192.168.1.2` cím keresésekor a `2.1.168.192.in-addr.arpa` névre keresünk.
A `domain` protokoll UDP- és TCP-kaput is használhat egyszerre. Az UDP a leggyakrabban használt protokoll, hiszen ebben az esetben a kapcsolat felépítése és lebontása elmarad. Az UDP-válaszcsomagban azonban a nagyobb válaszok nem férnek el, ezért ebben az esetben TCP-t kell használnunk.
A zónatranszfer kötelező módon TCP-t használ. Nagyon fontos, hogy mindkét elérést lehetővé tegyük, ellenkező esetben majd megfoghatatlannak tűnő hibáknak lehetünk a tanúi.
A vonali forgalom elemzésekor megfigyelhető, hogy az egyszerű felhasználói gépek a DNS-kiszolgálóktól eltérően működnek. Míg az UDP-kérések az egyszerű gépek esetén véletlenszerű (1023 feletti) forráskapuról érkeznek, addig a DNS-kiszolgálók általában rögzített forráskaput használnak.
Ez pedig jelentősen leegyszerűsíti a csomagszűrő szabályok létrehozását. A TCP protokoll használata esetén ez a különbség nem létezik: itt mind az egyszerű ügyfél, mind a DNS-kiszolgáló a kérdést 1023 feletti (úgynevezett non privileged) kapuról kezdeményezi. Egy DNS-címkérést és egy zónatranszfert láthatunk a 4. listán (26. CD Magazin/Konnyu).
Amennyiben valamely gépünkön DNS-kiszolgálót futtatunk, ne feledkezzünk meg az UDP-kapu rögzítéséről! Amennyiben a `domain`-kérések rögzített UDP-kapuról mennek ki, csomagszűrőnket sokkal „szorosabbra húzhatjuk”.

NTP

A kiszolgálók és az ügyfélgépek óráinak összehangolása nagyon hasznos dolog. Enélkül a naplóállományok ellenőrzése sokkal bonyolultabb, különösen, ha több rendszerről van szó. Bár utóbbi eset pusztán kellemetlenséggel jár (elvileg ismerhetjük a gépünk órája és a valós idő közötti különbséget, amit helyesbíthetünk is), számos esetben a gépek óráinak összhangtartása elkerülhetetlen. Ilyen lehet valamely osztott állományrendszer használata vagy az időfüggő azonosítás (például ***Kerberos*** vagy ***SecurID***). Most nézzük át, miként is valósítható meg ez a gyakorlatban!
Az egyik legelterjedtebb módszer, hogy a rendszergazda időnként ránéz a karórájára, majd a gép rendszeridejét átállítja. A megoldás számos hátránnyal jár: egyrészt a beállítás nem lesz túl pontos, másrészt az egész művelet kissé esetleges. Az ilyen műveletet a rendszergazdák csak szórványosan szokták elvégezni, emiatt előfordulhat, hogy két állítás között valamikor egy hét, de megeshet, hogy több hónap telik el. Szerencsésebb lehet, ha kapcsoltvonalas szolgáltatás esetén kitárcsázás után, állandó kapcsolat esetén meghatározott időnként egy parancsot indítunk el, amely egy megadott viszonyítási alaphoz hangolja gépünk óráját. Ilyen feladatot lát el például az `rdate` parancs. Az eddigiek folyamán még nem szóltunk a módszerek másik hátrányos tulajdonságáról, hogy az idő elkezd „furcsán viselkedni”. Gondoljunk csak bele, mi történne, ha hirtelen visszaugranánk a múltba! Márpedig ha a rendszerünk órája siet, akkor pontosan ez fog történni, ami a működésében számos hibát okozhat. (Gondoljuk át a `make` parancs működését: a programok fordítását úgy vezérli, hogy a lefordított állomány módosítási idejét összeveti a forrásállomány módosítási idejével. Amennyiben hátraugrunk az időben, a frissen módosított forrás régebbi lehet, mint az előzőleg keletkezett bináris változat!) Ezeket javítandó fejlesztették ki az NTP-t, azaz a Network Time Protocolt.
Az összehangolás két részben valósul meg. Az első rész (választhatóan) a rendszer elindulásakor történik meg, amikor is a rendszer órája rááll a viszonyítási időre (itt ugyanolyan ugrásról van szó, mint a fenti esetben). Ennek célja, hogy a rendszeridő megközelítőleg azonos legyen a viszonyítási idővel. Ez legegyszerűbben az `nptdate` parancs használatával valósítható meg. Ezekután következik a tényleges összehangolási folyamat, valamint az összhangban tartás. Ezt egy egyedi démon valósítja meg, ami az `ntpd` névre hallgat (a régebbi változatát `xntpd`-nek nevezték). Ez a démon folyamatosan méri a rendszeridőnek a viszonyítási időtől való távolságát, valamint a két rendszer közötti késleltetést (általában hálózati viszonyítási forrásokat használunk). Ezek figyelembevételével a gép rendszeróráját sietteti vagy késlelteti. Az idő múlási sebességének állításával elérhető, hogy a két idő közelítsen egymáshoz, ugyanakkor az idő monoton növekedése megmarad, ami a naplózó alrendszer szempontjából nagyon fontos.
Az NTP-protokoll UDP-t használ. Az NTP-kiszolgáló a számítógép 123-as UDP-kapuján vár csomagokra. Az ügyfelek esetén előfordulhat, hogy tetszőleges kapuról (nem csak 1024 alattiakról) indítják a kéréseiket. Amikor két NTP-kiszolgáló kapcsolatba lép egymással, mind a forrás-, mind a célkapu a 123-as. Az NTP-kiszolgálók a hatékonyabb kapcsolattartás érdekében képesek csoportcímzést alkalmazni, ilyenkor a csomag célcíme `224.0.1.1` lesz. Az NTP vonali működését az 5. lista (26. CD Magazin/Konnyu) szemlélteti.
Fontos azonban, hogy tisztában legyünk az önműködő óra-összehangolás veszélyeivel is. Egy támadó átveheti gépünk órája felett a hatalmat, vagy kihasználhatja az óra-összehangolással kapcsolatos programok valamelyikében található biztonsági réseket. Mindkét eset súlyos következményekkel járhat. Ezeket mindig mérlegeljük egy új szolgáltatás használata során. Megoldás lehet, hogy saját időalapot állítunk üzembe, vagy az időt a hálózatról vesszük ugyan, de eltérő megvalósításokat használunk (például az időt közvetlenül egy forgalomirányító berendezés veszi, és mi azzal hangoljuk össze a kiszolgálónkat).

Hivatkozásjegyzék

[1.] Linuxvilág: 2001. február–március 54. oldal – Könnyű álmok (4. rész)
[2.] Linuxvilág: IP-alapok 2001. január 44. oldal – Könnyű álmok 3. rész

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

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

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

A NIS és az NFS (2. rész)

Folytatjuk ismerkedésünket a központi felhasználókezelés két segédeszközével.

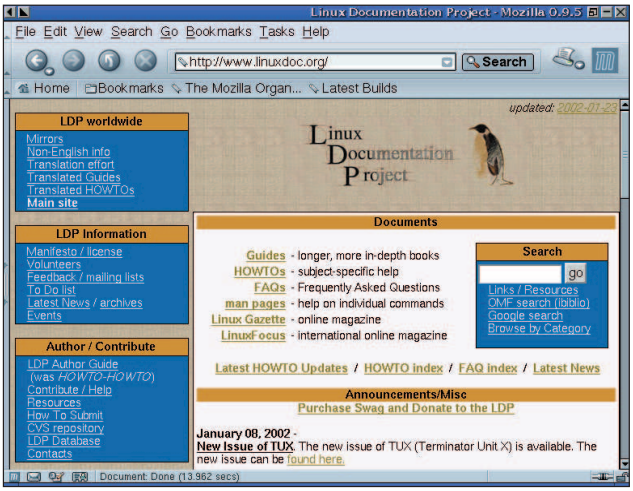
Múlt havi írásunk végére érve a NIS segítségével már bármelyik gépen be tudtunk jelentkezni a helyi hálózaton. Ezt úgynevezett térképállományokkal oldottuk meg, melyek egyszerű, kulcs–érték párokból álló adatbázisok. Így megoszthatjuk */etc/passwd* vagy */etc/group* fájljainkat, viszont a saját könyvtárunk elérése továbbra is gondot jelent. Erre keresünk megoldást cikkünk második részében.
Az NFS-t (Network File System) szintén a Sun dolgozta ki. Démonjai az RPC-t használják, akárcsak a NIS. Ezzel az eszközzel egy távoli gépen található megosztást ugyanúgy csatolhatsz a saját számítógépeden, mintha az illető könyvtár a saját merevlemezeden lenne. Fentebb vázolt feladatunkra úgy keresünk megoldást, hogy NFS-sel megosztjuk a */home* könyvtárat, amelyet minden ügyfél befűz rendszerindításkor. Lássuk, hogyan is lehetséges ez!

Amire szükséged lesz

Az NFS-nek több változata is elérhető, így fontos, hogy különbséget tudj közöttük tenni. Az NFSv2 egy régebbi, beállítási lehetőségekben kevésbé gazdag, ugyanakkor meglehetősen üzembiztos változat. Az NFSv3 szerintem elődjétől csupán a sebességében különbözik, mert személyes tapasztalataim alapján gyorsabbnak bizonyult, mint a 2-es. Végezetül az NFSv4 még fejlesztés alatt áll, a hivatalos rendszermagkiadásban nem is támogatott.
Ügyféloldali NFS-támogatáshoz a rendszermagba bele kell fordítanod a *File systems/Network File Systems/NFS file system support* lehetőséget, ami fordítható modulként is, ekkor a neve *nfs.o* lesz. Az NFSv3 protokollhoz még a *Provide NFSv3 client support* is szükséges, ami ugyancsak a fenti útvonalon érhető el. Ellenőrizd még a `mount` változatszámát a `mount -V` paranccsal, mert ebből legalább a 2.10m-es változatra lesz szükséged.
A kiszolgálóoldal valamivel összetettebb, itt ugyanis választhatsz rendszermag-, illetve alkalmazásszintű megvalósítás között. A rendszermagszintű megvalósítás mellett szól, hogy gyorsabb és a legtöbb terjesztés is csak ezt tartalmazza, vagy legalábbis alapértelmezésként ezt telepíti. A rendszermagban a fentebb megismert útvonalon válaszd ki az *NFS server support*-ot, modulként fordítva a neve *nfsd.o*. Az NFSv3-hoz ismét ki kell választani egy másik lehetőséget, ami a *Provide NFSv3 server support*. Természetesen akkor is szükséged van felhasználói szintű programokra, ha a rendszermagalapú kiszolgálót választod. Debian alatt a csomagok neve: *nfs-common* és *nfs-kernel-server*.

Kiszolgálóoldal – /etc/exports

Most – hogy túl vagy a nehezén – ismerkedj meg a fő beállítóállománnyal! Az NFS-kiszolgálót nem lehet úgy finomhangolni, ahogyan más alkalmazásoknál esetleg megszoktad. Rendszerint beállíthatod, hogy a kiszolgálón ki és mit érhet el, illetve megszabhatsz egy-két apróságot az egyes megosztásoknál. Itt ügyféloldali módosítások szükségesek a hatékony működéshez, de minderről később bővebben szót ejtek.
A */etc/exports* fájl nagyon egyszerű felépítésű: bejegyzésekből



További leírások a linuxdoc oldalon

áll és minden bejegyzés egy sor. A sor # utáni része megjegyzés. Egy bejegyzés a következőképpen fest:

```
helyi_konyvtar
        ➥gepA(tulajdonsag1,tulajdonsag2)
        ➥gepB(tulajdonsag1,tulajdonsag2)
```

A *helyi_konyvtar* egyszerű abszolút útvonal a megosztandó helyi könyvtárra.
A `gepA` DNS-név vagy IP-cím lehet, mindkettő esetében használhatod a `*`-ot és a `?`-et, ha egy tartományt kívánsz lefedni. A következő például teljesen szabályos: `*.turorudi.hu, 192.168.*`. IP-címek esetén még a következő kifejezéseket is használhatod: `192.168.0.0/255.255.0.0, 192.168.0.0/16`. Én az utóbbi jelölést pártolom, mert biztonságosabb, mint a tartománynevek használata, és a hálózati cím is azonnal látszik.
Végezetül a tulajdonság a következők egyike lehet:
`ro` – a megosztás csak olvasható, nem kötelező megadni, mivel ez az alapértelmezés.
`rw` – a megosztás írható és olvasható.
`no_root_squash` – alapértelmezésben a rendszergazda minden NFS-megosztásra történő írása és olvasása `nobody`-ként hajtódik végre. Ezt bírálhatod felül ezzel a beállítással, bár nem ajánlom a használatát, mert komoly biztonsági kockázatot jelent.
`no_subtree_check` – ha egy fájlrendszernek csupán egy alkönyvtárát osztod meg, NFS-kiszolgálóként minden egyes kérésnél ellenőriznie kell, hogy a kért állomány megtalálható-e az alkönyvtárban. Ezt az ellenőrzést kapcsolhatod ki ezzel a lehetőséggel, ami jelentősen gyorsítja a kérések lekezelését.
`sync` – ezzel a lehetőséggel minden egyes fájlművelet csak akkor ér véget, ha a tényleges lemezre írás megtörtént. A segítségével megakadályozhatod, hogy egy esetleges lefagyás

következtében elvesszenek a felhasználók mentett állományai. Ennek árát viszont a teljesítmény oldalán meg kell fizetni: a `sync` mind a kiszolgálót, mind a hálózatot erősen leterheli (lásd még: man 5 exports).
Most készítsünk egyszerű */etc/exports* fájlt! Én a felhasználók saját könyvtárait így osztottam meg:

```
# /etc/exports - NFS megosztasok
/home  192.168.0.0/16(rw,no_subtree_check,sync)
```

Eddig nem volt nehéz, igaz? A kapcsolókkal azért el lehet bíbelődni egy ideig, velem például érdekes módon az történt, hogy amikor a `sync` nem volt megadva, a NIS-en keresztül nemrég bejelentkezett felhasználó saját könyvtárát a helyi rendszergazda látta egy ideig. Ha bekapcsoltam, minden rendesen működött, a helyi rendszergazda semmit nem látott a központi felhasználók könyvtáraiból. Minderre rájönni azonban eltartott egy darabig. Ezt követően indítsd újra az NFS-démont: `/etc/init.d/nfsserver restart`. Ha valamilyen írásmódbeli hibát vétettél a */etc/exports* állományban, azonnal kiírja. A legtöbb esetben csupán annyi a hiba, hogy a gépnév és a lehetőségeket tartalmazó zárójel közé nem szabad szóközt tenni. Sok kezdő rendszergazda keresi órákig a hibát, mire rájön, hogy csak egyetlen szóközzel ütött többet.

Az ügyfél

Az ügyfélnek sincs nehéz dolga, mindössze be kell tűrnie a kiszolgáló megosztott könyvtárát, valahogy így:

```
$ mount -t nfs 192.168.0.254:/home /home
```

Figyelj arra, hogyha az ügyfélgépen a */home*-ban korábban volt valami, a befűzést követően nem fog látszani. Továbbá azt is tartsd szem előtt, hogy a fájlrendszer típusának `nfs`-t adtam meg. Teljesen mindegy, hogy a 192.168.0.254 gépen a */home* ext2fs csereterületen, a ReiserFS fájlrendszeren, vagy valami máson van-e. Az ügyfél ugyanúgy fűzi be, csak a megosztásnevet kell hozzá tudni. Meg lehet-e oldani, hogy rendszerindításkor ez önműködő legyen? Mi sem egyszerűbb: a */etc/fstab*-ba vedd fel a következő sort:

```
192.168.0.254:/home /home nfs defaults 0 0
```

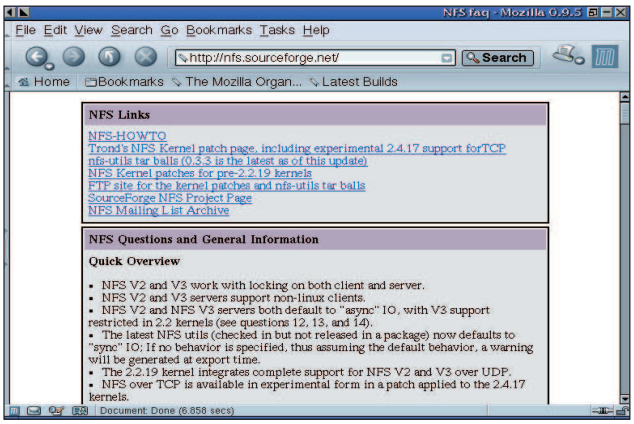
Itt is fontos, hogy a fájlrendszer típusa `nfs` legyen. Az `nfs` jelzésű lemezrészek lesznek utoljára befűzve. Az is igen lényeges, hogy az utolsó két szám 0 legyen, mivel az NFS nem használja őket.

Az utolsó simítások

Fentebb említettem, hogy az NFS-t igazán ügyféloldalon lehet szabályozni. Lássunk néhány fogást ezek közül!
Az NFS-megosztásokat lágyan (soft) és mereven (hard) is be lehet fűzni. Lágy befűzés esetén egy esetleges hiba következtében az NFS-ügyfél jelzi a folyamatnak a hibát és véget ér. Vannak programok, amelyek tudják ezt kezelni, de a legtöbb nem képes rá, így legtöbbször adatvesztés tapasztalható. Ezzel szemben merev befűzés esetén a kernel mindaddig nem ér véget, amíg a kiszolgáló újra elérhetővé nem válik. Ilyenkor alapértelmezésben le sem lehet lőni, csak egy „erős” `kill`-lel. Az `intr` lehetőség viszont megállíthatóvá teszi, így akár CTRL+C-vel, akár egyszerű `kill`-lel sírba lehet küldeni. Módosítsuk tehát a fenti */etc/fstab* bejegyzésünket, hogy a */home*-ot mereven fűzze be:

```
192.168.0.254:/home /home nfs rw,hard,intr 0 0
```

Létezik két lehetőség, az `rsize` és a `wsize`, amelyek azt a méretet határozzák meg, hogy a kiszolgáló, illetve az ügyfél olvasáskor és íráskor mekkora adatot küldjön egyben el egymásnak. Az alapértelmezés 4096 bájt, amely jó átlagérték, de a helyi igények kielégítésére, a hálózat leghatékonyabb kihasználására nem elégséges. Mindenképpen 1024 valamely többszörösében kell gondolkodnunk, amely azonban nem lehet nagyobb 8192-nél.



További hasznos források leírása az NFS-ről

Hogy melyik beállítás a nyerő, csak kísérletezéssel lehet megállapítani: fűzd ki a */home*-ot, majd vissza, és kezdd mondjuk 1024-gyel:

```
$ umount /home
$ mount -t nfs -o rsize=1024,wsize=1024
➥192.168.0.254:/home /home
```

Most mérd le az írás sebességét! Forrásnak érdemes a */dev/null*-t választani, mert ez akkora, „amekkorának akarod”. A próba akkor lesz pontos, ha az állomány meghaladja a kiszolgáló fizikai memóriájának kétszeresét:

```
$ time dd if=/dev/zero of=/home/balazs/teszt
➥bs=1k count=640000
```

A kapott időt jegyezd fel, majd a */home/balazs/teszt*-et írd vissza a */dev/null*-ba. Így az olvasás sebessége is megvan. Az újabb próbához fűzd ki a */home*-ot, ezáltal az átmeneti tárolók is kiürülnek. Lehet, hogy a végén más érték lesz a leghatékonyabb íráshoz, mint olvasáshoz, de ezen ne lepődj meg. Ha minden próbával végeztél, a */etc/fstab*-ot is szerkeszd át, hogy az ügyfél újraindítás után is megfelelő `rsize`, `wsize` értékekkel fűzze be a */home*-ot.

Tehát...?

Tehát készen vagyunk. Központilag, egy helyen veheted fel a felhasználókat, akiknek egy helyen van a saját könyvtáruk, mégis külön gépeken dolgoznak.
Gratulálok, szép munka volt!

Fülöp Balázs
(xut@freemail.hu) 17 éves, imádja a Túró Rudit, a Debian Linuxot és a teheneket. Az ELTE Radnóti Miklós Gyakorlóiskola tanulója immár ötödik éve. Kedvenc írója Slawomir Mro□ek. Leginkább a számítógépes hálózatok biztonsága érdekli.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

A fájlrendszerek titkosítása

Avagy hogyan rejtsünk el egész fájlrendszereket a kíváncsi szemek elől?

Számítógépünk használatba vételének pillanatától kezdve a lassan csordogáló idő folyamán rengeteg olyan adat gyűlik össze, amelyeket nem szívesen osztanánk meg másokkal. Gondolok ez alatt akár a személyes vagy üzleti levelezésünkre, üzleti kimutatásainkra, jelszavakra vagy bármilyen olyan adatra, melynek idegen kézbe kerülése kellemetlen perceket okozhatna számunkra. Néhányan abban a tévhitben élnek, hogy a gondosan megválasztott rendszergazdai jelszó elegendő biztonságot nyújt – őket, sajnos ki kell ábrándítanom. Adatainkat merevlemezünkön kódolatlanul tároljuk, amennyiben valaki megkaparintja, az teljes tartalmával, azaz összes adatunkkal együtt a birtokába kerül. Ilyenkor már nem véd meg sem a tűzfal, sem a rendszergazdai jelszó – kiszolgáltatottak leszünk.

Gondoltál már arra, hogy kódolni kellene azt a sok-sok adatot, hogy csak te férhess hozzá? Netán a jelszavaidat felejted el mindig? Mit szólnál hozzá, ha létezne egy megoldás, amelyhez mindössze egyetlen jelszót kell megjegyezned, és minden adatod biztonságban tudhatnád?

Létezik ilyen megoldás! Fájlrendszerszintű titkosításnak nevezik, és egész sor program létezik hozzá. Mi most kettőt fogunk alaposabban megvizsgálni: a CryptoAPI-t és a CFS-t. Az előbbi megoldás az úgynevezett ***international kernel patch***-ből nőtte ki magát, az utóbbi pedig egy rendszermag-támogatást közvetlenül nem igénylő, NFS-re épülő kiszolgáló. Mindkettő transzparens, azaz teljesen átjárható támogatást nyújt. Csupán használatuk megkezdésekor, a gép indítása után kell megadni egyetlen jelszót, akkortól kezdve a fájlrendszerben a csatolási pontjaik alá írt fájlok kódolva tárolódnak a lemezen, és csupán akkor fejtődnek vissza, ha valamilyen program szeretne hozzájuk férni.

Mielőtt belekezdenénk, egy fontos dologra még felhívnám a figyelmedet: ebben a titkosításban sem lehet vakon megbízni. A rendszeredre ezután is ügyelned kell! Amennyiben valaki, tegyük fel, betör a gépedre az adataidhoz ugyan nem fog tudni jelszó nélkül hozzáférni, viszont megfelelő jogosultságok megszerzésével a legközelebbi alkalommal, amikor beírod a jelszavadat, a betörő ellophatja tőled! Innentől kezdve pedig az összes adatod ismét szabad préda. Tökéletes biztonság tehát nem létezik, de minél jobban megpróbáljuk megközelíteni, annál jobban sikerül leszűkítenünk azoknak a körét, akik sikerrel pályázhatnának adatainkra.

A CryptoAPI

Ez a megoldás annak idején ***international kernel patch***-ként vált ismertté, és támogatottsága időről időre változik, szerencsére mostanában fejlesztője újra felkarolta a projektet. Jelenleg az összes friss változatú rendszermaghoz létezik támogatás, beleértve a 2.5.x-es fejlesztői sorozatot is. Az újabb CryptoAPI-k rendszermag-újrafordítást igényelnek, mivel használatukhoz egy új hurok (loop) eszköz szükségeltetik: a ***cryptoloop***.

A CryptoAPI alkalmazásával adatainkat különböző kódolási eljárásokkal titkosíthatjuk, például a Blowfish-, IDEA-, Serpent-, DES-, 3DES-eljárások segítségével.

A CryptoAPI telepítése

Első lépésben töltsük le a legújabb rendszermag forráskódját és a hozzá kapcsolódó CryptoAPI-foltot (lásd a ***Kapcsolódó címek*** részt). Miután a rendszermagot kibontottuk, lépjünk be a könyvtárába, és másoljuk oda a CryptoAPI-hoz tartozó foltot. Amennyiben a rendszermag bzip2-vel van tömörítve:

```
# tar Ixf kernelforrÆs.tar.bz2 -C /usr/src
```

Ha pedig gzippel:

```
# tar zxf kernelforrÆs.tar.gz -C /usr/src
```

Mindkét parancs létrehoz egy ***linux*** könyvtárat a ***/usr/src***-n belül, alatta a rendszermag forráskódjával. Ezt követően lépjünk be a ***/usr/src/linux*** könyvtárba és telepítsük a CryptoAPI-t a rendszermag forrására:

```
# cd /usr/src
```

2.4.16-os vagy újabb rendszermag esetén:

```
# zcat ~/patch-int-2.4.16.x.gz | patch -p1
```

Régebbi rendszermag esetén pedig:

```
# zcat ~/cryptoapi-2.4.x.x.diff.gz | patch -p1
```

2.4.16-os vagy újabb rendszermag esetén egy másik folt is szükséges:

```
# cat ~/loop-hvr-2.4.16.0.patch | patch -p1
```

(Ha a folt gzip helyett bzip2-vel lett tömörítve, `zcat` helyett a `bzcat` használható.)

Amennyiben a fenti parancsokat sikerrel végrehajtottuk, egy CryptoAPI-val felvértezett rendszermagforrást kapunk, ezt pedig csak le kell fordítanunk. A ***/usr/src/linux*** könyvtárban állva adjuk ki a `make menuconfig` parancsot. Ha a parancs hibát jelez, elképzelhető, hogy a futásához szükséges valamelyik csomag nincs feltelepítve, ez azonban a legtöbb esetben a ***libc6*** vagy az ***ncurses*** fejlesztői könyvtárainak a hiányát jelenti (***-dev***-csomagok). Amennyiben a parancs sikeresen lefut, egy menüben találjuk magunk. Itt lehet megadni, hogy rendszermagunkba mely összetevők forduljanak bele. Minket jelen esetben csak a CryptoAPI-val foglalkozó rész érdekel, így lépjünk be a ***Cryptographic options*** (régebbi változatú CryptoAPI esetén csak ***Crypto options***) menübe. Alapértelmezésben az összes elem modulként van megjelölve, ezeken szükségtelen változtatni. Ha a rendszermagunk egyéb tulajdonságait is beállítottuk, lépjünk ki ebből a menüből és folytassuk a fordítást. Futtassuk le a következő parancsokat:

```
# make dep && make clean && make install
# make modules && make modules_install
```

Ha minden jól ment egy új, működő titkosítási képességekkel bíró rendszermagot kaptunk, melyet a fentebbi `make install` lefuttatásával már működőképes állapotba is hoztunk. Nincs más dolgunk, mint újraindítani a rendszert és behúzni az új rendszermagot.

A CryptoAPI használata

A CryptoAPI használatához szükséges a `losetup` parancs, mely lehetővé teszi a titkosított állományok befűzését. Futtassuk le a `losetup` programot kapcsolók használata nélkül, és ha a megjelenő szövegben nem tesz említést a `-e` vagy a `-encryption` kapcsolókról, minden bizonnyal egy új util-linux csomagot kell beszereznünk, mely a `losetup` egy újabb változatát is tartalmazni fogja, ami remélhetőleg már képes lesz titkosított kötetek kezelésére.
Most már csak azt kell kiválasztani, hogy milyen kódolási eljárást alkalmazzunk. Az egyik legnépszerűbb és legelterjedtebb titkosítás a ***Blowfish***. Ha már korábban is foglalkoztál titkosítással, biztosan ismerősen cseng ***Bruce Schneier*** neve, ő a Blowfish megalkotója és tollából született az alapműnek számító Alkalmazott kriptográfia (Applied Cryptography) című könyv. Alapértelmezésben a CryptoAPI úgynevezett CBC-módban kódolja a befűzött fájlokat. Ez a CBC-mód a ***Cipher Block Chaining***-et takarja, ez azt jelenti, hogy a kódolás során minden blokkot az előző blokk kódolt értékével is kódol. Emiatt később, ha egy blokk, vagyis láncszem elvész vagy megsérül, az azt követő összes többi blokk, azaz a lánc további része is használhatatlanná válik. Ennek köszönhető az is, hogy ha van egy hosszabb szövegünk vagy karaktersorozatunk, amely blokkról blokkra ugyanabból a szövegből áll minden egyes blokk másképpen kódolódik. Ha CBC helyett ECB-t (Electronic Code Book) használunk, a megegyező blokkok ugyanúgy kódolódnának, mivel az ECB minden egyes blokkot külön kódol. Ennek az a hátránya – vagy előnye, attól függően, mire szeretnénk használni –, hogyha a lánc egyik blokkja megsérül, az még nem feltétlenül vonja maga után az egész szöveg használhatatlanná válását.
Térjünk vissza azonban a CBC-hez! A CBC sem tökéletes, ha ugyanis nagy mennyiségű adatot kódolunk, akkor az adat hosszával exponenciálisan nő annak az esélye, hogy ugyanúgy kinéző kódolt blokkok keletkeznek. Minél több ilyen megegyező kódolt blokk jön létre, annál könnyebben törhetővé válik kódolásunk. Emiatt például 64-bites blokkhosszúságú kódolásnál, mint amilyen a Blowfish is, nem ajánlatos átlépni a 2-3 gigabájtos határt. Ha nagyobb mennyiségű adatot szeretnénk kódolni, ajánlatos valamilyen 128-bites blokkhosszúságú kódolást választani, amellyel ennél jóval nagyobb köteteket is biztonsággal kódolhatunk. A ***Serpent*** egy 128-bites blokkhosszúságú kódolási eljárás, melyet az egyik leggyorsabbnak és legkevésbé processzort megterhelőnek tekintenek, így példáinkban ezt fogjuk használni.
Első lépésben töltsük be a titkosítási kötetek kezelését végző modult:

```
# modprobe cryptoloop
```

Erre a modulra mindig szükségünk lesz, ha titkosított kötetekkel szeretnénk dolgozni.
Ezt követően hozzunk létre egy 10 megabájtos fájlt „kodolt” néven, amelyben majd kódolt fájlrendszerünk fog helyet foglalni:

```
# dd if=/dev/zero of=/root/kodolt bs=1M
➥count=10
```

Fűzzük be a fájlt a hurok eszközre:

```
# losetup -e serpent /dev/loop0 /root/kodolt
```

Ezt követően a parancs végrehajtása során rákérdez, hogy milyen hosszúságú kulccsal szeretnénk dolgozni. Válasszuk ki a 128-at, bőven elegendő lesz. Jelszónak pedig adjunk meg egy tetszőleges karaktersorozatot.
A következő lépésben hozzunk létre egy fájlrendszert az eszközön:

```
# mke2fs /dev/loop0
```

Természetesen nemcsak `ext2`-t használhatunk, hanem bármilyen más egyéb, Linux által támogatott fájlrendszert is.
Ha az `mke2fs` elkészült, fűzzük be az eszközt a */**mnt*** alá:

```
# mount /dev/loop0 /mnt
```

Ha belépünk a */**mnt*** könyvtárba láthatjuk, hogy máris ott van egy ***lost+found*** könyvtár, melyet az `mke2fs` hozott létre. Készítsünk egy fájlt kedvenc szövegszerkesztőnkkel és írjunk bele néhány sort, majd mentsük a fájlt és miután kiléptünk a könyvtárból, fűzzük ki a kötetet:

```
# cd /
# umount /dev/loop0
# losetup -d /dev/loop0
```

Ha a `grep`-pel rákeresünk az előbbi szövegre a fájlban, azt tapasztalhatjuk, hogy a fájl nem tartalmaz ilyen szöveget. Ellenben ha a */**dev**/**loop0***-n grepelünk a fájl befűzött állapotában, akkor természetesen a `grep` találatot jelez:

```
# grep -c  sz veg  /root/kodolt /dev/loop0
```

Következő alkalommal, amikor szeretnénk újra hozzáférni a kötethez, csak a `losetup` parancsot kell kiadni ugyanúgy, mint az előbb:

```
#  losetup -e serpent /dev/loop0 /root/kodolt
```

Figyeljünk arra, hogy kulcsnak és jelszónak ugyanazt adjuk meg, mint az első használatnál, máskülönben a kötet rosszul kódolódik és a `mount` paranccsal sem leszünk képesek befűzni egészen addig, míg a kötetet nem fűzzük újra a helyes jelszóval.

Munka titkosított kötetekkel

Ha rendszeresen erre a titkosított fájlrendszerre szeretnéd menteni a dolgaid, a */**mnt*** helyett létrehozhatsz neki más könyvtárat is, illetve a */**etc**/**fstab*** fájlban beállítható néhány alapértelmezés, hogy kevesebbet kelljen gépelni az állandó használat során.
Nézzük meg, miként fest egy általános ***fstab***-bejegyzés kódolt fájlrendszerre:

```
/root/kodolt  /mnt  ext2  defaults,noauto,
➥loop=/dev/loop0,encryption=serpent  0  0
```

Az első oszlopban adjuk meg, hogy hol található a kódolt fájlrendszert tartalmazó fájl, a második oszlopban azt adjuk meg, hogy melyik könyvtárba fűződjön be, az `ext2` pedig értelemszerűen a fájlrendszer típusát jelenti. A `noauto` azt jelenti, hogy a rendszer ne fűzze be önmagától a fájlt rendszer-

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

induláskor. A `loop` kapcsolóval a hurkoló eszközt választjuk ki, az `encryption`-nel pedig a kódolás típusát. A következő két érték közül a második `0` azt jelenti, hogy rendszerinduláskor a fájlrendszert nem kell ellenőrizni.
Ha ezzel megvagyunk, a következő két paranccsal minden beállítást elvégezhetünk:

```
# mount /root/kodolt
# umount /root/kodolt
```

Ilyen esetekben a kötet jelszavát a `mount` parancs fogja kérni. Főleg nagyobb fájlrendszer esetén nem árt, ha időnként lefuttatunk valamilyen fájlrendszer-ellenőrző programot a titkosított kötetre is. Először fűzzük be a kötetet a `losetup` parancscsal a már tanult módon, majd ha `ext2`-t használunk, futtassuk le a következőt:

```
# e2fsck -f /dev/loop0
```

Ismerkedés a CFS-sel

A CFS egészen más elven működik, mint a CryptoAPI. Annyiban hasonlítanak csupán, hogy egyikük sem köt minket egy bizonyos fájlrendszer használatához. Ráadásul a CFS képes az éppen működő fájlrendszeren dolgozni, és egyszerű, felhasználó is tudja kezelni. A rendszer lényege, hogy a saját könyvtárunkban létrehoz egy könyvtárat, amelybe a titkosított fájlok kerülnek. Ebbe a könyvtárba nem szabad közvetlenül írni, helyette a CFS által létrehozott könyvtárba kapunk írási jogot, mely valójában egy NFS kötet, és a saját könyvtárunkon kívül található.

A CFS használa

A CFS-t a ***Kapcsolódó címek*** részben található címről tölthetjük le, vagy ha Debian Linuxot használunk, az `apt-get install cfs` paranccsal is feltelepíthetjük. Telepítés után azonnal megkezdhetjük a munkát, akár egyszerű felhasználóként. Először hozzunk létre egy könyvtárat a CFS-sel, ezt a `cmkdir` paranccsal tehetjük meg:

```
# cd ~
# cmkdir proba
```

Ezután a rendszer megkérdezi, hogy milyen jelszót szeretnénk a könyvtárhoz rendelni. A jelszónak legalább 16 karakteresnek kell lennie és nem árt, ha alaposan az emlékezetünkbe vésük. A létrejövő „proba” könyvtárban egy sor különös ponttal kezdődő bejegyzést találunk, melyet a CFS hozott létre saját belső használatra. Ezzel a könyvtárral a továbbiakban nem is kell törődnünk, végképp nem szabad fájlokat létrehozni benne, mert azok ilyen módon nem kódolódnak le. Helyette, ha mégis szeretnénk használni, fűzzük be a könyvtárat a `cattach` paranccsal:

```
# cattach proba
```

Ezt követően, ha belépünk a */**crypt**/**proba*** könyvtárba (Debian esetén */**var**/**cfs**/**proba***), az ott létrejövő fájlokat a rendszer kódolja, és titkosított névvel a saját könyvtárunkban lévő „proba” könyvtárba menti. Fontos tehát, hogy nem a saját könyvtárunkban lévő könyvtárban, hanem a */**crypt*** (illetve */**var**/**cfs***) alatt lévő könyvtárban kell dolgoznunk.
Ha végeztünk a munkával a könyvtárat a `cdetach` paranccsal fűzhetjük ki:

```
# cdetach proba
```

Már létező könyvtár jelszavát a `cpasswd` paranccsal lehet megváltoztatni, amely példánkban így működne:

```
# cpasswd proba
```

A CFS-ről bővebben

A CFS a kódoláshoz különleges DES-t vagy 3DES-t használ ECB-vel. Mivel a CFS az állományok visszafejtéséhez felhasználja a fájlleírójának (i-node) számát és a felhasználó UID-ját is, ezért a befűzött könyvtárakhoz csak az a felhasználó férhet hozzá, aki eredetileg létrehozta azokat, illetve visszafejtés nélkül a fájlokat másolni sem lehet, mivel az új fájloknak más lesz a fájl leírószáma és ezáltal megfejthetetlenné válnak. Ez azt jelenti, hogy még a könyvtár befűzött állapotában sem fér hozzá senki a fájljainkhoz, még a rendszergazdát sem. Ez nem egy életbiztosítás, mivel mint tudjuk, a rendszergazda jelszó nélkül válhat tetszőleges felhasználóvá, úgy pedig már akadály nélkül beletekinthet bármibe, ha a könyvtár éppen be van fűzve. Nem árt tudni azt sem, hogy a `cdetach` parancs nem tartalmaz semmilyen védelmet, így bárki leválaszthat bármely más felhasználó által használt könyvtárat.

Összegzés

Az eddigiekben a CryptoAPI-val és a CFS-sel ismerkedhettünk meg. Mindkettőnek egyaránt megvannak az előnyei és a hátrányai, ki-ki maga döntse el, melyik felel meg jobban az ő igényeinek. Ha nem elégedtél meg ennyivel, elárulom, létezik még néhány választható megoldás, mint például a BestCrypt vagy a LoopAES, melyekkel lehet, hogy a későbbiekben még foglalkozunk.

Gludovátz Gábor
(ggabor@sopron.hu)
1996 óta foglalkozik Linux-rendszerekkel. Egyik kedvenc időtöltése a programozás, jelenleg éppen egy C++-ban írt KDE-s játékon dolgozik, de szívesen kódol Pythonban és PHP-ben is. Honlapja ➲ http://www.sopron.hu/~ggabor/

Kapcsolódó címek

A CryptoAPI a 2.4.16-nál régebbi rendszermagokhoz
➲ ftp://ftp.hu.kernel.org/pub/linux/kerne/people/hvr/
A frissítésekhez csak próbaváltozatú CryptoAPI van
➲ ftp://ftp.hu.kernel.org/pub/linux/kerne/people/hvr/testing/
IV-számoló (initialization vector) eljárás javítása, mely a 2.4.16 vagy újabb rendszermagokhoz szükséges
➲ ftp://ftp.hu.kernel.org/pub/linux/kerne/people/hvr/testing/loop-hvr-2.4.16.0.patch
A CFS-t innen töltheted le
➲ http://computing.ee.ethz.ch/sepp/cfs-1.4b2-to.SEPP/
Linux-Crypto levelezési lista
➲ http://mail.nl.linux.org/linux-crypto/
Cryptography-FAQ
➲ http://www.faqs.org/faqs/cryptography-faq/
A *Handbook of Applied Cryptography* című könyv innen tölthető le ➲ http://cacr.math.uwaterloo.ca/hac/

© Kiskapu Kft. Minden jog fenntartva

Bevezetés a Tkinter használatába (3. rész)

Sorozatunk e részében a Tkinterhez létező nagyszerű kiegészítő modullal foglalkozunk, amelynek segítségével az alkalmazásaink írásához szükséges időt még jobban lerövidíthetjük.

Ezzel a modullal még egyszerűbben hozhatunk létre párbeszédablakokat, menüket, gördítősávval ellátott ablakokat, és még egy sor egyéb elemet, mint eddig. A kiegészítés neve PMW, azaz Python Megawidgets. A PMW-t teljes egészében Pythonban írták, és a Tkinterhez hasonlóan teljesen felületfüggetlen. Az elkövetkezőkben néhány gyakran használt PMW-s elemmel fogunk megismerkedni.
Ha egy programban ki szeretnénk használni a PMW képességeit, első lépésben a PMW-modult importálnunk kell a következő utasítással:

```
import Pmw
```

1. lista A balloon.py

```
1  from Tkinter import *
2  import Pmw
3
4  root = Tk()
5
6  status = Pmw.MessageBar(root,
   ➥entry_relief=GROOVE, labelpos=W,
   ➥label_text='Sæg :')
7  status.pack(fill=X, expand=YES,
   ➥side=BOTTOM)
8
9  balloon = Pmw.Balloon(root)
10 balloon.configure
   ➥(statuscommand=status.helpmessage)
11
12 start = Button(root, text='Start')
13 start.pack()
14
15 balloon.bind(start,
   ➥"Program ind tÆsa",
   ➥"A program elind tÆsa egy œj ablakban")
16
17 root.mainloop()
```

Ezt követően a PMW elemeire a `Pmw.elemneve` objektummal hivatkozhatunk.
A PMW-t a ***Kapcsolódó címek*** részben megadott címről tölthetjük le, illetve Debian esetén az `apt-get install python-pmw` parancs kiadásával.

Helyzetérzékeny súgórendszer – Balloon

Már a legegyszerűbb alkalmazások készítése során is felmerül a kérdés, hogy vajon mindenki ugyanolyan könnyedséggel használja-e majd a programot, mint mi magunk. Ha a leírás készítésén már túl vagyunk, okos ötlet lehet helyzetérzékeny súgót készíteni az alkalmazás felületén található ikonokhoz, gombokhoz, legördülő listákhoz és egyéb elemekhez. Ez annyit tesz, hogy amikor a felhasználó az egérrel megáll valamelyik gomb felett, akkor a gomb mellett egy kis mezőben, úgynevezett „lufiban” (buborékban) megjelentetjük a gombhoz tartozó tudnivalókat. Ilyen módon programunk kinézete átláthatóbbá tehető, mivel nem kell az ablakát magyarázó szövegekkel telezsúfolni – ezzel ahelyett, hogy megkönnyítenénk a használatát, inkább túlterheljük és a felhasználót is megijesztjük.
Az *1. listában* látható, hogyan valósítható meg mindez a gyakorlatban.
A listában hivatkozom egy `Pmw.MessageBar` osztályra, amellyel a állapotjelzősor készítjük el, erről az osztályról a későbbiekben még szót ejtünk.
A lufik elkészítése a `Pmw Balloon` osztályával történik. Ez az osztály felelős a állapotsor működéséért is, melynek `bind()` eljárása rendeli a szövegeket az egyes elemekhez. A `bind()` első értéke annak az objektumnak a változója, amelyhez a szöveget rendeljük, második értéke a lufiban megjelenő szöveg, mely a gombon lévő szöveghez hasonlóan még elég tömör, illetve a harmadik értékben a állapotjelzősorban megjelenő szövegen módosíthatunk, ez ugyanis jóval hosszabb is lehet.

2. lista A combobox.py

```
1  from Tkinter import *
2  import Pmw
3
4  def changeChoice(item):
5      choice.configure(text=item)
6
7  root = Tk()
8
9  menuitems = ('else elem', 'masodik elem',
   ➥'harmadik elem')
10
11 choice = Label(root, text='VÆlassz!')
12 choice.pack(side=TOP, expand=YES,
   ➥fill=BOTH)
13
14 items = Pmw.ComboBox(root, label_text=
   ➥'Elemek:', labelpos=W, dropdown=1,
15     selectioncommand=changeChoice,
   ➥scrolledlist_items=menuitems)
16 items.pack(expand=YES, fill=X)
17
18 root.mainloop()
```

© Kiskapu Kft. Minden jog fenntartva

3. lista A dialog.py

```
1  from Tkinter import *
2  import Pmw
3
4  root = Tk()
5
6  info1 = Label(root, text=
   ➥'A dial gusablak mØg nem tØrt vissza')
7  info1.pack()
8
9  dialog = Pmw.Dialog(root,
   ➥buttons=('Igen', 'Nem', 'MØgsem'),
   ➥defaultbutton='Igen',
   ➥title='Dial gusablak')
10 info2 = Label(dialog.interior(), text=
   ➥'Ez egy dial gusablak pØlda PMW-vel',
   ➥pady=20)
11 info2.pack(fill=BOTH, expand=YES, padx=5,
   ➥pady=5)
12 rv = dialog.activate()
13
14 info1.configure(text=rv)
15
16 root.mainloop()
```

4. lista Az entryfield.py

```
1  from Tkinter import *
2  import Pmw
3
4  root = Tk()
5
6  yourname = Pmw.EntryField(root,
   ➥labelpos=W, label_text='Neved:')
7  yourname.pack()
8
9  email = Pmw.EntryField(root,
   ➥labelpos=W, label_text='E-mail c med:')
10 email.pack()
11
12 Pmw.alignlabels((yourname, email))
13
14 root.mainloop()
```

Választás több elem közül – ComboBox

Ha több elem közül kell kiválasztani egyetlen elemet, nagyon helytakarékos és átlátható megoldást jelentenek a `ComboBox`-ok. A *2. listában* egy egyszerű ComboBox-példa látható.
A PMW-s `ComboBox` összetett elem, azaz sok egyéb PMW-s elemhez hasonlóan több elemből áll, jelen esetben kettőből: egy `Label`-ből és magából a `ComboBox`-ból. A lista 14. sorában látható, hogy e összevont elem belső elemeinek a tulajdonságait külön-külön is módosíthatjuk. A `label_text` tulajdonság megváltoztatásával például a beágyazott Label, azaz a címkeobjektum tulajdonságain változtathatunk.
A `labelpos` értékkel határozhatjuk meg, hogy az összevont elemen belül ez a címke hol jelenjen meg.

5. lista A group.py

```
1  from Tkinter import *
2  import Pmw
3
4  root = Tk()
5
6  g1 = Pmw.Group(root, tag_text=
   ➥'Elso csoport')
7  g1.pack(expand=YES, fill=BOTH,
   ➥padx=3, pady=3)
8  g1label = Label(g1.interior(),
   ➥text='Az elso csoport tartalma')
9  g1label.pack(expand=YES, fill=BOTH,
   ➥padx=5, pady=5)
10
11 g2 = Pmw.Group(root, tag_pyclass=None)
12 g2.pack(expand=YES, fill=BOTH,
   ➥padx=3, pady=3)
13 g2label = Label(g2.interior(),
   ➥text='A mÆsodik csoport tartalma')
14 g2label.pack(expand=YES, fill=BOTH,
   ➥padx=5,
pady=5)
15
16 g3 = Pmw.Group(root, tag_pyclass=
   ➥Checkbutton, tag_text='3. csoport')
17 g3.pack(expand=YES, fill=BOTH, padx=
   ➥3, pady=3)
18 g3label = Label(g3.interior(),
   ➥text='A harmadik csoport tartalma')
19 g3label.pack(expand=YES, fill=BOTH,
   ➥padx=5, pady=5)
20
21 root.mainloop()
```

Párbeszédablakok gyorsan – Dialog

Amennyiben a felhasználóval párbeszédablakon keresztül szeretnénk közölni valamit, a PMW `Dialog` osztályát felhasználva egyszerűen megtehetjük.
A párbeszédablak használatára látható példa a *3. listában*.
Ha a megjelenő párbeszédablakban a gombokon kívül egyéb elemet is el szeretnénk elhelyezni, a `Dialog` osztály `interior()` eljárásával hivatkozhatunk a párbeszédablak belsejére. Az ablak megjelenítéséért az `activate()` eljárás felelős, mely egyben a párbeszédablak visszatérési értéke is.

Szövegbevitel előre gyártott címkével – EntryField

Ha sokszor együttesen van szükségünk címkékre és beviteli mezőkre – például egy űrlap kitöltésénél –, jó szolgálatot tehet az `EntryField` osztály. Ez az osztály lényegében egy összevont `Label` és `Entry` elemet takar, kiegészítve néhány PMW-re jellemző szolgáltatással (például beviteli értékellenőrzés). Használata is éppen olyan egyszerű, mint az említett két elemé. Erre példát a *4. listában* találhatunk.
Az `alignlabels()` eljárás segítségével az egymás alatti címkék és beviteli mezők egymás alá igazíthatók. Az eljárás értéke azoknak az elemeknek a listája, amelyeket egymáshoz kell igazítani. Az `alignlabels` után a kettős nyitó zárójel nem elírás, hanem azért szükséges, mert az eljárás egyetlen listaértéket vár. A feladatot így is megoldhattuk volna:

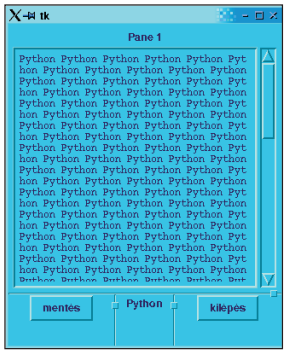
```
widgets = (yourname, email)
Pmw.alignlabels(widgets)
```

Az elemek csoportosítása – Group

Ezzel az összetevővel az ablakban található elemek csoportosítására nyílik lehetőség, így téve az ablak tartalmát még átláthatóbbá. A `Group` elem mindössze egy keretet rajzol elemeink köré, és esetleg valamilyen címkével látja el őket. A tartalma – a `Dialog` elemhez hasonlóan – egy `interior()` nevű eljáráson keresztül érhető el. Az *5. listában* az elem használatára található néhány példa.

Tagolt munkafelület a „Group”-elemmel

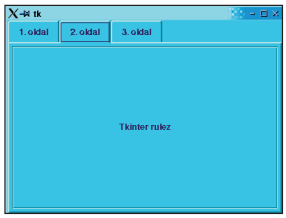
A `g2`-vel jelölt csoport esetében a `tag_pyclass=None` megadása azért szükséges, mert ezzel rendeljük el, hogy a `Group` elem fejlécében az alapértelmezett `Label` összetevőt nem kell megjeleníteni. A `g3`-mal jelölt csoportnál viszont a `tag_pyclass`-nél egy másik objektumtípusra hivatkozunk, így a `Group` fejlécében egy jelölőnégyzet (Checkbox) jelenik meg.



Egy „PanedWidget”-re épülő alkalmazás

Menük készítése – MenuBar

Ezzel az elemmel néhány egyszerű utasítás segítségével programjainkhoz bonyolult, többszintű menüszerkezetet készíthetünk. Az elemet bemutató példaprogram a 6. listában (26. CD Magazin/Tkinter könyvtárban) található.



A „NoteBook”-elem

Az `addmenu()` eljárás első értéke a menü nevét adja meg, míg a második a állapotsor használata esetén szükséges. Az itt megadott szöveg – ha az egérrel megállunk a menüsor adott eleme felett – megjelenik a állapotsoron. Az `addmenuitem()` eljárás első értéke a menüelem gazdájának nevét adja meg, a második pedig azt, hogy milyen típusú elemről van szó: elválasztóelemről vagy rendes szöveges menüelemről.

Státuszsor készítése – MessageBar

Ezzel az elemmel már találkoztunk néhányszor az eddigiek során. A `MessageBar` elsődleges feladata, hogy programunk számára állapotsorként működjön. Az elemeknek eddig a `helpmessage()` nevű eljárását használtuk, mely egyenértékű a `message( help , sz veg )` eljárással, azonban ennek a rövidítésnek köszönhetően visszahívófüggvényként egyszerűbben használhatjuk. Mint az gyanítható volt, nemcsak `help`-típusú üzenetek küldése lehetséges a állapotsorra, hanem léteznek még a `userevent`, `busy`, `systemevent`, `usererror`, `systemerror` üzenettípusok is, amelyek elsőbbségi sorrendje a felsorolással megegyező irányban növekszik. Ha nagyobb elsőbbségű üzenet érkezik és az előző üzenet megjelenítésére szánt idő még nem járt le, a magasabb elsőbbségű üzenet akkor is megjelenik – eltüntetve az előző, kevésbé fontos üzenetet.

Beviteli mezők tagolása – NoteBook

A `NoteBook` a `Group` elemhez hasonló szerepet tölt be, tehát segít a program űrlapjainak a ésszerű tagolásában, de a `Group`-pal ellentétben itt egyszerre csak az egy csoporthoz tartozó elemek látszanak. Az egyes csoportokat a fülekre kattintva érhetjük el. Ha a felhasználónak nagy mennyiségű vagy eltérő feladatú elemekbe kell adatot bevinnie, ajánlott a `NoteBook` felhasználása, mert így elkerülhetjük, hogy a felhasználót felesleges adatokkal árasszuk el. Egy egyszerű `NoteBook`-példa látható a 7. listában (26. CD Magazin/Tkinter könyvtár).

A `NoteBook`-hoz a füleket az `add()` eljárással adhatjuk hozzá, amely az adott fül belterületére mutató változóval tér vissza.

Munkaterület felosztása – PanedWidget, ScrolledText

A `PanedWidget` elem segítségével munkaterületünk egyes részeit oly módon oszthatjuk fel, hogy a felosztáson a felhasználó is változtathasson. Segítségével olyan jól alakítható felületek hozhatók létre, mint amilyen például az egyes levelező-programokban is látható.

A `ScrolledText` egyszerű szövegbeviteli mezőt takar görgetőkkel ellátva, amelyet példánkban egy `PanedWidget`-elemmel vonunk össze. A példa a 8. listában (26. CD Magazin/Tkinter könyvtárban) található.

A `NoteBook`-hoz hasonlóan új munkafelületet a `PanedWidget`-hez is az `add()` eljárással adhatunk hozzá, és a visszatérési értéke használható arra, hogy az adott munkafelületen belül újabb elemeket hozzunk létre.

Összegzés

Az eddig tanultak felhasználásával most már mi magunk is készíthetünk egyszerű Tkinter- vagy PMW-alkalmazásokat. Mindemellett az egyes elemek alapos megismeréséhez olykor szükséges lehet fellapozni egy-egy elektronikus kézikönyvet. A PMW-hez és a Tkinterhez egyaránt nagyon jó, részletekbe menő leírás érhető el az Interneten, címüket a ***Kapcsolódó címek***-nél találod meg.

A cikkhez kapcsolódó további kiegészítések a 26. CD Magazin/Tkinter könyvtárban találhatóak.

Gludovátz Gábor
(ggabor@sopron.hu)
1996 óta foglalkozik Linux-rendszerekkel. Egyik kedvenc időtöltése a programozás, jelenleg éppen egy C++-ban írt KDE-s játékon dolgozik, de szívesen kódol Pythonban és PHP-ben is. Honlapja ➲ http://www.sopron.hu/~ggabor/

Kapcsolódó címek

A Python honlapja ➲ http://www.python.org/
Tkinter-tanfolyam
➲ http://www.pythonware.com/library/tkinter/introduction/
A PMW honlapja, innen tölthető le a leírás és a modul is
➲ http://www.dscpl.com.au/pmw/
A PMW FTP-címe ➲ ftp://ftp.dscpl.com.au/pub/pmw

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

A syslog beállítása

Ha biztos akarsz lenni benne, hogy a rendszerfolyamatok és a fontos alkalmazások naplózzák az eseményeket és az állapotukat, mindenképpen meg kell ismerned a syslogot.

Sokat tehetsz Linux-rendszered biztonságáért, amelynek alapja a mindent magába foglaló, pontos és gondosan figyelt napló. A naplók többféle célt szolgálnak: először is segítenek a hibakeresésben – gyakorlatilag az összes elképzelhető gond esetén, amely a rendszerben vagy az alkalmazásokban keletkezik. Másodszor idejében felhívják a figyelmünket, ha rendszerünk használatával visszaéltek. Harmadszor, ha minden kötél szakad (ez azt jelenti, hogy vagy a rendszer omlik össze, vagy betörtek hozzánk), a naplók létfontosságú bizonyítékokat tartalmazhatnak.

Ez a cikk arról szól, miként naplózhatjuk a minket érdeklő rendszerfolyamatokat, és arról, hogy a fontos alkalmazások naplózzák az eseményeket és állapotukat. Kitűzött célunkat elérhetjük a jól bevált syslog programmal. A syslog adatokat vesz át a rendszermagtól (a `klogd`-n keresztül) vagy egy tetszőleges helyi folyamattól, de akár távoli rendszereken futó folyamatoktól is. Rugalmas, mivel megadhatjuk, hogy mit és hova naplózzon. Az előre beállított syslogtelepítés gyakorlatilag az összes Unix- és Linux-változatban része az alap operációs rendszernek.

Ebben a hónapban teljes mélységében tárgyaljuk a syslog beállítását és használatát, talán annál is részletesebben, mint amennyire elvárod tőlünk. Tapasztalatom szerint ugyanis a Linux-felhasználók túlnyomó többsége, még a rendszergazdák is a syslogot hajlamosak az alapértelmezett beállításokkal használni, esetleg apró módosításokat hajtanak végre. Ez azonban ritkán nevezhető jó ötletnek. Elöljáróban még megemlítenék néhány fontos dolgot: ha valakit tényleg érdekel a részletes és rugalmas naplózás, érdemes kipróbálnia ***Scheidler Balázs syslog-ng*** (syslog new generation) nevű kitűnő programját, amely közel annyira sem elterjedt, mint a syslog. A ***syslog-ng*** programról további ismereteket a ***Kapcsolódó címek*** részben található hivatkozás nyújt.

A syslog beállítása

Minden alkalommal, amikor a `syslogd`, azaz a syslogdémon naplóüzenetet kap, az üzenet típusának és fontosságának megfelelően cselekszik. A */etc/syslog.conf* fájl adja meg az üzenetek típusát és fontosságát, valamint a tevékenységek közötti összefüggést. A fájl minden sora egy vagy több típushoz, illetve fontosságkijelölőhöz tartozó tevékenységet ad meg. A kijelölő egy vagy több típusból és egy fontosságértékből áll.

A következő ***syslog.conf*** fájlban a `mail.notice` a kijelölő és a `/var/log/mail` a tevékenység (azaz „írd ki az üzenetet a */var/log/mail* fájlba”):

```
mail.notice   /var/log/mail
```

A kijelölőn belül a `mail` a típus (üzenetkategória), és a `notice` a fontosság szintje.

Típusok

A típusok egyszerű kategóriák. A Linux a következő típusokat támogatja: ***auth***, ***authprov***, ***cron***, ***démon***, ***kern***, ***lpr***, ***mail***, ***mark***, ***news***, ***syslog***, ***user***, ***UUCP*** és ***local0***-tól ***local7***-ig. Ezek közül néhány magától értetődő, de a következőkről érdemes pár szót ejteni:

- ***auth***: számos rendszerbiztonsággal kapcsolatos esemény használja,
- ***authpriv:*** hozzáférés szabályozásával kapcsolatos üzenetek esetében használatos,
- ***démon:*** rendszerfolyamatok és más démonok használják,
- ***kern:*** a rendszermag üzenetei esetében használatos,
- ***mark***: a syslog által létrehozott üzenetek, amelyek csak az időbélyeget és a `--MARK--` karakterláncot tartalmazzák; a két ilyen jelölés között eltelt percek száma a `syslogd -m [minutes]` kapcsolójával befolyásolható,
- ***user:*** az alapértelmezett típus, ha az alkalmazás vagy a kijelölő nem ad meg mást,
- ***local7:*** a rendszerindítás üzenetei,
- ****:*** helyettesítő, jelentése – „bármely típus”,
- ***none:*** helyettesítő, jelentése – „semelyik típus”.

Fontossági sorrend

A típusok semmilyen viszonyban ncsenek egymással, ezzel ellentétben a fontosságnak sorrendje van. A Linuxban a következő fontossági szinteket különböztetjük meg (sürgősségi sorrendben): ***debug***, ***info***, ***notice***, ***warning***, ***err***, ***crit***, ***alert*** és ***emerg***. Jegyezd meg, hogy egy adott üzenet sürgősségét a programozó határozza meg; a típust és a fontosságot az üzeneteket létrehozó programok állítják be, nem a syslog.

Akárcsak a típusoknál, a `*` és a `none` helyettesítők itt is használhatók. Egy kijelölőn belül csak egy helyettesítő vagy fontosság adható meg. A fontosság előtt szerepelhet az `=` és a `!` módosítók egyike vagy mindkettő.

Amennyiben a kijelölőben egyetlen fontosságot adsz meg (módosítók nélkül), akkor valójában nemcsak azt a fontossági szintet adod meg, hanem a felette lévőket is. Például a `mail.notice` azt jelenti, hogy „az összes ***mail***-típusú üzenet, amelynek a fontossága ***notice*** vagy magasabb”, azaz a fontosság ***notice***, ***warning***, ***err***, ***crit***, ***alert*** vagy ***emerg***.

Ezt a viselkedést a fontosság elé írott `=` (egyenlőslgjel) megszünteti. A `mail.=notice` kijelölő azt jelenti, hogy „az összes ***mail***-típusú üzenet, amelynek fontossága ***notice***”. A fontosságokat negálni is lehet: a `mail.!notice` jelentése: „az összes ***mail***-üzenet, kivéve a ***notice*** fontosságúak”.

© Kiskapu Kft. Minden jog fenntartva

Tevékenységek

A gyakorlatban a legtöbb naplóüzenet fájlba íródik. Ha egy fájl teljes elérési útját megadod a sor tevékenység részében a ***syslog.conf***-ban, akkor a sorra illeszkedő üzenetek hozzáíródnak ahhoz a fájlhoz (ha a fájl nem létezik, a syslog létrehozza). A fenti ***syslog.conf*** sorban arra utasítottuk a syslogot, hogy a megfelelő üzeneteket küldje a ***/var/log/mail*** fájlba.
Az üzeneteket más helyekre is küldheted. A művelet lehet fájl, névvel ellátott csővezeték, eszközfájl, távoli gép vagy egy felhasználó képernyője. A csővezetékeket általában hibakeresési célokra használják. A leggyakrabban használt eszközfájlok a TTY-k, de sokan szívesen küldik a biztonságot érintő adatokat a ***/dev/lp0***-ra, azaz a helyi sornyomtatóra. A kinyomtatott naplót a betörő nem tudja letörölni vagy megváltoztatni, és ez a szerep végre értelmet ad a régi mátrixnyomtatók életének.
A távoli naplózás képessége a syslog egyik leghasznosabb tulajdonsága. Ha a sor tevékenység részében egy `@` (atjellel) bevezetett IP-címet vagy gépnevet adsz meg, a syslog a megfelelő üzeneteket elküldi erre a távoli gépre. Például a

```
*.emerg                 @mothership.mydomain.org
```

sor a `syslogd`-t arra utasítja, hogy minden ***emerg*** fontosságú üzenetet a ***mothership.mydomain.org*** nevű gépre küldjön.
Jegyezd meg, hogy a távoli gépen (esetünkben a mothershipen) a `syslogd` folyamatot a `-r` kapcsolóval kell elindítani, hogy fogadja naplóüzeneteidet, ugyanis a `syslogd` alapértelmezés szerint semmilyen üzenetet nem fogad el távoli rendszerről.
Amennyiben központi naplókiszolgálót szeretnél működtetni, amit nagyon ajánlok, valamilyen módon befolyásolnod kell a bejövő üzenetek hozzáférési jogait. Használnod kell legalább a TCP-burkoló gépelérés-szabályozását (a forrás IP-címe szerint), vagy akár helyi tűzfalszabályokat (IP Chains vagy IP Tables).

Bonyolultabb kijelölők

Egyetlen ***syslog.conf*** kijelölőben több típust is felsorolhatsz vesszővel elválasztva. Bővítsük ki az eredeti ***syslog.conf*** sorunkat, hogy ne csak a ***mail-***, hanem az UUCP-üzeneteket is magába foglalja (továbbra is a ***notice*** és az afeletti fontosságúakat):

```
mail,uucp.notice            /var/log/mail
```

Ez a fontosságra nézvést nem működik. Ne feledd, egy adott kijelölőben csak egyetlen fontosság vagy fontosséghelyettesítő szerepelhet!
Egy sorban viszont több kijelölőt is meg lehet adni, pontosvesszővel elválasztva. Ha egy sor több kijelölőt is tartalmaz, a kiértékelés balról jobbra halad; az általános kijelölőket kell előre venned, utánuk következzenek a különlegesebbek. Gondolj úgy a kijelölőkre, mint a szűrőkre. Az üzenet balról jobbra haladva jut át a soron, először a durvább, majd a finomabb szűrőkön megy keresztül.
Folytatva egysoros példánkat, tegyük fel, hogy továbbra is a ***mail***- és UUCP-üzenetekre vagyunk kíváncsiak, amelyeket a ***/var/log/mail*** fájlba akarunk kiíratni, de ki szeretnénk zárni az ***alert*** fontosságú UUCP-üzeneteket. Sorunk ezután így néz ki:

```
mail,uucp.notice;uucp.!alert      /var/log/mail
```

Valójában a syslog viselkedése nem annyira kiszámítható, mint ahogy a példa sugallja. Ellentmondó kijelölők használatakor, vagy ha a különleges kijelölő megelőzi az általánost, váratlan eredményeket kaphatunk. Ezért pontosabb, ha úgy fogalmazunk, hogy a legjobb eredmény érdekében az általános kijelölőkkel kezdd a sort, majd a kivételekkel (és/vagy más különleges kijelölőkkel) folytasd.
Amennyire lehetséges, ne bonyolítsd túl a dolgokat. A `logger` parancs használatával ***syslog.conf*** szabályaidat kipróbálhatod (olvasd el „A rendszernaplózás kipróbálása a `logger` parancscsal” című részt a cikk vége felé).
Figyeld meg, hogy a második kijelölőben (`uucp.!=alert`) a fontosság előtt a `!=` előtagot használtuk, ami azt jelentette, hogy „nem egyenlő”. Ha ki szerettük volna zárni az összes ***alert*** és nála nagyobb fontosságú UUCP-üzenetet (azaz ***alert***-et és ***emerg***-et), az = jelet elhagytuk volna:

```
mail,uucp.notice;uucp.!alert      /var/log/mail
```

Mi történik egy ***info*** fontosságú UUCP-üzenettel? Illeszkedik a második kijelölőre, tehát a ***/var/log/mail***-be kellene naplózódnia, igaz? A fenti példák alapján azonban nem ez fog történni. Mivel a sor első kijelölője csak a ***notice*** és az annál fontosabb ***mail***- és UUCP-üzeneteket engedi tovább, a kérdéses üzenet nem fog eljutni a második kijelölőig.
Természetesen semmi sem akadályozza meg, hogy létrehozz egy másik sort, amelyben az ***info***-szintű UUCP-üzenetekkel foglalkozol. Akár több ilyen sorod is lehet, ha úgy akarod.
A tűzfalszabályokkal ellentétben itt a ***syslog.conf*** összes során minden naplóüzenet végigfut, és annyi műveletet hajt végre, ahányszor illeszkedik.
Tegyük fel, azt szeretnénk, hogy az ***emerg***-szintű üzeneteket minden bejelentkezett felhasználó megkapja, és ezenkívül a megfelelő alkalmazások naplóiba is bekerüljenek. Az *1. listán* olvashatókhoz hasonló megoldással célt érhetünk. Figyeld meg a `-` (mínuszjel) a fájlba író műveletek előtt – ez arra utasítja a `syslogd`-t, hogy az adott naplófájlt a sorra illeszkedő üzenet kiírása után ne hangolja össze.
Az összehangolás elhagyása növeli a hibák lehetőségét: a naplóüzenetek esetleg töredékesen vagy sehogysem kerülnek be

1. lista syslog.conf példafájl

```
# PØlda syslog.conf fÆjl, amely az  zeneteket
# a mail, kernel Øs  egyØb  csoportokba
# osztja, Øs az emerg-fontossÆgœakat
# minden bejelentkezett felhasznÆl
kØpernyıjØre ki rja

# a legt bb rendszeresemØny a tty10-re
# Øs az xconsole csıbe  r dik, az emerg
# mindenkinek

kern.warn;*.err;authpriv.none
|/dev/console
*.emerg                                 *

# a mail, news (leginkÆbb) Øs rendszermag-,
# illetve tßzfal zenetek a megfelelı
# napl fÆjlba ker ljenek
mail.*      -/var/log/mail
kern.*      -/var/log/kernel_n_firewall

# a t bbit ments k egyetlen fÆjlba
*.*;mail.none      -/var/log/messages
```

© Kiskapu Kft. Minden jog fenntartva

***1. táblázat* – A syslog.conf használatának és értékeinek összefoglalása**

Típusok	Típusok kódja**	Fontosság (növekvő sorrendben)	Fontosság kódjai	Műveletek	
auth	4	none	n/a	/valami/fajl	*(naplózás a megadott fájlba)*
auth-priv	10	debug	7	~/valami/fajl	*(naplózás a megadott fájlba, de nem hangol össze írás után)*
cron	9	ifo	6		
daemon	3	notice	5	/valami/csovezeték (pipe)	*(naplózás a megadott csővezetékbe)*
kern	0	warning	4	/dev/valami/tty_vagy_konzol	*(naplózás megadott konzolra)*
lpr	6	err	3	@tavoli.gepnev vagy IP-cím	*(naplózás a megadott távoli gépre)*
mail	2	crit	2	felhasznalo1, felhasznalo2 stb.	*(naplózás a felhasználó képernyőjére)*
mark	n/a	alert	1		
news	5	emerg	0		
syslog	7	* (minden fontosság)	n/a		
user	1				
uucp	8				
local	(0–7)16–23				
* *(minden típus)*	n/a				

A „!” és az „=” előtagok használata a fontosság előtt

*.notice = minden esemény, amely .notice vagy magasabb fontosságú
*.!notice = egyik esemény sem, amelyik .notice vagy magasabb fontosságú
*.=notice = csak a .notice fontosságú esetek
*.!=notice = nem nézi a .notice fontosságú eseteket

** A számkódokat *syslog.conf*-ban nem szabad Linux-rendszereken használni. Csak a tájékoztatás kedvéért adtuk meg őket, például ha syslog-üzeneteket szeretnénk küldeni a naplókiszolgálóra, és esetleg nem linuxos syslogdémont kell beállítanod, amely csak a számadatokat ismeri, mint például a Cisco IOS.

Mi az a klogd?

Létezik egy démon, amelynek beállításait valószínűleg nem kell megváltoztatnod, de nem árt tudnod róla. Ez a `klogs`, a Linux-rendszermagjának naplózódémona. Ezt a démont rendszerindításkor ugyanaz a parancsfájl indítja el, mint az általános rendszernaplózót (ez lehet a */etc/init.d/syslogd* vagy a */etc/init.d/sysklogd* az általad használt Linux-terjesztéstől függően).
Alapértelmezés szerint a `klogd` minden rendszermagtól érkező naplóüzenetet a rendszernaplózónak továbbít, ezért a legtöbbünknek nem kell a `klogd` miatt aggódnia. A rendszermag üzeneteinek kezelését a `syslogd` beállításait tartalmazó fájl módosításával befolyásolhatod.
A `klogd` egyedülálló naplózóként is elindítható, azaz a rendszermag üzeneteit közvetlenül a konzolra vagy fájlba küldheti. Ráadásul, ha még nem fut démonként, a `klogd` arra is használható, hogy a rendszermag átmeneti naplótárolóit (azaz a legfrissebb rendszermagüzeneteket) kiírja fájlba vagy a képernyőre. A `klogd` ilyen alkalmazása főleg a rendszermag fejlesztői számára hasznos.
A legtöbbünknek elég annyit tudni, hogy a szokásos esetekben a `klogd`-ot nyugodtan békén lehet hagyni (azaz meghagyva az alapbeállításokat és az alapértelmezett indítási módot sem szabad letiltani). Csak annyira emlékezz, hogy ha a syslogot használod Linux alatt, akkor minden rendszermagüzenet először a `klogd`-n megy keresztül.

a fájlba, viszont csökkenti a lemez használatának gyakoriságát, ezért teljesítménynövelő hatású. Ahol gyakori fájlírási műveleteket vársz, azoknál a soroknál használd a - (mínuszjelet).
Az *1. listán* egy bizonyos hasznos ismétlődés látható. A rendszermag figyelmeztetései, valamint az összes ***error***- és annál magasabb szintű üzenet, kivéve az ***authpriv***-üzeneteket, az X-konzol ablakába íródnak. Minden ***emerg***-szintű üzenet nemcsak ide, hanem minden bejelentkezett felhasználó képernyőjére is kiíródik.
Továbbmenve minden ***mail***- és ***kernel***-üzenet a megfelelő naplófájlba íródik. Az összes egyéb üzenet – kivéve a ***mail***-üzeneteket – a */var/log/messages* fájlba íródik.
Az előző példákat abból az alapértelmezett ***syslog.conf*** fájljából vettem át, amelyet a SuSE 7.1 telepített az egyik gépemre. Miért nem felel meg ez az alapértelmezett beállítás? Minek ezt egyáltalán megváltoztatni?
Talán nem kell, azonban valószínűleg érdemes. Az alapértelmezett ***syslog.conf*** fájl a legtöbb esetben egy fontos üzenethez vagy olyan tevékenységet rendel, amely nem hatékonyan hívja fel a figyelmedet (például kiírja az üzenetet a TTY-konzolra, míg te a rendszert csak SSH-n keresztül éred el), vagy bizonyos üzenettípusokat az igényeidhez képest túlságosan alaposan vagy túl felületesen kezel.
Az *1. táblázat* összefoglalja a ***syslog.conf*** nyelvtanát, a típusok értékeit, a fontosság fokozatait és a műveletek fajtáit. Jegyezd meg, hogy a három fő oszlop független egymástól: semmilyen kapcsolat nincs a típusok, a fontosság és a tevékenységek között, azaz bármilyen típusú üzenet bármilyen fontossággal bírhat, és tetszőleges művelet hajtható végre rajta. Szintén lényeges, hogy a típusok és a fontosság számkódjai szigorúan a teljesség kedvéért vannak felsorolva, a ***syslog.conf***-ban ne

2. táblázat – **A syslogd indítási kapcsolói**

Kapcsoló	*Leírás*
`-m` (percek a jelek között)	Ennyi perc telik el két jelölőüzenet között (csak időbélyeget tartalmazó üzenetek, amelyek a naplófájlt nézőpontodtól függően áttekinthetővé vagy kuszává teszik. A 0 érték azt jelenti, hogy nincs jelölés).
`-a` (/további/foglalat)	Segítségével további foglalatok adhatók meg a */dev/log*-on kívül, ahonnan a `syslogd` üzeneteket fogad el.
`-f` (/utvonal/syslog.conf)	Ha nem a */etc/syslog.conf*-ot használjuk, a beállításokat tartalmazó fájl elérési útját itt kell megadni.
`-r`	Távoli gépek syslog-üzeneteinek figyelése.

használd őket. Esetleg találkozhatsz velük a forráskódokban vagy a hálózati forgalom fájlba mentett csomagjaiban.

A syslogd futtatása

Ahogy az alapértelmezett ***syslog.conf*** esetleg nem felel meg az igényeidnek, úgy a syslogd alapértelmezett elindítási módja is változtatásra szorulhat. A 2. *táblázatban* és a következő bekezdésekben néhány olyan `syslogd`-kapcsolót mutatunk be, amelyek különösen érdekesek a biztonság szempontjából, de a teljes listát a sysklogd(8) súgóoldalon láthatod.
Ráadásul arra is célszerű figyelmet fordítanod, hogy amikor a syslogd beállításait és indítási módját megváltoztatod, a `syslogd`-t és a `klogd`-t általában egyszerre kell elindítani és leállítani (ha nem tudod, mi a `klogd`, olvasd el a „Mi az a klogd?” széljegyzetet). Az a legjobb, ha ezeket úgy indítod és állítod le, ahogy a rendszered is teszi; azt javaslom, használd rendszered `syslogd`-, illetve `klogd`-indító parancsfájljait.
A legtöbb Linux-rendszeren ez az indító parancsfájl vagy a */etc/init.d/syslog* vagy a */etc/init.d/sysklog* (a sysklog a „syslog és klogd” rövidítése).
Az első bemutatandó `syslogd` kapcsoló az egyetlen, amelyet a Red Hat 7.x az alapértelmezett */etc/init.d/syslog* parancsfájlban használ, ez az `-m 0`, amely a jelölőüzeneteket tiltja. Ezek az üzenetek csak az időbélyeget és a `--MARK--` karakterláncot tartalmazzák. Ezt számos felhasználó hasznosnak tartja, ha hosszú naplófájlokban kell eligazodni, sokan viszont ellenszenvesnek és feleslegesnek gondolják, hiszen minden üzenetnek van időbélyege.
A jelölőüzenetek bekapcsolásához a `-m` után egy pozitív egész számot kell megadnunk, amely megmondja a `syslogd`-nek, hogy hány percenként küldjön magának jelölőüzenetet. Ne feledd, a jelölőüzenet más típusba tartozik, a ***mark***-ba. Legalább egy kijelölőnek ***mark***-típusú üzenetekre kell vonatkoznia (például ilyen a `mark.`, amely minden ***mark***-típusú üzenetre illeszkedik, vagy a `*.*`, amely minden típusú üzenetre illeszkedik).
Például adjuk meg, hogy a `syslogd` félóránként hozzon létre jelölőüzeneteket, és jegyezze fel őket a */var/log/messages*-be. Először a ***syslog.conf***-hoz a következőhöz hasonló sort add hozzá:

```
mark.*          -/var/log/messages
```

Ezután indítsd el a `syslogd`-t:

```
mylinuxbox:/etc/init.d# ./syslogd -m 30
```

Egy másik hasznos `syslogd`-kapcsoló a `-a [foglalat]`. Ennek segítségével adhatod meg a */dev/log* eszközön kívüli egyéb foglalatokat, amelyről a `syslogd` üzeneteket fogadhat.
Ha már próbáltál valaha biztonságossá tenni egy BIND-ot futtató névkiszolgálót, akkor esetleg használtad a `-a` kapcsolót, hogy a `chroot`-olt `named`-folyamat átadja az üzeneteit a nem `chroot`-olt `syslog`-folyamatnak egy *dev/log* eszközfájlon keresztül. Ebben az esetben a `named` nem éri el a */dev/log*-eszközt, de látja a sajátját, amely például a */var/named/dev/log*. Ezért a következő sort kell a */etc/init.d/syslog* fájlba írni:

```
daemon    syslogd  -m 0  -a  /var/named/dev/log
```

A démonutasítás a sor elején kizárólag a Red Hat indító parancsfájljaira jellemző, a fontos rész ez:

```
syslogd  -m 0  -a  /var/named/dev/log
```

Egynél több `-a` kapcsoló is megadható, például így:

```
syslogd -a /var/named/dev/log
➥-a /var/masikchroot/dev/log
➥-a /megintmasik/dev/log
```

Folytatva a 2. táblázat kapcsolóinak ismertetését, tegyük fel, hogy szeretnéd kipróbálni a ***syslog.conf.test*** fájlban megadott új syslog-beállításokat, de nem kívánod felülírni az eredeti ***syslog.conf***-ot, ahonnan az alapértelmezés szerint a `syslogd` a beállításait veszi. Használd a `-f` kapcsolót, amely után megadhatod a `syslogd` új beállításait tartalmazó fájl nevét:

```
mylinuxbox:/etc/init.d# ./syslogd
➥-f ./syslog.conf.test
```

Már említettük a `-r` kapcsolót, amelynek hatására a `syslogd` távoli gépekről is elfogad üzeneteket, viszont nem beszéltünk még a biztonsági megfontolásokról. Egyrészt a biztonság egyértelműen nő, amennyiben központi naplókiszolgálót használsz, vagy ha bármi olyat csinálsz, ami könnyebbé teszi a naplók kezelését és figyelését.
Másrészt figyelembe kell venned különféle fenyegetéseket. Érzékenyek a naplóadatok? Ha az üzenetek nem megbízható hálózaton utaznak keresztül, és az üzeneteket küldő kiszolgálók belső működését jobb titokban tartani, akkor a kockázat gyakorlatilag nagyobb, mint a haszon (legalábbis mint a `syslogd` hitelesítés nélküli, egyszerű szöveges távoli naplózási folyamatának a haszna).
Amennyiben ez az eset áll fenn, mindenképpen fontold meg a ***syslog-ng*** használatát. A syslog-ng képes a TCP-protokollon keresztül küldeni az üzeneteket, így együtt tud működni az Stunnel, az SSH-val és más programokkal, amelyek a biztonságát nagymértékben növelhetik. Mivel a syslog távoli naplózásra csak a kapcsolat nélküli UDP-protokollt használja, és ennek következtében az üzeneteket nem tudja átküldeni az SSH- vagy Stunnel-alagúton, kevésbé biztonságos, mint a syslog-ng.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Ha az üzenetek nem érzékenyek (legalábbis azok, amelyeket távoli kiszolgálón naplózol), még mindig ott van a szolgáltatás megtagadásának és az üzenethamisításnak a gondja. Amennyiben a `syslogd`-t a `-r` kapcsolóval indítod, minden távoli üzenetet elfogad, és egyáltalán nem vizsgálja sem az üzenet forrását, sem az üzenet tartalmát. Ez a kockázat is úgy csökkenthető a leginkább, ha áttérsz a syslog-ng használatára.

Létezik olyan eszköz, amelyet ha a sysloggal együtt használsz, részben mérsékelhető az érvénytelen távoli üzenetek kockázata: ez a TCP-burkoló. Pontosabban a TCP-burkolók gépelérés-engedélyező tulajdonságának segítségével egyszerűen megadható, hogy melyik gép kapcsolódhat és milyen protokollon keresztül a naplókiszolgálóhoz. A gépelérés-engedélyezőt könnyű becsapni a forrás IP-címének hamisításával (főleg azért, mert a syslog-tranzakciók szigorúan egyirányúak), azonban ez is jobb, mint a semmi, és valószínűleg elegendő ahhoz, hogy megakadályozza a kártékony, de lusta támadókat a syslog megzavarásában.

Ha a te véleményed is ez, szerezd meg és telepítsd a TCP-burkolót (bináris csomagját minden korszerű Linux-terjesztés tartalmazza, közülük számos alapértelmezés szerint telepíti is), a részletes útmutatóért olvasd el a host_access(5) súgóoldalt. Megjegyzendő, hogy bár a program neve mást sugall, a TCP-burkoló gépelérés-engedélyezését az UDP-t használó alkalmazások is használhatják.

Titkos naplókiszolgálók

Lance Spitzner, a Honeynet Project alkotója (➲ http://www.honeynet.org) javasolt egy trükköt, amely csalétekhálózatokon, de talán éles DMZ-kben (lásd Linuxvilág 2001. májusi szám 40. oldal) is használható: ez a titkos naplózás. A trükk segítségével lehetővé válik, hogy egy elosztóra (hub) vagy más osztott eszközre csatlakoztatott gép a naplófájlokat nem IP-címmel azonosított rendszerre küldje, amely látja és elkapja a naplóüzeneteket, de nem érhető el közvetlenül a hálózaton keresztül – ezért a hálózatba behatolónak a naplófájlokat sokkal nehezebb módosítania.

Az ötlet egyszerű: tegyük fel, hogy egy *syslog.conf* műveletben megadsz egy hamis IP-címet, azaz egy olyan IP-címet, amely érvényes a géped helyi hálózatában, csak éppen nem használja semmi, ami `syslogd`-t futtat. Mivel a syslog üzenetei a kapcsolat nélküli (egyirányú) UDP-protokollon keresztül utaznak, a küldő fél semmilyen visszajelzést nem vár, miután üzenetet küldött el.

Tegyük fel továbbá, hogy a DMZ gépei egy osztott eszközre, például egy elosztóra csatlakoznak, és minden hálózaton keresztül küldött syslog-üzenet szétszóródik a helyi hálózaton. Nem szükséges, hogy ezen a helyi hálózaton elhelyezkedő központi naplókiszolgálónak IP-címe legyen, mert a csomagokat passzívan leszippanthatja a `snort`-on vagy más csomagszippantón keresztül (a `tcpdump` erre nem alkalmas, mert csak a csomagok fejlécét figyeli, azonban az adatokat nem).

Magától értetődik, hogy az IP-cím nélküli naplókiszolgáló a szokásos IP-alapú rendszerfelügyeleti eszközökkel nem lesz elérhető, a naplófájlok megnézéséhez hozzá kell férned a konzoljához (hacsak nem nyomtatod ki őket sornyomtatón). Ráadásul nem elég a hamis IP-címet minden DMZ-be tartozó gép *syslog.conf*-jába beírni, minden küldő gépen egy hamis ARP-bejegyzést is be kell jegyezni. Ha nem teszed, a rendszerek hiába próbálnák meghatározni a gép ethernet-címét, amely az adott IP-hez tartozik, nem küldenének semmit.

Ha például azt akarod, hogy egy adott gép tegyen úgy, mintha a 192.168.192.168 hamis címre küldene csomagokat, add meg a @192.168.192.168 műveletet a *syslog.conf* megfelelő sorában vagy soraiban, és add ki a következő parancsot a héjból:

```
arp -s 192.168.192.168 03:03:03:31:33:77
```

Ez nem szükséges, ha a csomagokat a szokásos módon küldöd a naplózó gépre, azaz 192.168.192.168 a `syslogd`-t a `-r` kapcsolóval a futtató gép IP-címére.

2. lista Üzenet létrehozása minden üzenettípushoz minden fontossági szinten

```
#!/bin/bash
#
# A parancsfÆjl minden egyes  zenett pushoz
# minden egyes fontossÆgi szinten lØtrehoz
# egy  zenetet
#
for i in
{auth,authpriv,cron,daemon,kern,lpr,mail,
mark,news,syslog,user,uucp,local0,local1,loc
al2,
local3,local4,local5,local6,local7}

do
       for k in
{debug,info,notice,warning,err,crit,
              alert,emerg}
       do
              logger -p $i.$k "Pr ba zenet,
t pusa $i,
                              fontossÆga $k"
       done
done
```

A rendszernaplózás kipróbálása a „logger” paranccsal

Mielőtt befejeznénk a rendszernaplózó beállításának és használatának témakörét, még beszélnünk kell arról az eszközről, amellyel az új beállításokat kipróbálhatjuk – a használt naplózódémon típusától függetlenül. A `logger` parancssori alkalmazás, amely üzeneteket küld a rendszernaplózónak. Nemcsak állapotvizsgáló eszközként lehet használni, segítségével a héj parancsfájljai is felruházhatók a naplózás képességével.

Itt és most számunkra a diagnosztikai képességek érdekesek (bár gondold csak meg, ezt az eszközt minden fontos parancsfájlban alkalmazni kellene, amelyet rendszeresen futtatsz, különösen azokban, amelyek felügyelet nélkül a `cron`-ból vagy `at`-n keresztül futnak). A `logger` működése legegyszerűbben egy példán keresztül szemléltethető.

Tegyük fel, hogy megváltoztattad a syslog beállításait, és minden ***warn*** fontosságú ***daemon***-üzenetet a */var/log/warnings* fájlba küldesz. Az új ***syslog.conf*** fájlt úgy próbálhatod ki, hogy

© Kiskapu Kft. Minden jog fenntartva

További érdekességek

klogd(8): a `klogd` rendszermagnaplózó programot leíró súgóoldal, amely a nem alapértelmezett használati módokra is kitér (diagnosztika stb.).
logger(1): a logger segédprogramot leíró súgóoldal.
sysklogd(8): a syslog linuxos megvalósítását leíró súgóoldal, amely a `klogd`-vel való kapcsolatra is kitér; továbbá megadja a syslogd indítási kapcsolóinak listáját és leírását
syslog.conf(5): a syslog beállítófájljának lehetőségeit, nyelvtanát és használatát részletesen leíró súgóoldal.
syslog-ng: következő nemzedékbeli rendszernaplózó, amely sokkal hatékonyabb, mint a syslog.
➲ http://www.balabit.hu

először újraindítod a `syslogd`-t és a `klogd`-t, majd kiadod a következő parancsot:

```
mylinuxbox:~# logger -p daemon.warn
➥ Ez csak egy pr ba.
```

Láthatod, hogy a `logger` nyelvtana egyszerű. A `-p` kapcsoló után kell megadni a típust és a fontosságot. A `logger` mindent, ami ez után következik, legyen az kapcsoló vagy kijelölő, az üzenet részének tekint.
Mivel gyorsan gépelek, sokszor használok `while-do` ciklusokat a `bash` parancssorban, így rögtönzött parancsfájlok jönnek létre. A következő `bash` parancssorozat akár közvetlenül beírva, akár parancsfájlként is működik:

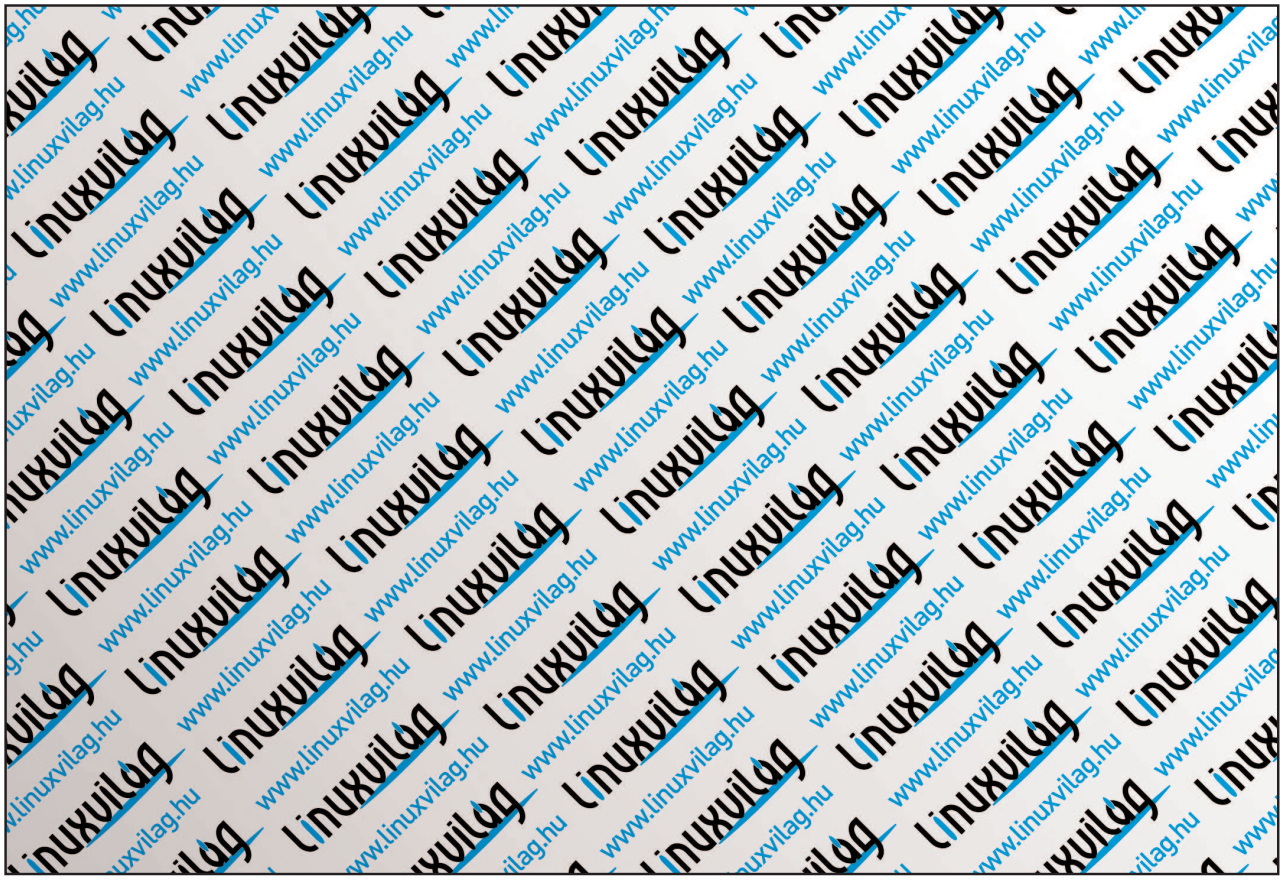
```
mylinuxbox:~# for i in {debug,info,notice,
➥warning,err,crit,alert,emerg}
> do
> logger .p daemon.$i  $i szintß daemon
➥pr ba zenet
> done
```

Ez ***daemon***-típusú próbaüzeneteket küld az összes lehetséges fontossági szinten. A 2. listán olvasható parancsfájl minden lehetséges üzenettípusból az összes szinten egy üzenetet hoz létre.

Összegzés

Remélhetőleg ennyi elég ahhoz, hogy elkezdd a saját syslog-beállításaidat felépíteni, kipróbálni és használni. Kívánom, hogy naplóid legyenek részletesek, bőségesek, jól ellenőrzöttek és unalmasak!

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



© Kiskapu Kft. Minden jog fenntartva

Nyílt forrású webkiszolgálók

Ibrahim három nyílt forrású webkiszolgáló teljesítményét próbálja ki az Ericsson Research telephelyén használt Linux-fürtben.

A kanadai Ericsson Researchnél 2000. januárjában indult az ARIES (Advanced Research on Internet E-Servers) projekt. Célja az volt, hogy egy a Linuxra és nyílt forrású programokra épülő, összetett telekommunikációs feladatok ellátására képes kiszolgálófürt képességeinek bizonyításához szükséges módszereket fejlesszünk ki.
A linuxos fürtök iránt támasztott telekommunikációs követelmények igen szigorúak és a szakágban mindenki jól ismeri az ehhez kapcsolódó nehézségeket. Itt ugyanis biztos elérhetőségre (a rendszernek a hét minden napján, a nap minden órájában elérhetőnek kell lennie), garantált válaszidőre (statisztikailag szavatolt legnagyobb késedelemre), méretezhetőségre és garantált teljesítményre van szükség (a rendszernek képesnek kell lennie arra, hogy időegység alatt legalább egy meghatározott mennyiségű kérelmet feldolgozzon).
Mindemellett a telekommunikációs internetes kiszolgálóknak további fontos feltételeknek is meg kell felelniük, például az internetes forgalom ugrásszerű (félévente százszázalékos) növekedésének, azonban említhetnénk a felhasználók minőségbeli elvárásainak növekedését vagy az igen magas biztonsági követelményeket is.
Ezek az internetes kiszolgálók nagyteljesítményű és nagymértékben méretezhető webkiszolgálókat igényelnek. Mivel az ARIES-nél végzett munka a nyílt forrású programokra épül, már csak egy olyan nyílt forrású webkiszolgáló hiányzott, amely lehetővé tette volna rendszerünk kiépítését.
Az ARIES-nél az egyik célunk az volt, hogy olyan internetes kiszolgálót készítsünk, amely a letöltési sebességek észrevehető csökkenése nélkül képes akár több ezer felhasználót is egyidejűleg kiszolgálni. Ezt a fajta méretezhetőséget legkönnyebben kiszolgálófürtök alkalmazásával érhetjük el. Amikor egy adott honlap egy bizonyos oldalára irányuló kérelem érkezik, azt a legkevésbé terhelt kiszolgálóhoz irányítjuk (egy okos forgalomelosztó megoldással, mely lehet gépi vagy programból megvalósított is).
Három kiszolgáló: az Apache, a Jigsaw és a Tomcat kipróbálása mellett döntöttünk. Az Apache a világ legnépszerűbb webkiszolgálója, az ARIES 2000. évi indulása óta kísérletezünk vele. A Java-alapú Jigsaw jelenleg a kísérleti Linux-fürtön használatos. A Tomcat is Java-alapú webkiszolgáló, a jövőben talán a Jigsaw helyére léphetne, amennyiben nagyobb teljesítményűnek bizonyul.
Az Apache webkiszolgáló hatékony, rugalmas, az előírásainak megfelelő HTTP 1.1 webkiszolgáló. A Netcraft Web Servers felmérése alapján az Apache 1996 áprilisa óta az Internet legnépszerűbb webkiszolgálója. Ez nem meglepő, ha számos előnyös tulajdonságára gondolunk: több felületen használható, megbízható, jól beállítható és forráskódja bárki számára hozzáférhető. Próbáink során az Apache 1.3.14-gyel (ez akkoriban a legfrissebb megbízható változat volt) és az Apache 2.08 alpha változattal kísérleteztünk.
A Jigsaw a W3C nyílt forrású projektje, mely 1996 májusában indult. Ez egy webkiszolgálófelület, mely a HTTP 1.1 bemutató jellegű támogatását és számos további lehetőséget tartalmaz, és egy Javában készített fejlett rendszerre épül.
A Jigsaw-t nem nyilvános megjelenésre, sokkal inkább a különféle módszerekkel való kísérletezés céljából készítették. Próbáinkban a Jigsaw 2.0.1-et és a Java 2 SDK-t használtuk.
A Jigsaw a 8001-es kapunk fogadta a http-kérelmeket.
A Tomcat a Java Servlet 2.2 és a JavaServer Pages 1.1 referenciamegvalósítása. Az Apache felhasználási szerződése alatt fejlesztett Tomcat egy JSP-környezettel rendelkező servletburok, azaz egy futásidejű héj, mely a servleteket a felhasználók nevében kezeli és hívja meg. A Tomcatet önálló kiszolgálóként vagy egy másik webkiszolgáló, például az Apache bővítményeként használhatjuk. Próbáinkban a Tomcat 3.1-es változatát telepítettük önálló kiszolgálóként, mely a kérelmeket a 8080-as kapun fogadta.

1. kép Egy jellegzetes Linux-fürt az Ericsson Research laborjában

A Linux-fürt beállítása

A fentebb említett webkiszolgálók kipróbálása és összehasonlítása érdekében felállítottunk egy az Ericsson Researchnél általánosan használt linuxos fürtöt (*1. kép*).
Ezt a felületet csúcsteljesítményű kiszolgálóalkalmazások számára terveztük. A próbakörnyezet az alábbi elemekből épült fel:

- Nyolc merevlemez nélküli, 500 MHz-en futó és 512 MB memóriával felszerelt Pentium III CompactPCI processzorkártya. A processzorok két felületszerelt hálózati csatlakozóval rendelkeznek, s ehhez jön még egy négycsatlakozós ZNYX ethernetkártya, melyek együttesen több hálózati kapcsolatot tesznek lehetővé.
- Nyolc számítógép ugyanezzel a felépítéssel. Az egyetlen különbség, hogy ezek a gépek egy közös merevlemezblokkal is rendelkeznek, mely három, a legjobb elérhetőség érdekében RAID 1 és RAID 5 rendszerbe kötött 18 GB-os SCSI-merevlemezből állt.
- Fő elemek: a lemezes gépek közül kettő egymás felváltására képes NFS-, NTP-, DHCP- és TFTP-kiszolgálóként működik a többi gép számára. Az egymás kiváltását lehetővé tevő kódot saját magunk fejlesztettük ki, csakúgy, mint a két NFS-kiszolgáló ugyanazon csatolási pontra történő befűzését lehetővé tevő különleges `mount` programot.

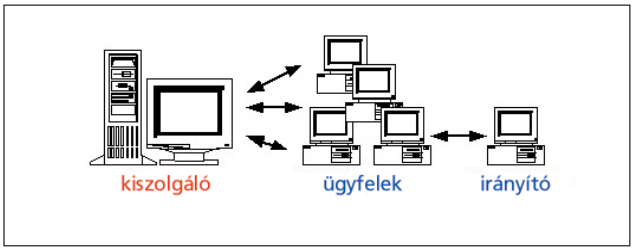
2. kép A próbaegységek

A gépek a helyi hálózatról indulnak el (vagy az 1-es, vagy pedig a 2-es LAN-ról, attól függően, hogy épp melyik működik), majd DHCP-kérelmet intéznek a hálózat minden IP-címe felé. A fő elemek DHCP-választ küldenek és egyúttal továbbítják a gépeknek a hálózati adataikat, így az IP-címek (minden csatoló, az eth0, az eth1, a znb1 és a znb1 számára egy-egy cím), az átjáró, a hálózati maszk, a tartománynév, az indítókiszolgálók IP-címei és az indítófájl nevének beállításához szükséges adatokat. A lemez nélküli gépek ezután letöltik és elindítják a DHCP-beállításfájlban meghatározott indítófájlt, mely a DHCP-kiszolgáló */**tftpboot*** könyvtárában található rendszermagfájl. A következő lépésben a gépek betöltenek egy ramlemezt és indítják az alkalmazáskiszolgálókat, azaz az Apache-t, a Jigsaw-t és a Tomcatet. Egy lemez nélküli kiszolgáló elindítása (az indítás másodpercétől a készenléti jel megjelenéséig) kevesebb mint egy perc alatt lezajlik.
A lemezes gépek szintén a DHCP-beállításfájlban meghatározott indítófájlt töltik le és indítják el (ez a rendszermagfájl is a DHCP-kiszolgáló */**tftpboot*** könyvtárában található). Ezt követően megtörténik a RAID önműködő beállítása és a Red Hat 6.2 testreszabott telepítése. Amikor a gépek működésre készen állnak, elindítják az Apache-, Jigsaw- és Tomcat-kiszolgálókat, mindegyiket külön kapun. A lemezes gépek üzemkész állapotba hozása nagyjából öt perc alatt történik meg (ebbe a RAID1 és a RAID önműködő beállítását, illetve a RedHat 6.2 telepítését is beleszámoltuk).
Próbáink számára a lemezes gépeket (pontosabban közülük hatot, tehát a fő elemeket nem) szintén lemez nélküli gépként indítottuk el az egységes próbakörnyezet kialakítása érdekében.

A próbakörnyezet

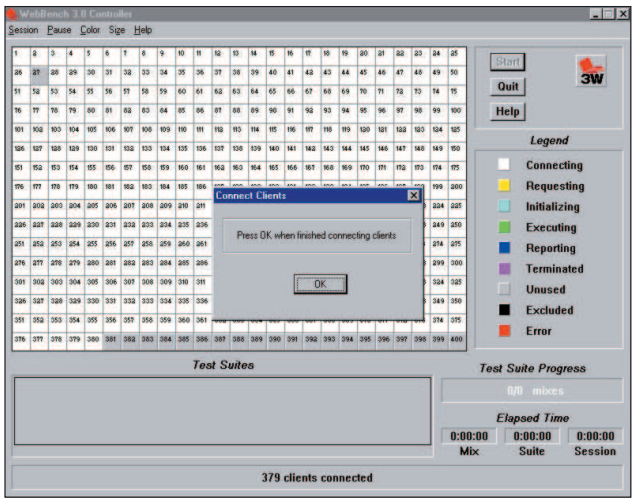
A webkiszolgálók és a kiszolgálóalapú rendszerek teljesítményét számos tényező befolyásolja, például az ügyfél által használt felület és alkalmazás, a kiszolgálón használt felület és alkalmazások, valamint a hálózat és az azon használt protokollok. A Világhálót elemző teljesítménytesztek általában két dologra összpontosulnak: az általános hálózati teljesítményre és a webkiszolgáló alkalmazás- és felület-teljesítményére.
Próbáinkban egy webkérelmeket létrehozó, szabványos mérések végzésére alkalmas módszert használtunk. A forgalmat a WebBench nevű ingyenes programmal (letölthető a ➲ http://www.zdnet.com címről) 16 darab 500 MHz-es, 1 keretheletyet elfoglaló (1U méretű) Intel Celeron gép állította elő (*2. kép*).

Az alap-próbakörnyezetben ügyfélprogramok egy csoportja kérelemfolyamot továbbított a kiszolgálók felé és mérte a rendszer válaszát. A kérelemfolyamot terhelésnek nevezzük. A WebBench a webkiszolgálók teljesítményének mérésére alkalmas módszert biztosít, egy vezérlőből és számos ügyfélprogramból áll (*3. kép*). A vezérlő a WebBench-tesztek beállítását, indítását, leállítását és figyelését végzi, valamint az ügyfélprogramoktól érkező adatok összegyűjtéséért és elemzéséért is felelős.



3. kép A WebBench felépítése

A másik oldalon az ügyfelek elindítják a WebBench-teszteket és kérelmeket küldenek a kiszolgálóhoz. A WebBench az ügyfélgépek használatával a böngészőprogramokat utánozza.
A valódi böngészőkkel ellentétben a kiszolgálóról válaszként kapott fájlokat nem jelenítik meg. Ehelyett amikor egy ügyfélprogram választ kap a kiszolgálótól, akkor a válasszal kapcsolatos minden adatot rögzít, majd azonnal újabb kérelemmel bombázza a kiszolgálót.
A webkiszolgálók teljesítményét sokféleképpen mérhetjük. Próbánkban a kiszolgáló által egy másodperc alatt teljesített kérelmek számát, illetve az egy másodperc alatt küldött bájtok számát rögzítjük (*4. ábra*).



4. kép A WebBench vezérlőablaka

A WebBench a kiszolgáló teljesítményét szabványos próbacsomaggal méri. A próbacsomag tömörített formában szerezhető be, melyet saját magunknak kell kibontanunk a webkiszolgáló sajátkönyvtárában (a webkiszolgáló itt keres minden HTML- és egyéb fájlt). Így létrejön a WBTREE nevű könyvtár, mely 61 MB webdokumentumot tartalmaz, melyeket a WebBench-ügyfelek kérni fognak. Mivel néhány gép nem tartalmaz merevlemezt, a terhelési csomagot az NFS-kiszolgálóra telepítettük, s a webkiszolgálót úgy állítottuk be,

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

hogy a webdokumentumok alapértelmezett keresési útvonala az NFS-kiszolgáló megfelelő könyvtára legyen.
A WebBench beállításának részeként azt is beállítottuk, hogy a próbagépek által gerjesztett forgalom a kiszolgálókon egyenlő mértékben osztódjon szét. Az *5. kép* azt mutatja be, hogy miként állítottuk be az egyes kiszolgálóelemeket és az általuk fogadott forgalmi százalékokat.

CPU	%
CPU 1	25%
CPU 2	25%
CPU 3	25%
CPU 4	25%

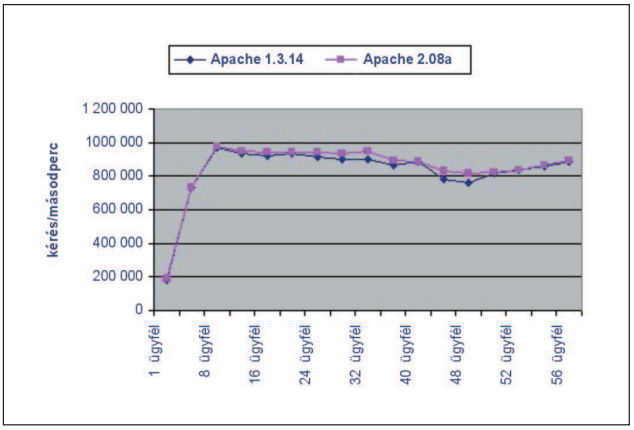
5. kép Egy példa a WebBench beállítására

Ügyfelek száma	Apache 1.3.14	Apache 2.08a
1_ügyfél	182,517	187,333
4_ügyfél	731,067	735,783
8_ügyfél	971,900	978,800
12_ügyfél	931,500	946,067
16_ügyfél	922,767	939,133
20_ügyfél	933,217	943,333
24_ügyfél	913,383	939,050
28_ügyfél	903,333	935,717
32_ügyfél	899,467	945,817
36_ügyfél	867,700	894,633
40_ügyfél	885,933	887,933
44_ügyfél	782,283	828,650
48_ügyfél	759,583	816,017
52_ügyfél	817,150	821,700
56_ügyfél	840,217	838,950
60_ügyfél	856,833	864,133
64_ügyfél	883,067	890,366

6. kép Az Apache 1.3.14/2.08a egygépes próbaadatai

A próbák

A Linux-fürt és a próbakörnyezet felállítása után készen álltunk arra, hogy meghatározzuk a kipróbálásra kerülő rendszereket. Mindhárom webkiszolgálót (Apache, 1.3.14 és 2.08a, Tomcat 3.1, Jigsaw 2.0.1) kipróbáltuk 1, 2, 4, 6, 8, 10 és 12 processzoron. Minden próbánál a betöltendő ramlemezen határoztuk meg, hogy a ramlemez betöltése után mely webkiszolgáló legyen elindítva. A fentiek eredményeképpen négyféle próbát hajtottunk végre, mindegyiket különböző kiszolgálóval és több tesztgéppel.
Cikkemben csak három összehasonlító próbaeredményt mutatunk be: az Apache 1.3.14 és a 2.08a összehasonlítását egy gépen, az Apache 1.3.14 és az Apache 2.08a összehasonlítását nyolc gépen, illetve a Jigsaw 2.0.1 és a Tomcat 3.1 összehasonlítását egy gépen.
Az első teszt során az összes webkiszolgálót egyetlen gépen próbáltuk ki. A WebBench beállításában azt határoztuk meg, hogy az összes ügyfél forgalma egyetlen gépre kerüljön.
A *6. képen* látható mérés 64 ügyfélprogram egyidejű működése mellett készült. Az Apache 1.3.14. átlagosan 828, míg az Apache 2.08a 846 kérelmet volt képes feldolgozni egy másodperc alatt. Utóbbi tehát 2,1 százalékos fölénnyel büszkélkedhet. A *7. kép* az Apache 1.3.14 és 2.08a eredményeit mutatja be. Amint látható, a két kiszolgáló csaknem egyforma teljesítményű.



7. kép Az Apache 1.3.14/2.08a egygépes próbaeredményei

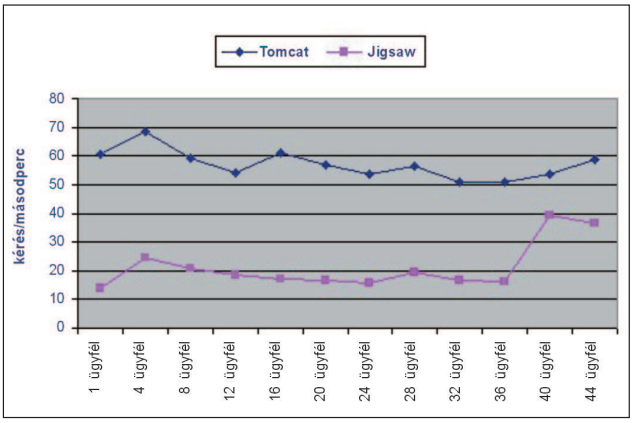
Ügyfelek száma	Tomcat	Jigsaw
1_ügyfél	60,45	14,067
4_ügyfél	68,283	24,296
8_ügyfél	59,283	20,604
12_ügyfél	54,267	18,304
16_ügyfél	60,917	16,908
20_ügyfél	57,067	16,721
24_ügyfél	53,633	15,696
28_ügyfél	56,2	19,533
32_ügyfél	50,866	16,634
36_ügyfél	50,784	16,183
40_ügyfél	53,767	39,358
44_ügyfél	58,816	36,583

8. kép A Tomcat/Jigsaw egygépes próbaadatai

A Java-alapú kiszolgálók (Tomcat és Jigsaw) adatait a *8.* és a *9. képen* láthatjuk. A Jigsaw másodpercenként 39, míg a Tomcat 60 kérelmet volt képes kiszolgálni. A Jigsaw silány teljesítményén nem kell csodálkoznunk, hiszen csupán „kísérleti nyúlnak" tervezték, nem pedig széles körben használt alkalmazásnak. Amikor a próbát nyolc egyidejűleg működő gépre terjesztettük ki, az Apache 2.08a magabiztosabbá vált és az egyidejűleg futó ügyfelek számának növelésével egyre több kérelmet volt képes kiszolgálni, ami a feldolgozott kérelmek számában sem okozott kilengéseket (*10. kép*).
A *11. képen* világosan látszik, hogy ebben a helyzetben az Apache 2.08a mennyivel jobb az 1.3.14-es változatnál. Nyolc gép esetén az Apache 2.08a képes volt megtartani a másodpercenkénti 4434 kérelem–kiszolgálást, míg az Apache 1.3.14 csupán 4152-t teljesített.

A méretezhetőségi próbák eredményei

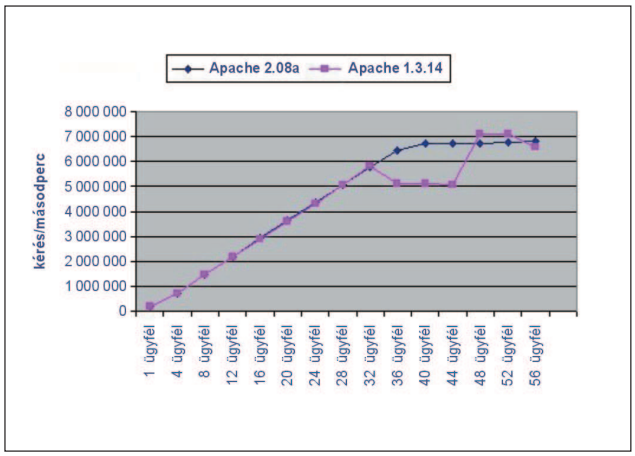
Az 1, 2, 4, 6, 8 és 10 linuxos gépből álló rendszerek grafikonjait gyűjtöttük össze. Minden egyes grafikonnál rögzítettük az adott rendszer által egy másodperc alatt kiszolgált kérelmek legnagyobb számát. Ha ezt elosztjuk a gépek számával, meg-



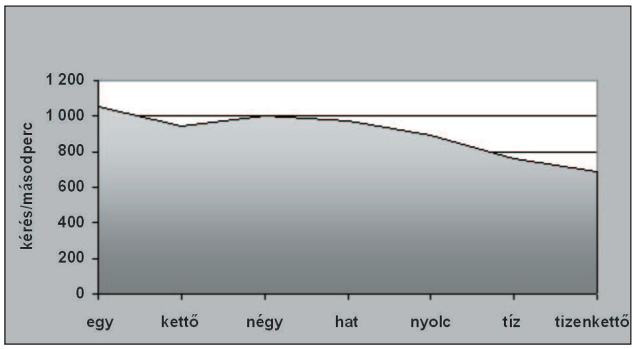
9. kép A Tomcat/Jigsaw egygépes próbaeredményei

Ügyfelek száma	Tomcat	Jigsaw
1_ügyfél	176,417	187,017
4_ügyfél	730,700	728,117
8_ügyfél	1460,483	1446,617
12_ügyfél	2197,383	2162,467
16_ügyfél	2921,216	2869,700
20_ügyfél	3642,583	3598,400
24_ügyfél	4363,667	4300,117
28_ügyfél	5071,800	5088,100
32_ügyfél	5767,833	5821,950
36_ügyfél	6442,083	5124,150
40_ügyfél	6701,150	5119,900
44_ügyfél	6721,317	5085,717
48_ügyfél	6743,100	7092,250
52_ügyfél	6775,300	7092,833
56_ügyfél	6808,617	6570,800

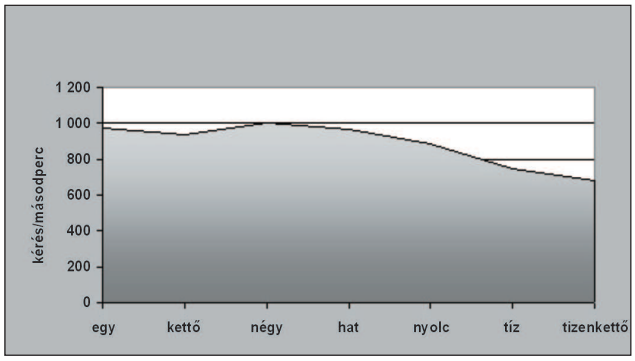
10. kép Az Apache 1.3.14/2.08a nyolcgépes próbaadatai



11. kép Az Apache 1.3.14/2.08a nyolcgépes próbaeredményei



12. kép Az Apache 2.08a méretezhetőségi táblázata



13. kép Az Apache 1.3.14 méretezhetőségi táblázata

kapjuk, hogy az egyes összeállításokban egy gép mekkora teljesítményre képes.

A *12.* és *13. képek* a két Apache-változat gépenkénti átviteli teljesítményét mutatják be, szembesítve a fürtmérettel. A vonal egyik ábrán sem egyenes, ez azt jelenti, hogy a méretezhetőség nem lineáris, azaz nem a leginkább megfelelő.

Ha összegyűjtjük az Apache 1.3.14 és a 2.08a méretezhetőségi adatait (*14. kép*) és grafikonná alakítjuk (*15. kép*), akkor azt figyelhetjük meg, hogy egymáshoz képest mindkét kiszolgáló azonos méretezhetőséggel büszkélkedhet.

Linux-rendszerekben a kiszolgáló mindkét változata egyforma méretezhetőségi adatokkal rendelkezik. Eredményeink alapján az Apache 2.08a két százalékkal jobban méretezhető, mint az 1.3.14-es változat. Mindkét esetben lassú lineáris csökkenést tapasztalhatunk. Nyolc gépnél több elemből álló rendszereknél minél több gépet helyezünk a fürtbe, gépenként annál kisebb teljesítményt kapunk. Nézzük a Java-alapú kiszolgálókat! Bár a Tomcat jobb teljesítményt ért el (több kérelmet dolgozott fel másodpercenként), mint a Jigsaw, ennek ellenére bizonyos méretezhetőségi gondok merültek fel vele kapcsolatban. A *16. képen* az látszik, hogyha több gépet helyezünk el a fürtben, akkor az egyes gépek teljesítménye csökken. Azonban ezt a méretezhetőségcsökkenést számos okkal magyarázhatjuk.

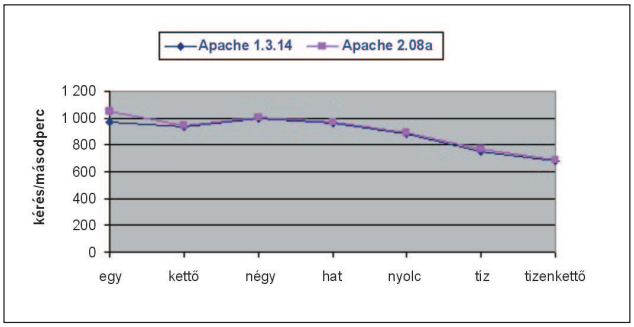
	1.1.14	2.08a
Egy	971	1053
Kettő	937	945
Négy	1000	1003
Hat	964	974
Nyolc	886	892
Tíz	750	764
Tizenkettő	683	685
Átlag	884	902

14. kép A méretezhetőségi adatok összehasonlítása

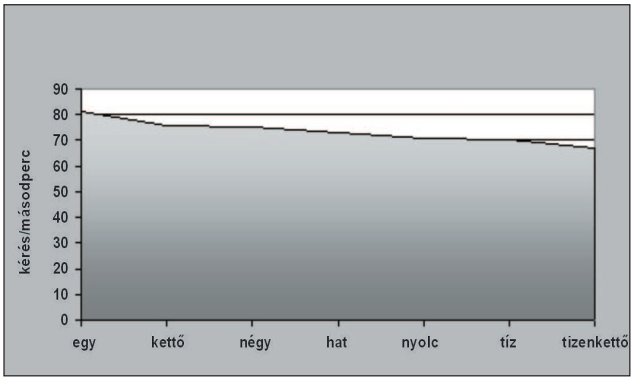
Az eredményeket befolyásoló tényezők

A próbák eredményeit az alábbi tényezők befolyásolhatták:

1. A WebBench próbacsomagját egy NFS-kiszolgálón tettük elérhetővé az ügyfélgépek számára. Így az NFS-nél is jelentkezhet dugulás, amikor másodpercenként több száz ügyfél próbálja meg elérni az NFS-ben tárolt fájlokat.
2. A Jigsaw és a Tomcat Java-alapú webkiszolgáló, így teljesítményük nagymértékben függ Java Virtuális Gép teljesítményétől, melyet egyébként szintén egy NFS-lemezrészről indítottunk el (hiszen a gépek nem tartalmaznak merevlemezt, és a tárhelyet az NFS segítségével osztják meg egymás között).
3. A forgalom gerjesztését csupán 16 Celeron-géppel végezhettük. Az így létrejött forgalom nem biztos, hogy elég volt a gépek túlterheléséhez, és különösen igaz ez az Apache esetében, ahol hatnál több gépet is kipróbáltunk.



15. kép Az Apache 1.3.14 és 2.08a méretezhetőségének összehasonlítása



16. kép A Tomcat méretezhetőségi táblázata

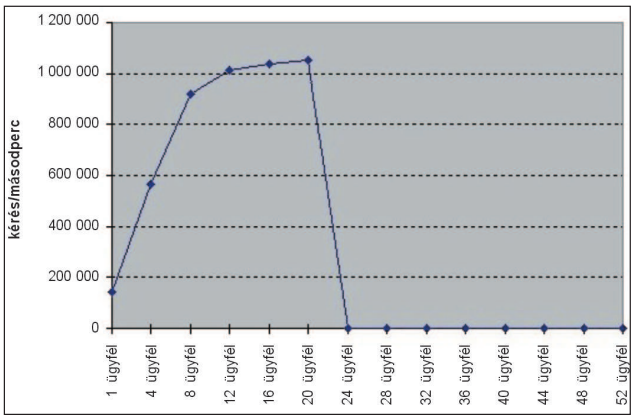
© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

A gondok

Munkánk során számos gonddal szembesültünk: nem megfelelő gépek, kísérleti jellegű alkatrészek és programok, nem támogatott meghajtók és eszközök. Ebben a szakaszban csak azokat a gondokat sorolom fel, melyekkel a próbák végrehajtásához szükséges munkák során találkoztunk.

A ZNYX ethernet linuxos meghajtókkal gondjaink voltak a megbízhatóság tekintetében. Ezek a meghajtók még mindig fejlesztés alatt állnak, a kipróbálás idején még nem voltak piacra dobható állapotban. Ha az egy másodpercre jutó adatátviteli műveletek viszonylag magas számot értek el, a meghajtó egyszerűen lefagyott. Nézzünk például egy próbát, melyet egy Apache 2.08a-t futtató gépen végeztünk el! Amikor a gép terhelése eléri a másodpercenként 1053 kérelemkiszolgálást (az átvitt adatmennyiség 6 044 916 bájt volt másodpercenként), az ethernetmeghajtó lefagyott és a ZNYX-csatlakozókkal elvesztettük a kapcsolatot (*17. kép*).



17. kép A túlterheléstől lefagyott ethernetmeghajtó

A ZNYX munkatársai segítségével számos próbát és hibakeresést végrehajtottunk, végül sikerült kiigazítanunk a meghajtó hibáját és a továbbiakban ezen a téren semmiféle gond nem merült fel.

A fürt indításakor a megoldásra váró második gond az `inetd`-vel kapcsolatos gond volt. Az `inetd` démon a többi rendszerszolgáltatás vezérlőjeként működik. A háttérben maradva figyel a hálózati kapukon érkező kapcsolati kérelmekre. Ha létrejön egy kapcsolat, az `inetd` a megfelelő kapuhoz tartozó szolgáltatás démonjának egy példányát indítja el. A gond a mi esetünkben az volt, hogy az `inetd` ismeretlen okból megtagadta az UDP-kérelmek kiszolgálását és minden ilyen esetben újra kellett indítanunk a démont. Ez a gond még mindig fennáll, az `xinetd` legfrissebb kiadásának használata sem oldotta meg.

A másik nyilvánvaló nehézség az volt, hogy nem voltunk képesek elegendő forgalmat gerjeszteni a próbagépek túlterheléséhez. Több erőre volt szükségünk, mint amennyivel a próbákat végeztük. Tevékenységünk kezdetekor csak 17 gép állt készen (egy vezérlő- és 16 ügyfélgép) a próbákhoz. Ez lehet az egyik oka annak, hogy miért nem voltunk képesek nagyobb méretezhetőséggel számolni. Most azonban a próbakörnyezetet kibővítettük 63 gépre, s a közeljövőben újrafuttatjuk a próbákat és ismét ellenőrizzük a friss eredményeket.

Összegzés

Az ARIES program azért indult, hogy bebizonyítsuk: lehetséges-e telekommunikációs osztályú internetes kiszolgálókat építeni a Linux és nyílt forrású programok felhasználásával. Több Linux-terjesztéssel, web- és adatfolyam-kiszolgálóval, forgalomelosztó és forgalomkiegyenlítő eljárással, a HA Linux és az egymás kiváltására képes rendszerek működtetését lehetővé tevő terjesztett és naplózó fájlrendszerekkel és megoldásokkal (NFS, ethernet, programból megvalósított RAID) kísérleteztünk. A jövőben az ARIES-ben végzett munka arra fog irányulni, hogy a Linux méretezhetőségi képességeit növeljük, hogy a rendszer ezáltal a már felállított webkiszolgáló-alkalmazások mellett más mobil internetes szolgáltatásokat is képes legyen futtatni. A fő cél az lesz, hogy a rendszer a fürt erőforrásait minél hatékonyabban használja ki, illetve hogy a mobil internetes alkalmazásokban oly fontos szerepet játszó biztonság is magasabb színvonalra emelkedjen. Továbbá a projekt a felépített rendszerek képességeinek körét az IPv6-eljárás bevezetésével fogja tovább bővíteni. A Linux-fürtön jelenleg megtalálható három webkiszolgálót megtartjuk. A kipróbált webkiszolgálók méretezhetősége az egyre több gép beillesztése után nem lineárisan növekedett, azonban igen jó teljesítménnyel és közel lineárisan növekvő méretezhetőséggel rendelkeznek (a próbában legfeljebb 12 gépet használhattunk). Jelenleg a webkiszolgálók legfrissebb változatainak telepítése folyik (Apache 2.0.15a, Jigsaw 2.2.0, Tomcat 3.2). Próbáink eredményei alapján megállapíthatjuk, hogy az Apache jelentősen gyorsabb és megbízhatóbban működik, mint a többi webkiszolgáló. Szeretnénk próbákat végrehajtani a 2.0-s változat végleges kiadásával is. Erre a változatra a fejlesztők ígérete szerint a tökéletesen tiszta kód, jól szervezett I/O-rétegezés és jóval magasabb szintű méretezhetőség lesz jellemző.

Köszönetnyilvánítások

A szerző szeretne köszönetet mondani az Ericsson Research Open Architecture Research Development csapatának a cikk közlésének engedélyezéséért, illetve az Ericsson Research Canada munkatársainak, ***March Chatel***-nek és ***Evangeline Paquin***-nek a próbák során nyújtott segítségéért.

Ibrahim F. Haddad (ibrahim.haddad@ericsson.com) az Ericsson Research montreali központú Open Architecture laboratóriumának dolgozik, kutatási területe a telekommunikációs osztályú kiszolgálóelemek valós idejű vizsgálata teljes IP-környezetben. Jelenleg a Concordia Egyetem számítástudományi tanszékének doktorandusza.

Kapcsolódó címek

Apache HTTP Server Project ➲ http://www.apache.org/httpd.html
Jigsaw ➲ http://www.w3c.org/jigsaw
Java Sun ➲ http://java.sun.com
Linux IPv6 ➲ http://www.bieringer.de/linux/IPv6
Linux Kernel Home Page ➲ http://www.kernel.org
Netcraft ➲ http://www.netcraft.com
Red Hat ➲ http://www.redhat.com
ServerWatch ➲ http://serverwatch.internet.com
Tomcat ➲ http://jakarta.apache.org/tomcat
WebBench ➲ http://www.zdnet.com/etestinglabs/stories/benchmarks/0,8829,2326243,00.html
WWW Consortium ➲ http://www.w3c.org
ZNYX ➲ http://www.znyx.com

© Kiskapu Kft. Minden jog fenntartva

PoV-Ray ismeretek (7. rész)

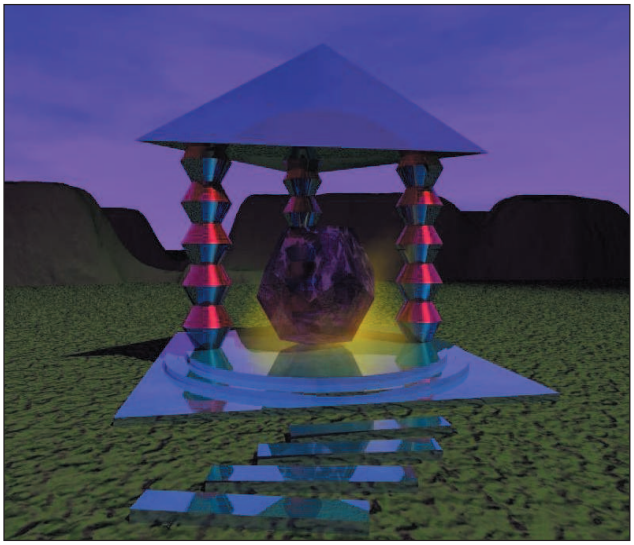
Bevezetés az animációkészítés rejtelmeibe

Mindeddig sikeresen elsajátítottuk azokat az ismereteket, amelyek elegendőek ahhoz, hogy szép állóképeket állítsunk elő, így jogosan merülhet fel a kérdés, hogy a PoV-Ray vajon alkalmas-e animációk készítésére. A válasz természetesen igen, és a cikksorozat utolsó két részében ezeket a lehetőségeket szeretném bemutatni.
A PoV-Rayben nem használhatunk úgynevezett kulcskocka-alapú animációt, ahol az egyes mozgásfázisokat, változásokat csak néhány képkocka számára kell meghatározni és a köztes képeken az egyes tárgyak, fények, kamerák helyzetét a program számolja ki. A PoV-Ray esetében minden képkockán meg kell adni minden tárgy tulajdonságát, és a program ezekből az adatokból állítja elő a képeket. Egy-egy tárgy mozgását függvényekkel írhatjuk le, ezért bemutatom azokat a függvényeket, amelyeket alkalmazhatunk és használatuk módját is igyekszem érthetően elmagyarázni.
A PoV-Ray programozói szerencsére számos függvényt beépítettek a programba, amelyekkel lebegőpontos számokon, vektorokon és szövegállandókon egyaránt műveleteket végezhetünk. Egy-egy függvényt a következő módon hívhatunk meg:

```
f ggvØnynØv(paramØter1,
paramØter2, ..., paramØterN)
```

Ez a hívási mód azonos a C és Pascal nyelvben megszokottakkal. Kezdjük az ismerkedést a lebegőpontos számokkal kapcsolatos függvényekkel, amelyek értékei lebegőpontos számok és eredményül szintén lebegőpontos számot kapunk.

- `abs(A)`: az értékként kapott `A` szám abszolút értékét adja vissza.
- `acos(A)`: Arcus Cosinus, eredménye az a szög radiánban mérve, melynek koszinusza az `A` érték.
- `asin(A)`: Arcus Sinus, eredménye az a radiánban mért szög, amelynek szinusza `A`.
- `atan2(A, B)`: az (`A`/`B`) érték arcus-tangensét kapjuk eredményül. Természetesen `B` értéke nem lehet 0. Egyetlen érték arcus-tangensének kiszámítására is használható a következő formában: `atan2(A,1)`. Erre akkor lehet szükségünk, ha a kiszámítandó szög tangense már ismert. A visszaadott értéket szintén radiánban szolgáltatja.
- `ceil(A)`: meghatározza a legkisebb egész számot, amely nagyobb `A`-nál, vagyis az értékként kapott számot felfelé kerekíti.
- `cos(A)`: a radiánban mért `A` szög koszinuszát adja vissza.
- `degrees(A)`: visszaadja az `A`-nak megfelelő szög fokban mért értékét a `deg=(180/Pi)*A` képlet felhasználásával.
- `exp(A)`: a természetes szám `A` kitevőjű hatványát adja vissza.
- `floor(A)`: az `A` számnál kisebb legközelebbi egész számot határozza meg.
- `int(A)`: a kerekítés matematikai szabálya szerint az `A` számot egészre kerekíti; ha `A` törtrészének értéke nagyobb 0,49-nél, akkor felfelé kerekít, ellenkező esetben lefelé.
- `log(A)`: az `A` szám természetes alapú logaritmusát határozza meg.
- `max(A, B)`: `A` és `B` számok közül a nagyobb értéket adja vissza.
- `min(A, B)`: `A` és `B` számok közül a kisebb értéket kapjuk eredményül.
- `mod(A, B)`: a maradékos osztás által szolgáltatott maradék meghatározása a `mod=A-floor(A/B)` képlet alapján.
- `pow(A, B)`: az `A` szám `B` kitevőjű hatványát határozza meg.



- `radians(A)`: a fokban mért `A` szög radiánértékét kapjuk eredményül.
- `rand(A)`: a következő álvéletlen számot kapjuk vissza – mielőtt használnánk ezt a függvényt, a véletlenszám-generátort a `seed()` függvénnyel be kell töltenünk; a függvény visszatérési értéke 0 és 1 közé esik.
- `seed(A)`: a véletlenszám-generátort alaphelyzetbe állítja – több véletlenszám-generátort is használhatunk, ekkor azonban célszerű különböző kezdőértékeket beállítanunk a számukra; az alábbi példa erre mutat egy lehetőséget:

```
#declare R1=seed(0)
#declare R2=seed(87568)

sphere {
    <rand(R1), rand(R1), rand(R1)>,
    rand(R2)
}
```

Láthatjuk, hogy egy változónak úgy adhatunk kezdőértéket – ezzel egyben a PoV-Rayjel is tudatjuk, hogy az adott azonosítót változóként értelmezze –, hogy a `#declare` kulcsszó után megadjuk a változó nevét és kezdeti értékét.

© Kiskapu Kft. Minden jog fenntartva

1. lista

```
#if (FELT TEL)
            utas tÆsok, ha a feltØtel igaz
      ...
#else
      utas tÆsok, ha a feltØtel hamis
#end
```

2. lista

```
#switch (AZONOSITO)
      #case (FELT TEL_1)
        Utas tÆsok, ha AZONOSITO
        ØrtØke = FELT TEL_1
      #break

      #case (FELT TEL_2)
        Utas tÆsok, ha AZONOSITO
        ØrtØke = FELT TEL_2
      #break

      #range (MIN_1, MAX_1)
        Utas tÆsok, ha MIN_1 <= AZONOSITO
        ØrtØke <= MAX_1
      #break

      #range (MIN_2, MAX_2)
        Utas tÆsok, ha MIN_2 <= AZONOSITO
        ØrtØke <= MAX_2
      #break
      #else
        Utas tÆsok, ha az elızı feltØtelek
        egyike sem teljes lt.
#end
```

- `sin(A)`: az `A` szög szinuszát szolgáltatja – természetesen a szöget itt is radiánban adjuk meg.
- `sqrt(A)`: az `A` szám négyzetgyökét adja vissza.
- `tan(A)`: ahogy már megszokhattuk, az `A` szöget szintén radiánban adjuk meg, e függvény eredménye pedig a szög tangense lesz.

Ezekkel a műveletekkel nagyon sok számítást elvégezhetünk a lebegőpontos számok körében, ezért ideje rátérni a vektorokkal kapcsolatos függvényekre.
Az alábbiakban bemutatott műveletek értékül egy vagy több vektort kapnak, némelyik egy lebegőpontos számot is, eredményük pedig vektor vagy lebegőpontos szám lesz.
Itt vektorként csak háromelemű vektort használhatunk, de a térbeli ábrázolás során valószínűleg ennél többre nincs is szükség. A jelölésekkel kapcsolatban még annyit említenék meg, hogy az `A` és `B` betűk vektort jelölnek, míg az `F` betű lebegőpontos számot.

- `vaxis_rotate(A, B, F)`: az `A`-t elforgatja a `B` vektor által meghatározott tengely körül `F` szöggel; a szöget fokban kell megadni és visszatérési értéke az elforgatott vektor.
- `vcross(A, B)`: `A` és `B` keresztszorzatát számítja ki; a visszaadott vektor merőleges `A`-ra és `B`-re, hosszúsága pedig arányos a közöttük lévő szöggel.
- `vdot(A, B)`: a két vektor skaláris szorzatát határozza meg a következő képlet alapján: `vdot=AxBx+AyBy+AzBz`.
- `vlength(A)`: a vektor hosszúságát adja meg – kiválóan használható két pont távolságának meghatározására, a felhasznált képlet a következő: `vlength= (vdot(A,A))`.
- `vnormalize(A)`: visszatérési értéke egy egységnyi hosszúságú vektor, melynek iránya megegyezik az értékként kapott `A` vektor irányával – a következő képlet használatával állítható elő: `vnormalize=A/vlength(A)`.
- `vrotate(A,B)`: eredménye az az elforgatott `A` vektor, amelyet az `X` tengely körül `Bx` szöggel – az `Y` tengely körül `By` szöggel –, a `Z` tengely körül pedig `Bz` szöggel forgatunk el; az elforgatás mértékét fokban adjuk meg.

A PoV-Rayben használható függvények utolsó csoportját alkotják azok, amelyek segítségével szövegeken végezhetünk műveleteket. Értékük egy vagy több szöveges állandó és esetlegesen lebegőpontos szám, visszatérési értékük pedig szöveg vagy szám lehet. Az alábbiakban `S1` és `S2` jelöli a szöveges értékét, míg az `A`, `L` és `P` jelek lebegőpontos számot vagy ilyen típusú eredményt szolgáltató kifejezést jelölnek. Természetesen egy-egy szám vagy vektor az előbbi függvények esetében is helyettesíthető a szükséges típusú eredményt adó kifejezéssel, függvénnyel.

- `asc(S1)`: visszaadja a szöveg első karakterének ASCII-kódját, például az asc(ABC) értéke 65 lesz, mert az A karakter kódja 65.
- `chr(A)`: azt a karaktert adja visszatérési értékül, melynek ASCII-kódja `A` – a kérdéses számnak 0 és 255 közé kell esnie, például a `chr(70)` értéke `F` lesz, mert ennek a karakternek a kódja 70; amikor ezt a függvényt szövegek megjelenítésére használjuk, figyelembe kell venni azt a tényt, hogy a 127-nél nagyobb kódok által meghatározott karakterek az adott karakterkészlettől függenek; a legtöbb TrueType font az ISO-8859-1-es (Latin-1, nyugat-európai) karakterkészletet használja, melyben a hosszú ő és a hosszú ű nem a magyar írásban használatos ékezetekkel jelenik meg.
- `concat(S1, S2 [,S3, ...])`: a függvény használata során legalább két karakterláncot kell megadnunk, eredményül pedig egy szöveges állandót kapunk, amely az értékekként megadottakat összefűzve tartalmazza; az alábbi példa jól szemlélteti a függvény használatát:

```
#declare cat_str=concat("Alma", "fa")
```

  ennek eredménye az „Almafa” szó lesz.
- `file_exists(S1)`: amennyiben az `S1` által meghatározott állomány létezik, visszatérési értéke 1, máskülönben 0; a keresési útvonal elsősorban a pillanatnyi könyvtár, továbbá a `+L` parancssori kapcsolóval megadott elérési utak.
- `str(A, L, P)`: az `A` lebegőpontos értéket szöveggé alakítja – az `L` határozza meg a szöveg legnagyobb hosszúságát, a `P` pedig a tizedesjegyek számát, ha a szám kisebb helyen is elfér, mint az `L` érték által megadott hosszúság, és a szöveget jobbra igazítja; a `P` értéke negatív szám is lehet, ekkor a függvény a bal oldalon

3. lista

```
//
// Filename: gomb_forog.pov
// Start rendering: povray anim.ini -i
// gomb_forog.pov
//
// anim.ini:
// Initial_Frame=0
// Final_Frame=360
// Initial_Clock=0
// Final_Clock=720
//
#include "colors.inc"
#include "stones.inc"

camera {
   location <3,4,5>
   look_at <0,0,0>
}

light_source {
       <1,5,1>
       1.2*White
}

#declare gomboc=sphere{
            <0,0,0>,1
            pigment {checker
                       pigment {Jade},
                       pigment {Black_Marble}
                    }
            }
object {
   gomboc
   rotate y*clock
}
```

4. lista

```
// `ltalÆnos  zenet
#debug " zenet a DEBUG sorba"
// VØgzetes hiba
#error "A szÆm tÆsi folyamat leÆll"
//  zenet a RENDER sorba
#render "Most szÆmolgatok..."
// Statisztikai adat
#statistics "A kØp elkØsz lt"
// Figyelmeztetøs
#warning "KØrem, ne kapcsolja ki
#a szÆm t gØpet, Øppen szÆmol"
```

keletkező üres karaktereket 0-val tölti fel.

- `strcmp(S1, S2)`: összehasonlítja a két szöveget; eredménye 0, ha az értékek tartalma megegyezik, pozitív szám, ha az ASCII-kódok szerinti rendezés során `S2` megelőzi `S1`-et, és negatív, ha `S1` előzi meg `S2`-t.
- `strlen(S1)`: az `S1`-ben található karakterek számával tér vissza és meghatározza a szöveg hosszúságát
- `strlwr(S1)`: a megadott szöveget kisbetűssé alakítja, tehát minden nagybetűt kisbetűre cserél ki, így az `strlwr(Szia Olvaso)` által visszaadott szöveg „szia olvaso” lesz.
- `substr(S1, P, L)`: az `S1` részét határozza meg a P karaktertől kezdve `L` számú karakter figyelembevételével, tehát a `substr(ALMA,1,2)` hívás a `LM`-részletet adja vissza, mert a PoV-Rayben a szöveg első karaktere a 0. indexet kapja; ha `P+L` nagyobb az `S1` karakterlánc hosszánál, a PoV-Ray hibát jelez.
- `strupr(S1)`: a megadott szöveget nagybetűssé alakítja.
- `val(S1)`: az `S1` szöveget számmá alakítja, vagyis visszatérési értéke lebegőpontos szám.

Ennyi lenne tehát azoknak a függvényeknek a tárháza, amelyeket animációkészítés során is használhatunk, viszont még nem tudjuk, miként lehetne őket a gyakorlatban is felhasználni. Az alábbiakban a vezérlőutasítások ismertetésével ehhez nyújtok segítséget.
Első csoportjuk a feltételek kezeléséhez szükséges úgynevezett feltételes utasítások, ide tartozik az `IF-ELSE`, az `IFDEF` és az `IFNDEF` utasítás. Az `IF-ELSE` utasítás formája a *1. listában* látható.
Itt a feltétel logikai kifejezés, és az ilyen kifejezés akkor igaz, ha az értéke különbözik 0-tól. Amennyiben egyenlő 0-val, a PoV-Ray értelmezése szerint hamis. Az alábbiakban látható példa alapján az utasítás használata talán könnyebben megérthető:

```
#if (file_exists("szentely.inc")
      #include "szentely.inc"
#else
      #declare szentely=sphere{<0,0,0>,1.2}
#end
```

Itt azt adtuk a program tudtára, hogy ha létezik a ***szentely.inc*** állomány, akkor azt használjuk, ha pedig nem létezik, akkor a szentélyt egy gömbbel helyettesítjük. Megjegyezném, a ***szentely.inc*** tartalma célszerűen így kezdődhetne:

```
#define szentely=mesh{...}
```

A következő vezérlőutasítás, amiről szót kell ejtenünk, az `#IFDEF` nevet kapta. Használata akkor javasolt, ha valamilyen változó létezéséhez szeretnénk kötni valamely más utasítás értelmezését. A fenti példához hűen a kép számításának menetét alakítsuk át egy kicsit:

```
#if (file_exists("szentely.inc"))
      #include "szentely.inc"
#end
```

A lista alapján láthatjuk, hogyha létezik a szentélyt tartalmazó állomány, ellátjuk anyaggal, de az előző példával ellentétben nem helyettesítjük gömbbel, tehát amennyiben a külső állományban nem találjuk, nem is fogjuk használni. Az `#IFDEF` utasítás általános formája a következő:

```
#ifdef (szentely)
      object {szentely
            texture {...}
            }
```

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

```
#end
```

Az `#else` utasítást az előző esetek egyikénél sem kötelező használni, de az `#end` mindkét tárgyalt esetben kötelező. Ebbe a csoportba tartozik a harmadik feltételes utasítás is, az `#ifndef`. Használata szinte azonos az `#ifdef` alkalmazásával, a különbség csak annyi, hogy a PoV-Ray a `#ifndef` és az `#end` közötti utasításokat akkor veszi figyelembe, ha a megadott AZONOSITO nem létezik.
Külön soroltam a `#SWITCH CASE` és a `#RANGE` utasításokat, bár ezekkel is feltételes elágazásokat hozhatunk létre, ám általuk nagyobb szabadságot kapunk. Az utasítások általános formája a *2. listán* látható.
A listán alkalmazott módon változatos feltételeket szabhatunk egy esemény bekövetkeztének (például az objektumok létrehozása, elforgatása és eltolása során), amelyet alapvetően az AZONOSITO értéke határoz meg. Amennyiben ez megegyezik valamely `#case`-ágban meghatározott értékkel, az abban az ágban megadott utasítások kerülnek végrehajtásra egészen a következő `#break` vagy `#end` utasításig. Ha a `#case`-ágak egyike sem került végrehatásra, a következő utasítások a `#range` feltételek által meghatározottak lehetnek. Az itt megadottak akkor kerülnek végrehajtásra, ha az AZONOSITO értéke a `#range` utasítás MIN és MAX értéke közé esik. Ez esetben is a végrehajtás a következő `#break` vagy `#end` utasításig folytatódik. Az `#else`-ágban meghatározottakra pedig akkor kerülhet sor, ha az előzők egyike sem teljesült.
A PoV-Rayben ciklusszervező utasítás használatára is lehetőségünk nyílik, azonban egyedül az elölteszlelő ciklust alkalmazhatjuk. A `#WHILE` utasítással több hasonló objektumot hozhatunk létre, és az animációkészítés folyamán a tárgyakon, a kamerákon és a fényforrásokon egyaránt átalakításokat hajthatunk végre. Szintén kifejezésként kell megadnunk őket, amelynek értéke logikai igaz vagy hamis lesz. A cikluson belüli utasítások akkor kerülnek végrehajtásra, ha a kifejezés értéke igaz. Az utasítás általános formája a következő:

```
#while (FELT TEL)
       Utas tÆsok
       ...
#end
```

Szemléltetésképpen készítsünk öt gömböt egymás mellé:

```
#declare Count=0
#while (Count<5)
       sphere {
              <0,0,0>, 1
              translate x*2*Count
              }
       #declare Count=Count+1
#end
```

A vezérlőutasítások ismeretében már csak egy morzsányi ismeret szükséges első animációnk elkészítéséhez. A képkockák kiszámítása során egy állandóan változó értékre van szükségünk, ugyanis ez alapján tudjuk kiszámítani a mozgásokat és az egyéb változó értékeket. A PoV-Rayben ez az érték a `clock` változó, amelynek kezdő és végső értékét, valamint növekményét a számolás megkezdése előtt be lehet állítani. A beállításokat egy kezdő értéket megadó állomány segítségével végezhetjük el, amely rendszerint ***.ini*** kiterjesztésű. Az animáció kiszámításához legalább két értéket meg kell határoznunk: az egyik az `Initial_frame`, a másik a `Final_Frame`. Ezekkel adjuk a PoV-Ray tudtára, hogy az animáció milyen hosszú lesz, a `clock` változó pedig alapértelmezésben 0-tól 1-ig változik, azonban az `Initial_clock`-ot és a `Final_clock`-ot az `.ini` értékeinek segítségével változtathatjuk meg. Az alábbi példánkkal a könnyebb megértést kívánjuk elősegíteni:

```
Initial_Frame=0
Finale_Frame=360
Initial_Clock=0
Final_Clock=720
```

Itt úgy határoztuk meg az animációhoz szükséges értékeket, hogy a teljes animáció 360 képkockából álljon és a `clock` változó értéke 0–720-ig változzon. Ezt felhasználhatjuk például arra, hogy egy gömböt kétszer teljesen körbeforgassunk a *3. listán* látható módon.
Az animáció kiszámítására a következő paraméterezéssel vehetjük rá a PoV-Rayt:

```
povray anim.ini -i gomb_forog.pov
```

vagyis meg kell adni, hogy melyik ***.ini*** állományt szeretnénk használni.
Természetesen ez csak egy egyszerű példa volt az animációra, hiszen a kamerát, a fényforrásokat és minden elképzelhető dolgot mozgathatunk, ezt azonban sorozatunk záró részében fogom bemutatni.
Ebben a részben csupán egyetlen kiegészítő adatot szeretnék még olvasóinkkal megosztani, ez a felhasználói üzenetek létrehozására vonatkozik. A PoV-Ray a különböző típusú üzenetek megjelenítésére különböző üzenetfolyamokat használ. Ezek mindegyike átirányítható vagy akár le is tiltható. Az alább felsoroltakat különböztetjük meg:

- `DEBUG`-üzeneteket, amelyek általában a fejlesztők számára hasznosak, bár a felhasználók is hasznosíthatják.
- A `FATAL`-üzenetsor olyan üzenetek megjelenítésére alkalmas, amelyek a PoV-Ray leállásához vezetnek.
- A `RENDER`-üzenetek arra használhatók, hogy a számítások során általános adatokat jelenítsünk meg.
- A `STATISTICS`-üzenetsor pedig minden képkocka kiszámítása után a számítások során készített kimutatásokat tartalmazza.
- A `WARNING`-üzenetek olyan hibákat jeleznek, amelyek bekövetkeztekor a számítási folyamat nem áll le, és ez az üzenet jelenik meg figyelmeztetésként.

A felsorolt öt üzenetsorba a felhasználók is írhatnak üzeneteket a *4. listán* látható utasítások segítségével.
Mindenkinek kellemes alkotást kívánok!

Fábián Zoltán
(dzooli@freemail.hu, dzooli@yahoo.com)
24 éves, jelenleg programozóként dolgozik. Szabadidejében szívesen kirándul, túrázik. Emellett szeret rajzolni, érdekli a 3D grafika és a Linuxszal kapcsolatban minden olyan program és programnyelv, amit még nem ismer vagy nem próbált ki.

© Kiskapu Kft. Minden jog fenntartva

A LaTeX2HTML

Sok matematikai képletet, jelölést alkalmazó dokumentumok átalakítása az Interneten is használható formába a LaTeX2HTML segítségével.

Különös, de a matematikai dokumentumok webes közzététele még a Világháló huszadik születésnapja körül sem egyszerű. Számos új szabvány jelent már meg, például a MathML, azonban addig, amíg ezeket a népszerű böngészőprogramok nem támogatják, a képleteket úgy tehetjük elérhetővé a lehető legszélesebb közönség számára, ha formázatlan (in-line) szövegként HTML-fájlokba illesztjük. Az Interneten kívüli világban a Tex/LaTeX csomagot használják a legjobb minőségű tudományos szövegek létrehozására. Olyannyira, hogy a LaTeX nagyon sok egyetemi, főiskolai szaklap hivatalos formátuma. A kutatók és tudósok számára e két világ legjobb lehetőségeinek ötvözése jelenti a megoldást: az anyagokat LaTeXben írják, majd HTML-formában teszik közzé a Világhálón, ugyanis erre a LaTeX2HTML a legmegfelelőbb eszköz.

A LaTeX2HTML-t eredetileg ***Nikos Drakos*** írta 1993-ban, és fejlesztése nyílt forrású volt. A programot a Nyílt Forrás közösségének számos tagja fejlesztette tovább. A LaTeX2HTML LaTeX-dokumentumokat alakít szabványos HTML formátumra. Mivel a HTML egy szövegformátum, így a program minden nem szöveges elemet beágyazott grafikává alakít át. A LaTeX2HTML egy Perl-alkalmazás, sokféle felületen, így a Unix, illetve a Linux legtöbb változatán és Windows alatt egyaránt futtatható.

Írásomban először egy példán keresztül szemléltetem, miként alakíthatunk egy egyszerű LaTeX-dokumentumot HTML-formátumba és azt is elmesélem, hogy a program hogyan kezeli a beágyazott grafikákat és a stílusfájlokat. Ezután a LaTeX2HTML-re jellemző LaTeX-parancsokat és -környezeteket ismertetem. Végül a LaTeX2HTML használatának egy ötletesebb módját is bemutatom, melyben a kiszolgálóoldali felhasználásról lesz szó. Az egyszerűség kedvéért egyezzünk meg abban, hogy a „képlet” kifejezés mindenre vonatkozik, ami a LaTeX matematikai üzemmódjában szerepel, így tehát a szövegben szereplő matematikai jelekre, a „displaymath”-képletekre és a számozott egyenletekre is.

Egy egyszerű példa

A parancs használata nagyon egyszerű. Amennyiben a ***mydoc.tex*** fájlunkat kívánjuk webes formára alakítani, akkor a

```
latex2html -local_icons mydoc.tex
```

parancsra van szükségünk. A LaTeX2HTML egy új könyvtárat (***mydoc***) hoz létre és az összes HTML- és képfájlt ide helyezi. Ezután a ***mydoc*** könyvtár tartalmát webkiszolgálónk saját könyvtárába másolhatjuk, s a LaTeX-formátumból átalakított anyagok a ➲ http://kiszolgálónév/útvonal/mydoc/index.html címen lesznek elérhetők.

A címet, a címsorokat és a kiemelt szövegrészeket megfelelő HTML-elemek jelölik. A program minden olyan képletet, táblázatot és ábrát, amelyet a HTML nem képes megjeleníteni, grafikává alakít, és ezeket külön képfájlokban tárolja.

Ha a ***mydoc.tex*** fájl jó néhány szakaszból, illetve alfejezetből áll, a LaTeX2HTML minden egyes részhez külön HTML-fájlt, majd egy tartalomjegyzéket is készít, amelyben az egyes részekre mutató hivatkozásokat is elhelyezi. Minden oldal rendelkezik egy navigációs sávval, mellyel az előző (***prev***) és a következő (***next***) szakaszra vagy a tartalomjegyzékre ugorhatunk. Ha csak egyetlen nagy HTML-fájlt szeretnénk létrehozni, használjuk a `-split 0` parancssori kapcsolót.

Ötlet

Kereszthivatkozások

A kereszthivatkozások adatai (ábrák és táblázatok sorszáma, idézetek stb.) egy *.aux* fájlban tárolódnak, melyet a LaTeX állít elő. A LaTeX2HTML-nek ezt be kell olvasnia, hogy létrehozhassa a megfelelő kereszthivatkozásokat. Tehát a kereszthivatkozásokat tartalmazó dokumentumokat először a LaTeX segítségével is fel kell dolgoznunk.

A fájlnevek kiterjesztése

A LaTeX2HTML alapértelmezés szerint *.html* kiterjesztésű HTML-fájlokat készít. Ez bizonyos operációs rendszerek és programok esetében gondot okozhat. A `-short_extn` kapcsoló használatával arra utasíthatjuk a programot, hogy a *.htm* kiterjesztést alkalmazza.

A LaTeX2HTML-nek tudnia kell, hogy a navigációs sáv ikonjait hol találja. A legegyszerűbb megoldás a `-local_icons` kapcsoló használata, mely arra utasítja a LaTeX2HTML-t, hogy a navigációs ikonokat a ***mydoc*** könyvtárba másolja. Saját navigációs ikonokat is használhatunk, ha lecseréljük a ***mydoc/*motif*.gif*** fájlokat.

Beágyazott grafikai elemek

A következő példában matematikai képleteket használok, hogy bemutassam, a LaTeX2HTML miként kezeli a beágyazott grafikai elemeket. Megjegyzem, hogy a program ugyanígy bánik a lebegő tárgyakkal, például az ábrákkal és táblázatokkal is.

A LaTeX2HTML végigolvassa a ***mydoc.tex*** fájlt és az összes matematikai képletet az ***images.tex*** fájlba másolja, ahol minden képlet külön oldalon szerepel. Ezt követően meghívja a `latex images.tex` és a `dvips -S 1 -i` parancsokat, melyek egyoldalas PostScript fájlt készítenek minden képletből. A PostScript fájlokat ezután a GhostScript alakítja gif- vagy png-formátumú képekre. A LaTeX2HTML emlékezni fog a képek neveire és a ***mydoc.tex*** fájlból létrehozott HTML-fájlokba visszahelyezi a képekre mutató hivatkozásokat.

A képalkotási folyamatot számos parancssori kapcsolóval befolyásolhatjuk. Például a `ps_images` kapcsoló hatására

© Kiskapu Kft. Minden jog fenntartva

A LaTeX-karakterláncból HTML-elemeket létrehozó Perl-függvény

```
sub LaTeX2HTML {
   # Az elkØsz lt kØpfÆjlok helye
   my $ImgPath =
   ➥ /usr/apache/htdocs/l2h/images ;
   # URL -> kØp k nyvtÆr
   my $ImgURL =
   ➥ http://kiszolgalo/l2h/images ;
   # A latex2html futtathat  fÆjlja
   my $L2H =  /opt/local/bin/latex2html ;

   my $latexString = shift;
   # A fÆjlneveket a rendszeridıbıl Øs a
   # folyamat-azonos t b l (PID) Ælltja elı
   my $filename =  image$^T$$ ;

   open (LATEXFILE,  >$filename.tex )
            || die  Nem lehet megnyitni a
                  ➥fajlt ;
   print LATEXFILE  \\documentclass[12pt]
             {article} \n
            \\begin{document} \n
            \\begin{equation} \n
            $latexString \n
            \\end{equation} \n
            \\end{document} ;
   close (LATEXFILE);

   system ( $L2H $filename.tex >
            ➥/dev/null );
   rename  $filename/img1.gif ,
         ➥ $ImgPath/$filename.gif ;
   system ( rm -rf $filename.tex $filename );

   my $HTMLstr =  <IMG
➥SRC=\ $ImgURL/$filename.gif\ > ;

   return $HTMLstr;
}
```

a LaTeX2HTML a külső PostScript fájlokra hivatkozik és nem a grafikus fájlokra.

Stílusfájlok

A LaTeX2HTML a szövegek és a grafikák stílusfájljait különböző módon kezeli. Amikor a program a fő HTML-fájlokat hozza létre a ***mydoc.tex*** fájlból, akkor minden stílusfájlt figyelmen kívül hagy. Azonban a stílusfájlok gyakran olyan új parancsokat és környezeteket határoznak meg, melyeket nem lehet figyelmen kívül hagyni. Áthidaló megoldásként a LaTeX2HTML lehetővé teszi, hogy a stílusfájlt Perl-formátumban újraalkossuk. Amikor a LaTeX2HTML egy stílusfájllal találkozik, megkeresi annak Perl-fordítását és ezt dolgozza bele a fő parancsfájlba. Szerencsére számos népszerű stílusfájlt már lefordítottak Perlre. Egy ilyen fordítás elkészítéséhez tökéletesen tisztában kell lennünk a LaTeX2HTML belső felépítésével, működésével, azonban ne keseredjünk el: a forráskód mindenki számára hozzáférhető.
Azt szeretnénk, hogy a képletek, ábrák és táblázatok ugyanúgy nézzenek ki a böngészőben, mint nyomtatásban, és hogy minden jel, betűtípus és szövegköz megfelelő legyen. Mivel az ***images.tex*** fájlt a LaTeX dolgozza föl, így a stílusfájlokat a program már megfelelően kezeli.

A webes megjelenés

A LaTeX2HTML-t jó eredménnyel használhatjuk dokumentumaink webes formátumra történő átalakítására. A LaTeX2HTML saját LaTeX-parancsai és -környezetei segítségével saját készítésű HTML-elemeket is elhelyezhetünk a szövegben. A hagyományos HTML minden lehetőségét kihasználhatjuk: űrlapokat, kattintható képtérképeket (image map), külső hivatkozásokat és grafikákat, de akár Java kisalkalmazásokat, parancsfájlokat is alkalmazhatunk a dokumentumban. Ha például a létrehozott HTML-oldalainkba egy külső weblapra mutató hivatkozást kívánunk elhelyezni, akkor csak az alábbi kódot kell beillesztenünk a ***mydoc.tex*** fájlba:

```
\htmladdnormallink{hivatkozÆsnØv}{URL}
```

vagy általánosabban:

```
\begin{rawhtml}
<A HREF= URL >hivatkozÆsnØv</a>
\end{rawhtml}
```

Ha ezen HTML-elemekkel bővített dokumentum nyomtatott változatát szeretnénk papíron megtekinteni, akkor fűzzük be a ***html.sty*** stílusfájlt, majd a dokumentumot a LaTeX segítségével dolgozzuk fel. A LaTeX2HTML saját parancsainak és környezeteinek legtöbbjét a LaTeX figyelmen kívül hagyja.

A kiszolgálóoldal

A LaTeX2HTML a kiszolgálóoldalon is felhasználható, a webalapú matematikai kapcsolattartás megvalósítására. *Listánk* egy Perl-függvényt mutat be, mely egy matematikai üzemmódú LaTeX-karakterláncot olvas be, kimenetként pedig a képletet tartalmazó képet megjelenítő HTML-kódot adja. Létezik ennél hatékonyabb módszer is, de ez csupán egy példa a LaTeX2HTML képességeinek bemutatására. A programot felhasználhatjuk például webes csevegőszobákban, fórumokon, vendégkönyvekben, így a felhasználók egyszerűen tehetnek közzé összetett képleteket.

Michael Yuan
asztrofizikus doktorandusz képzésben vesz részt a Texasi Egyetemen, Austinban. Távoli (húszmilliárd fényévnél messzebbre lévő) kvazárokat tanulmányoz, így próbálja megérteni a Világegyetem történetét és fejlődését. Amikor épp nem kvazárok után kutat, akkor földi nyelvekkel foglalkozik, például Javával és Perllel.

Kapcsolódó címek

Robert Kiesling „Ghostscript" című írása (Linux Journal 47. szám, 1998. március, ➲ http://www.linuxjournal.com/lj-issues/issue47/2328.html)
A LaTeX honlapja ➲ http://www.latex-project.org
A LaTeX2HTML honlapja (letöltések, kézikönyvek, fejlesztések) ➲ http://www.latex2html.org

Enterprise JavaBeans

Reuven egy olyan környezetet mutat be nekünk, amelynek segítségével Jboss Java alkalmazáskiszolgálót használó megosztott alkalmazásokat készíthetünk.

Ahogy a webes alkalmazások kezdenek egyre komolyabbá válni, a fejlesztők is egyre igényesebbek lesznek az eszközválasztás terén. Az előzőek folyamán (Linuxvilág 11. szám 74. oldal és 12. szám 74. oldal) két objektumrelációs átalakító eszközt (az Alzabo és a DODS) rejtelmeibe is belepillantottunk. Ezek lehetővé tették, hogy az adatbázisokat objektumokon keresztül kezeljük, s ezáltal elkerüljük az SQL-utasítások kódba illesztését.

Számos olyan helyzet akad azonban, amikor ezek a módszerek sajnos nem alkalmazhatók eredményesen: hogyan lehet például az objektumokat különböző számítógépekre szétválasztani?

Amennyiben valahogy mégis sikerül szétválasztani őket, miként találják meg egymást az objektumok? És ha az objektum adott állapota az adatbázis egy vagy több sorának felel meg, akkor hogyan kezeljük a tranzakciókat?

A fent felsoroltak igen fogós és bonyolult kérdések, olyannyira, hogy akár évekig is elbirkózhatnánk megoldásukkal. Ezekre (és számos más) kérdésre az egyik legátfogóbb megoldást a J2EE (Java 2 Enterprise Edition) felület és az Enterprise JavaBeans objektummodell kínálja. Az EJB-t, ahogy gyakran emlegetik, arra tervezték, hogy összetett, nagyméretű weboldalak készítéséhez nyújtson hathatós segítséget, hiszen nagymértékben képes enyhíteni az infrastruktúrával a programozói háttérrel kapcsolatos gondokat.

E hónapban a JBoss alkalmazáskiszolgáló által megvalósított EJB-változat rejtelmeibe kukkantunk bele. A JBoss a GNU Lesser General Public License (LGPL) felhasználási szerződés hatályába tartozik, nem használ különösebben sok memóriát, és viszonylag egyszerű használni.

Amint azt a Java programozási feladatoknál már megszokhattuk, az EJB-vel való munkához ismét meg kell tanulnunk egy új eszközkészletet kezelni, be kell állítanunk néhány XML fájlt, és természetesen vigyáznunk kell, hogy a fordítás és a futtatás alatt a `CLASSPATH` változó a megfelelő értékeket tartalmazza.

Ha viszont végül ezeken a szerkesztési buktatókon sikerül keresztülverekednünk magunkat, a JBoss kitűnő alapot nyújt a kiszolgálóoldali módszerekkel való hatékony munkához.

Java Enhydra és egyéb gondok

Mielőtt továbblépnénk, fontos, hogy nyomatékosítsuk, az EJB és a Java nyelv nem teljes mértékben szabad programok. Igaz ugyan, hogy nem kell fizetnünk a Java vagy a J2EE könyvtárak letöltéséért, de a Sun birtokol mindent, aminek akár csak a legcsekélyebb köze van a Javához, beleértve a leírásokat és szabványokat is. A Sun Community Source License ugyan nyitottabb, mint sok más felhasználási szerződés, azonban igen messze áll a nyílt forráskódtól.

Ez különösen nyilvánvalóvá vált most, hogy a Lutris Corporation, amely a nyílt forrású Enhydra alkalmazáskiszolgálót támogatta, befagyasztotta a J2EE-minősítésű Enhydra Enterprise kiszolgálójának fejlesztését. A Lutris zárt forrású fejlesztéssé változtatta az Enhydra Enterprise-t, arra hivatkozva, hogy a Sun felhasználási szerződése nem teszi lehetővé, hogy teljes mértékben együttműködő, nyílt forrású J2EE-kiszolgálót hozzanak létre.

Ez természetesen hatalmas felháborodást (és védekezést) váltott ki a legfőbb Enhydra levelezőlistán, és számos megválaszolatlan kérdés is nyitva maradt. Lehet, hogy a Lutrisnak jogilag (és pénzügyileg) valóban muszáj volt megtennie, amit megtettek, azonban az a viselkedésmód ahogyan tették, ékes példája annak, hogyan nem szabad véget vetni egy nyílt forrású fejlesztésnek.

Szerencsére, a JBoss csapat megerősítette, hogy a JBoss továbbra is nyílt forrású marad, tovább növekedik, és támogatni fogja az összes J2EE-szabványt, akkor is, ha az nem rendelkezik a megfelelő hivatalos bizonyítvánnyal, ennek oka pedig általában az, hogy nincs elegendő pénz a Sun bizonyítványának megszerzéséhez.

Mi az az EJB?

A helyes első kérdés ez lenne: „Miért van szükségem az EJB-re?”. Való igaz, sok olyan alkalmazás létezik, amelyhez EJB-t használni merő túlzás lenne. Ugyanakkor az EJB olyan képességeket tesz elérhetővé számunkra, amelyeket magunktól csak nagy erőfeszítések árán fejleszthetnénk ki a kiszolgáló, más néven a tároló (container) belsejében:

- Az EJB lehet az alkalmazással azonos számítógépen, de egy távoli gépen is. Így akár több részből álló alkalmazásokat is készíthetünk, ahol minden egyes rész egy-egy külön gépen fut, miközben a programunk változatlanul működik akkor is, ha egyik részről a másikra másoljuk, vagy megváltoztatjuk a gépek beállításait.
- Az EJB-tároló képes kezelni az objektumrelációs-átalakítás nehézségeit. Csak meg kell határoznunk a táblát és az azt kezelő objektumot, minden további feladatot a tároló végez helyettünk. Avagy, ha jobban szeretjük finomhangolni a dolgokat, lehetőségünk nyílik arra is, hogy babunk saját rétegét módosítsa.
- A relációs adatbázis-kezelők tranzakciókat is képesek használni, ez lehetőséget ad arra, hogy két vagy több műveletet egyetlen műveletként kezeljünk. Az EJB ehhez hasonló tranzakciószerű képességekkel ruházza fel objektumunkat, lehetővé téve, hogy az eljárásunk több műveletet hajtson végre egyetlen „mindent vagy semmit” blokkban.

Fontos tisztában lennünk azzal is, mit nem tud nyújtani az EJB. A hasonló elnevezés ellenére, az Enterprise JavaBeansnek szinte semmi köze nincs a hagyományos

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

JavaBeanshez. A JavaBeans egységes felülettel rendelkezik, ami lehetővé teszi, hogy JSP-ből egészen kevés kód felhasználásával vagy akár kód nélkül is elérhessük. Az EJB-t ezzel szemben úgy tervezték, hogy bármilyen Java programból használható legyen, ideértve a servleteket is. Továbbá, az EJB-hez tartozó szabványos felület (API) gazdagabb, összetettebb és rugalmasabb, mint a JavaBeanshez tartozó. Sajnálatos módon a JavaBeans kifejezést meglehetősen elkoptatta ez a két népszerű kiszolgálóoldali módszer, azonban úgy tűnik, ez ellen már nem tehetünk semmit.
Az EJB egyik leglenyűgözőbb tulajdonsága, hogy a hozzátartozó API szabványos az alkalmazáskiszolgálókon. Így aztán a fejlesztést elkezdhetjük egy olyan nyílt forrású EJB-kiszolgálón, mint a JBoss, majd, amikor elérkezettnek látjuk az idejét, telepíthetjük egy kereskedelmi kiszolgálóra. (Természetesen az is előfordulhat, hogy amikor megtudjuk, mennyibe is kerülne egy kereskedelmi kiszolgáló, mégis inkább megmaradunk a JBoss mellett.)
A Java-fejlesztésekben talán az a legbosszantóbb dolog, hogy számtalan akronímet, projektnevet és változatszámot kell fejben tartanunk. Ez a cikk például a JDK (Java Development Kit) 1.3 és az EJB 1.1 szabványnak megfelelő Jboss-kiszolgáló 2.4.1a változatára épít. Ráadásul, míg magukat az EJB-osztályokat nem különösebben nehéz létrehozni, a fordítással és üzembe helyezéssel járó feladatok kifejezetten bosszantóak és nehézkesek lehetnek az avatatlanok számára.

Munka az EJB-kel

Az alkalmazásunk EJB alá helyezése azt jelenti, hogy az üzleti logika akkora részét tesszük külön objektumokba, amennyit csak tudunk. EJB alatt az objektumok két fajtáját különböztetjük meg:

- Az entitás babok olyan objektumok, amelyek a relációs adatbázisokat alakítják át. Az entitás babok minden egyes példánya általában az adatbázis egy-egy sorának felel meg. A példányváltozók pedig az adattábla oszlopainak feleltethetőek meg. Az objektumunkhoz tartozó adattáblát hagyományos módon kell létrehoznunk az adatbázisban, azonban az EJB-tároló az összes `SELECT`, `INSERT`, `UPDATE`, és `DELETE` lekérdezést már elvégzi helyettünk.
- A session babok műveleteket hajtanak végre, akár saját maguk akár egy vagy több entitás babon keresztül. A session babok szokott esetben nem rendelkeznek saját állapottal, ez pedig hatékonyabbá teszi őket az entitás babok-nál. Akadnak azonban olyan esetek, amikor jól jön, ha mégiscsak létezik állandó elem a session babban. Ezért aztán az EJB lehetőséget nyújt állandósított session babok kialakítására is, amelyek állapota megőrződik az egyes hívások között.

Ha egy hálózati csevegőszobát szeretnénk készíteni EJB-vel, valószínűleg el kellene készítenünk pár entitás babot (és a nekik megfelelő táblákat) a felhasználók, a hírcsoportok és a küldemények számára. Szükségünk lenne természetesen pár session babra is, amelyek a hozzáadást, módosítást és törlést valósítják meg az imént felsorolt entitásbab-típusokra, illetve egész hírcsoportokat vagy egyedi leveleket kérdeznek le.

A JBoss telepítése

A JBoss Java alkalmazáskiszolgáló, amely lehetővé teszi, hogy többszörözött J2EE-alkalmazásokat használjunk. A JBoss nem is próbálja meg kezelni az alkalmazásoldali eseményeket; ha ilyesmire van szükségünk, akkor a Jakarta-Tomcat vagy más servlettárolóhoz kell fordulnunk. Elérhetővé tesz viszont olyan háttérszolgáltatásokat, mint a könyvtárszolgáltatás, az üzenetváltó szolgáltatás illetve maga az EJB-tároló.
Feltételezve, hogy már korábban telepítettük a JDK-t, a JBoss telepítése meglepően könnyű. A Sun rpm-formátumban is elérhetővé tett egy JDK-másolatot, amit a
➲ http://www.java.sun.com honlapról tölthetünk le. Ezenkívül még az Ant eszközt is le kell töltenünk és telepítenünk. Ez a Java program a vénséges-vén `Unix make` programot hivatott helyettesíteni. Ha tájékozottak vagyunk az XML-formátumban és ismerjük a `make` programot, akkor az 1. listában látható (26 CD Magazin/Javabeans), az Ant által használt ***build.xml*** fájl formátumát is viszonylag egyértelműnek fogjuk találni.
A JDK és az Ant telepítése után a JBoss feltelepítése már gyerekjáték. Letöltöttem a bináris kódot a
➲ http://www.jboss.org honlapról, ahol az összevont Jboss- és Jakarta-Tomcat-támogatást állítottam be. A fájl zipállományként töltődik le, ez azt jelenti, hogy szükségünk lesz az Info-zip-alkalmazásokra is (amelyek az általam használt Linux-terjesztésben alapértelmezés szerint benne foglaltattak), hogy kicsomagolhassuk őket.
Kicsomagolás után a Jboss-Jakarta-terjesztés két alkönyvtárat tartalmaz, ***jboss*** és ***tomcat*** néven. Állítsuk be a `JBOSS_DIST` környezeti változót úgy, hogy a ***jboss*** könyvárra mutasson. Így a különféle Jboss-szal kapcsolatos eszközök és lehetőségek képesek lesznek megtalálni a megfelelő fájlokat.
Ha idáig eljutottunk, akár el is indíthatjuk a Jboss-kiszolgálót a következő két paranccsal:

```
cd $JBOSS_DIST/bin
sh run.sh
```

Alapértelmezés szerint a JBoss elég sok adatot naplóz a terminálablakban.

Egy számológép bab megírása

Az első EJB-nk a ***Calculator*** (számológép) lesz. Ez egy állandó tag nélküli session bab, melynek `multiply()` tagfüggfénye két egész számot vár, és a szorzatukat adja vissza.
Miután megírtuk a Calculatort és a hozzá tartozó EJB-felületeket, megvizsgáljuk, hogyan tudjuk használni egy önálló Java programból.
Egy szorzóeljárással felruházott egyszerű számológép-osztály elkészítése alapesetben nem lenne túl nagy feladat. Egyszerűen létre kellene hoznunk a ***calculator.java*** fájlt, és meg kell határozni benne a tagfüggvényt a következő módon:

```
public int multiply (int num1, int num2)
```

Az EJB lehetővé teszi, hogy ***Calculator*** babunkat távolról is megkereshessük és meghívhassuk, ez egyben azt is jelenti, hogy meg kell írnunk azokat a tagfüggvényeket, amelyek segítségével meg lehet találni a ***Calculator***t. Végső soron az alkalmazásunk egy távoli hivatkozással fog dolgozni, ahelyett hogy a valódi objektumot érné el. A session babok írása tehát egy Java-osztály és két csatolófelület létrehozását jelenti.
A Java-osztály az a babosztály, amely a tényleges munkát végzi. A babosztály nem tud róla, hogy egy másik gépen lévő objektumról hívták meg; bár lekérdezheti a környezetét, azonban szokásos esetben ez nemigen szükséges. A babosztályához általában az EJB egyszerűsített nevét használják,

3. lista ejb-jar.xml, a Calculator bab telepítés-leírója

```
<?xml version="1.0" encoding="UTF-8"?>

<ejb-jar>
  <description>ATF Calculator Session
  Bean</description>
    <display-name>Calculator Session Bean
      ➥/display-name>
  <enterprise-beans>
    <session>
       <ejb-name>Calculator</ejb-name>

<home>il.co.lerner.calculator.CalculatorHome
➥</home>

<remote>il.co.lerner.calculator.Calculator
➥</remote>
       <ejb-class>il.co.lerner.calculator.
➥CalculatorBean</ejb-class>
       <session-type>Stateless
➥</session-type>
       <transaction-type>Bean
➥</transaction-type>
    </session>
  </enterprise-beans>
</ejb-jar>
```

kiegészítve a „bean” szócskával. A ***Calculator*** EJB-nk babosztálya így a ***CalculatorBean*** nevet kapja és a ***CalculatorBean.java*** fájlban határozzuk meg. A babosztálynak, típusának megfelelően, létre kell hoznia egy ***SessionBean*** vagy egy ***EntityBean*** csatolófelületet.
Az első csatolófelület a távoli csatolófelület (remote interface), amely lehetővé teszi, hogy az alkalmazás megtalálja ***Calculator*** EJB-t, illetve hivatkozást kérjen hozzá. A távoli csatolófelületnek hagyományosan egyszerű nevet adunk, mint például ***Calculator***, és ennek megfelelően a ***Calculator.java*** fájlba helyezzük. A távoli csatolófelület a babosztály minden egyes nyilvános eljárásához meg kell határoznia egy tagfüggvényt. A távoli csatolófelületnek az `EJBObject`-osztályt kell kibővítenie.
A második csatolófelület, a saját csatolófelület (home interface), amely lehetővé teszi az EJB-tároló számára, hogy létrehozza, beazonosítsa és törölje, illetve más módokon kezelje az Enterprise JavaBeant. A saját csatolófelületnek hagyományosan ugyanazt a nevet adjuk, mint a távoli csatolófelületeknek, kibővítve a ***Home*** szóval. Az EJB-nk saját csatolófelületének neve tehát ***CalculatorHome*** lesz, és a ***CalculatorHome.java*** fájlban helyezkedik majd el. A saját csatolófelületnek az `EJBHome`-osztályt kell kiterjesztenie.
Az egyik igen hasznos dolog az EJB-ben, hogy a létrehozott osztályaink a legtöbb esetben az alapértelmezett EJB-megoldásokra hagyatkozhatnak. Meglehet, nem ez a leghatékonyabb módja a dolgok kezelésének, de ezáltal a kód funkcionális részeinek megírására összpontosíthatunk az EJB-tárolóra hagyva a háttér kezelésének döntő részét.

Az osztályok megírása

Most, hogy már megértettük milyen osztályokat kell létrehoznunk, végre elkezdhetünk magával a kóddal foglalkozni.

Hamar észre fogjuk venni, hogy valójában nem is kell sok kódot írnunk. A ***Calculator*** bab esetében például, sok tagfüggvényt csak üres programtestek határoznak meg.
Ez azért van így, mert a ***SessionBean*** csatolófelület, amelytől a ***CalculatorBean*** örököl, rákényszerít bennünket arra, hogy akkor is megadjuk ezeket a tagfüggvényeket, ha a babunk olyan egyszerű, hogy nem is használja őket. Az üres testek használatával teljesítjük a csatolófelület elvárásait, ugyanakkor a kód is egyszerű marad.
Az összes Java fájlt az ***il.co.lerner.calculator*** csomagba helyeztem, bizonyítván, hogy az én üzleti tartományomból érkeztek, illetve hogy ez a ***Calculator*** nevű projekt. Ennek megfelelően az összes ***.java*** forrásfájlt a ***/il/co/lerner/calculator*** könyvtárszerkezetben helyeztem el.
A babosztályunk a ***CalculatorBean*** (lásd a 2. listát 26. CD Magazin/Javabeans), egyetlen `multiply` tagfüggvényt határoz meg, amely két egész szám bemenetet vár, és egy egész számot térít vissza a hívóhoz. A munkamenet (session) csatolófelület-megvalósításán kívül a ***CalculatorBean***-nek nem túl sok köze van az EJB-hez. Tulajdonképpen ez egy meglehetősen unalmas osztály egyetlen tagfüggvénnyel. Minden, amit a ***System.out***-ra írunk az a JBoss naplójába fog kerülni.
A saját csatolófelületünk, a ***CalculatorHome*** segítségével létrehozhatunk egy új ***CalculatorBean*** példányt. A saját csatolófelület megadja a csatolófelület leírását, ideértve, hogy a visszatérési értéke a távoli csatolófelület (Calculator) egy példánya. Ezenkívül már csak nyúlfarknyi kódot tartalmaz:

```
package il.co.lerner.calculator;

import java.io.serializable;
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface CalculatorHome extends EJBHome
{
   Calculator create() throws RemoteException,
   ➥CreateException;
}
```

Végül a ***Calculator*** nevű távoli csatolófelületünk a ***CalculatorBean*** minden nyilvános eljárásához megad egy-egy tagfüggvény-meghatározást:

```
package il.co.lerner.calculator;

import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface Calculator extends EJBObject
{
   public int multiply(int num1, int num2 )
➥throws RemoteException;

}
```

Az ügyfélprogram nem közvetlenül a babban, hanem a távoli csatolófelületen keresztül fogja meghívni a tagfüggvényeket. A távoli csatolófelület és a babosztály megadásának meg kell egyeznie, különben komoly nehézségekkel kell szembenéznünk a későbbiek folyamán.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

4. lista UseCalculator.java, amely összekapcsolódik a Calculator EJB-vel és felhasználja azt

```
package il.co.lerner.calculator;


import javax.naming.initialContext;
import javax.rmi.portableRemoteObject;

import il.co.lerner.calculator.calculator;
import
➥il.co.lerner.calculator.calculatorHome;


class UseCalculator
{
  public static void main(String[] args)
  {
    try
       {
          //ElnevezØs context megszerzØse
          InitialContext jindiContext = new
            ➥InitialContext ();

          //Calculator Bean hivatkozÆs
          // megszerzØse
          object ref = jindiContext.lookup
      ➥("calculator/calculator");
          System.out.println("Got reference");

          //HivatkozÆs megszerzØse a
          //bab sajÆt csatol fel letØhez
          CalculatorHome home =
      ➥(CalculatorHome)
             PortableRemoteObject.narrow
      ➥(ref, CalculatorHome,class);

          //Calculator objektum kØsz tØse a
          // a sajÆt csatol fel letbıl
            Calculator Calculator =
      ➥home.create()

          //multiply() megh vÆsa
          System.out.println("Multiplying
➥2 x 3:");

System.out.println(Calculator.multilpy(2, 3));
       }
      catch(Exception e)
          {

System.out.println(e.toString());
          }
  }
}
```

A bab telepítése

Most, hogy elkészítettük, itt az ideje, hogy a ***Calculator*** nevű session babunkat elültessük a Jboss-kiszolgálóba. A session babunk elültetése annyit jelent, hogy valamennyi összetevőt egyetlen Java archív állománnyá (jar) alakítjuk. A ***.jar*** állománynak a lefordított `Calculator`-, `CalculatorHome`- és `CalculatorBean`-osztályokat fogja tartalmazni.
Található benne továbbá egy ***ejb.jar.xml*** nevű XML fájl, az úgynevezett „telepítés-leíró" (deployment descriptor), amely az EJB-tároló számára írja le a ***.jar*** fájl tartalmát. A telepítés-leíró kötelező része az EJB-szabványnak és minden alkalmazáskiszolgálón azonosan működik. Feladata az, hogy közölje az EJB-tárolóval az általunk kiválasztott osztályok és csatolófelületek neveit, illetve magunk is meghatározhatunk itt bizonyos elemeket, például a babunk által támogatott tranzakciók típusait.
A ***Calculator*** EJB-nkhez tartozó telepítés-leírót a *3. listában* találjuk. Ezt is ugyanoda kell helyeznünk, ahová a többi ***.java*** forrásfájt tettük.
A ***.jar*** fájlunk tartalmaz egy ***jboss.xml*** nevű rövidke XML fájlt is, amit az ***ejb-jar.xml*** mellé fogunk helyezni:

```
<?xml version="1.0" encoding="UTF-8"?>
<jboss>
    <enterprise-beans>
       <session>
             <ejb-name>Calculator</ejb-name>
             <jndi-
name>calculator/Calculator</jndi-name>
       </session>
    </enterprise-beans>
</jboss>
```

A ***jboss.xml*** fájl egyedi Jboss-jellegzetesség, amely a babunkat a Java elnevezés és könyvtár felületéhez (Java Naming and Directory Interface azaz JNDI) kapcsolja. Ha a ***jboss.xml*** a helyén van, bármely ügyfélprogram, amely a JNDI-n keresztül a ***calculator/Calculator*** után érdeklődik, visszakaphatja a vonatkozó hivatkozást.
Kézzel is felépíthettük volna a ***.jar*** fájlt, azonban miért ne használhatnánk fel az Antot a fájl létrehozására és megfelelő helyre mozgatására. Az 1. lista egy Ant ***build.xml***-t tartalmaz, amely elvégzi a telepítést és kezeli az (alapértelmezett) ***ejb-jar***-állományt. Ha az ***build.xml***-t a `$CALCULATOR` könyvtárba helyezzük, a ***.java*** fájljainknak, illetve az ***ejb-jar.xml*** és ***jboss.xml*** fájloknak a `$CALCULATOR/il/co/lerner/calculator` könyvtárban kell kerülniük. Az eredményállományokat az Ant a `$CALCULATOR/build/calculator` könyvtárba helyezi, a ***build.xml*** `build.calculator.dir` tulajdonságának megfelelően.
Ha az Ant a `$ANT` könyvtárba telepítettük, lefordíthatjuk ***.java*** fájljainkat, ezáltal EJB-megfelelő ***.jar*** fájllá változtatva őket (az ***ejb-jar.xml*** a kötelező `META-INF` könyvtárba kerül) és beilleszthetjük a JBoss rendszerébe a következő paranccsal:

```
$ANT/bin/ant deploy
```

Számos üzenetet láthatunk a képernyőn, melyek a fordítás és a telepítés menetét írják le. Ha a fordítás vagy a telepítés kudarcot vall, ellenőrizzük, hogy a környezeti változóink jól vannak-e beállítva vagy a Java fájlokban nem maradt-e hiba, illetve hogy a könyvtárak megfelelő jogosultságokkal bírnak-e.
Amennyiben a Jboss-kiszolgáló már fut, a Calculator ***.jar*** fájl telepítésekor észre fogjuk venni, hogy a kiszolgáló újraindítás nélkül, önműködően felismeri és beilleszti a fájlt.
Ez a JBoss egyik nagyon kényelmes szolgáltatása; ha telepíteni szeretnénk egy fájlt, egyszerűen csak bemásoljuk a `$JBOSS_DIST/deploy` könyvtárba.

Írjunk alkalmazást!

Most, hogy végre elkészült egy telepített ***Calculator*** EJB-nk, írjunk egy rövid Java fájlt, amely ki is használja. A *4. listább* van egy ilyen osztály, mely az ***UseCalculator.java*** forráskódját tartalmazza.

Bár programunk teljes mértékben független az EJB-osztályoktól, és külön is lefordítható, valamint futtatható (akár egy másik gépen is), most is az Antot használjuk a `CLASSPATH` követésére (amelynek egyaránt kell tartalmaznia a ***.jar*** fájlunkhoz tartozó és a Jboss-osztályokat), a fordításhoz és a futtatáshoz is. Az alkalmazásunk futtatásához egyszerűen ennyit szükséges begépelnünk:

```
$ANT/bin/ant use-calculator-ejb
```

Az Ant lefuttatja a programunkat, azonban előbb biztosítja, hogy EJB-nk le legyen fordítva, ***.jar*** fájllá alakított és fel van telepítve.

Bármi, amit a `UseCalculator.main()` a ***System.out***-ra ír (más néven `stdout` fájlkezelő), megjelenik a képernyőn, amikor lefuttatjuk az Antot. Ugyanakkor minden, amit a `CalculatorBean` tagfüggvény ír a `stdout`-ra, az a JBoss naplókimenetén fog megjelenni.

Ha az Antot az egyik, a Jbosst pedig egy másik terminálablakban futtatjuk, láthatjuk amint párbeszédet folytatnak egymással.

Az `UseCalculator main()` tagfüggvénye néhány hagyományos lépést alkalmaz EJB-nk elérésére és használatára. Először felvesszük a kapcsolatot a JNDI-vel, amely a Jbossban jelenleg telepített objektumokat tartja nyilván. Ezt a kapcsolatot nevezik `context`-nek. A programunk egy rövid tulajdonság-leíró fájlt, a ***jndi.properties***-t keresi, amelyből megtudhatja, hol találhatja a `context`-et (ezt a fájlt a ***build.xml***-nek megfelelően a `$CALCULATOR/resources/` könyvtárba kell helyezni). Ez a fájl Java erőforrás-formátumú, ahol minden sor `nØv=ØrtØk` alakú.

```
java.naming.factory.initial =
➥org.jnp.interfaces.NamingContexFactory
java.naming.provider.url = localhost:1099
java.naming.fasctory.url.pkgs =
➥org.jboss.naming:org.jnp.interfaces
```

Amint megszereztük a `contex`-et, megkereshetjük az objektumunkat az ***ejb-jar.xml***-ben, azon a néven, amit a ***jboss.xml***-ben adtunk neki. A ***jboss.xml*** nélkül a JBoss nem rendelt volna helyes nevet az EJB-nkhez, ebből következik, hogy nem is tudtuk volna megtalálni a JNDI-n keresztül.

A JNDI egy objektumhivatkozást ad vissza, amit aztán ***CalculatorHome*** példánnyá sorolhatunk be, amivel végül létrehozhatjuk a ***Calculator*** egy példányát. Figyeljük meg, hogy a ***CalculatorBean*** példány helyett a ***Calculator*** (a távoli csatolófelület) egy példányát hozzuk létre. A távoli csatolófelület ugyanis átlátszó kapcsolatot biztosít számunkra a kiszolgálón lévő ***CalculatorBean*** példánnyal, akárhol legyen is az. Így aztán egyáltalán nem kell tudnunk, hol van valójában a ***CalculatorBean*** példány.

Végül, meghívjuk a ***Calculator***-ban (a távoli csatolófelületen) megadott egyik tagfüggvényt. A tagfüggvényhívásunk a `CalculatorBean`-hez (a babosztályhoz) továbbítódik, ahol végrehajtódik, kiír néhány naplóadatot, majd visszatér (ahol aztán az eredményt a `stdout`-ra írjuk).

Összegzés

Írásunkban az Enterprise JavaBeans rejtelmeibe pillanthattunk bele, amely tulajdonképpen Javát használó megosztott alkalmazások létrehozására használható háttér. Igaz, az EJB messze összetettebb, mint a SOAP (lásd Linuxvilág 2001. áprilisi szám 70. oldal), XML-RPC vagy egyéb megosztott objektumrendszerek, azonban sokkal bonyolultabb feladatok kezelésére is tervezték. (Például a SOAP ne is próbálja kezelni a tranzakciókat; hagyja az alkalmazásrétegre ennek megvalósítását.)

Másrészről a Javával való munka gyakran azt jelenti, hogy az ember több időt tölt felügyeleti és logisztikai feladatokkal, mint tényleges programozással. Kitalálgatni, hogy melyik fájlt melyik könyvtárba kell helyezni, elég elkedvetlenítő lehet, különösen, ha valamilyen sokkal dinamikusabb nyelven szoktunk dolgozni, mint például a Perl vagy a Python. Mindazonáltal a fájdalom gyorsan elmúlik, amint meglátjuk, milyen könnyedén tudunk megosztott alkalmazásokat írni az EJB segítségével. Az a tény, hogy a JBoss könnyen letölthető, telepíthető és futtatható, ráadásul viszonylag kevés memóriát fogyaszt, lehetővé teszi mindenki számára, hogy kipróbálja az EJB-t.

Következő alkalommal folytatjuk a munkát az EJB-vel. Belenézünk a belsejébe és megvizsgáljuk az entitás babokat, amelyek a relációs adatbázisok eléréséhez nyújtanak számunkra objektumalapú felületet.

Reuven M. Lerner (reuven@lerner.co.il) kisebb webes és internetes módszerekkel foglalkozó tanácsadó cég tulajdonosa és vezetője. A cikk megjelenésének időpontjában valószínűleg már végleg elkészült Core Perl című könyvével, melyet idén jelentet meg a Prentice-Hall. Az ATF honlapon érhető el (➲ http://www.lerner.co.il/atf/).

Kapcsolódó címek

A JBoss hivatalos oldala a ➲ http://www.jboss.org. A honlapon találunk fórumot, levelezési listát, letöltési oldalakat és leírást is. A Jboss-leírás néhol kicsit zavaros, nem mindig írja le átfogóan a dolgokat és meglehetősen jó EJB-tudást feltételez. Ugyanakkor a legtöbb dolgot, amire szükségünk lehet, megtaláljuk itt. Nyelvezete elég egyszerű és szép példaprogramokat is találhatunk, amelyeket könnyen fel tudunk használni.

Richard Monson-Haefel tollából az Enterprise JavaBeans című O'Reilly által kiadott könyve, kitűnő forrásmunka az EJB-témában. A ➲ http://www..jboss.org bevezetőjével és útmutatóival kiegészítve, e könyv segítségével már viszonylag rövid idő alatt elkezdhetünk egyszerű EJB-ket írni.

Az Ant a `make` és a Makefile fájlok javás változata, az Apache Software Foundation Jakarta projektjének része. Az Antot a ➲ http://*jakarta.apache.org/ant* címről tölthetjük le.

A Sun honlapja a ➲ http://www.java.sun.com rengeteg adatot tartalmaz a J2EE-leírásról, az Enterprise JavaBeans-ről és a kiszolgálóoldali Java általános alkalmazásairól.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Határtalan szórakozás

Hívd ki a régi barátaidat – bárhol éljenek is a világon – egy Rizikó- vagy Monopoly-partira a Világhálón keresztül!

A Linux-közösség legnagyszerűbb vonása, François, hogy valóban közösségként működik. Ez a vonása a Linux születése előtti időkig nyúlik vissza. Ma, több mint tíz évvel azután, hogy monsieur ***Torvalds*** megajándékozta a világot a Linuxszal, a hagyomány folytatódik, a Világháló csak tovább erősíti a számítógép-rajongók közti kötelékeket. Ebben a kapcsolatrendszerben rejlik ennek a közösségnek az ereje, François.

Mais oui, François, igazad van. Ebből az elképzelésből fejlődött ki az üzleti kommunikáció, mégis bizonyára egyet fogsz velem érteni abban, hogy a kulcsszó a kommunikáció… ami nem jelent mást, mint közelebb hozni az embereket egymáshoz – s mi lehetne alkalmasabb eszköz erre, mint a játék? Az ember játékos faj, mon ami. Úgy gondolom, a Világháló igazi célja, hogy a világ bármely pontján lévő embereknek lehetővé tegye az egymással való játékot – ja, és természetesen borászati tapasztalataikat elektronikus levélben cserélhessék ki.

Á, bonjour, mes amis! François, gyorsan a borospincébe, hozd fel kérlek az 1997-es argentin Mendozát a vendégeink számára! Foglaljatok helyet, mes amis, helyezzétek magatokat kényelembe. François és én éppen arról beszélgettünk, hogyan lazíthatunk egy kicsit azokkal a barátainkkal, akik távol vannak tőlünk. Tudjátok, amikor fiatalabb voltam, legszívesebben egy asztal körül ülve, egy üveg bor társaságában hangosan beszélgetve és társasjátékot játszva töltöttük az estéket. A korszerű hálózati technológiák és a Linux segítségével ma is megtehetjük ezt, még akkor is, ha a barátaink a bolygó másik felén élnek. Á, François, merci, kérlek, tölts a vendégeinknek!

Az első ilyen játék, amit meg szeretnék vizsgálni, az egyik kedvencem. Biztosan ismeritek, amikor hajókat kell elhelyeznünk a kockáspapír tengeren. Két ellenfél néz szembe egymással vakon lövöldözve:

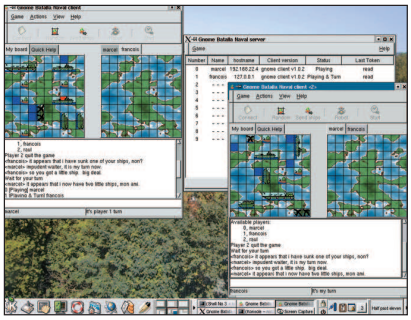
– E-7.
– Nem talált!
– E-8.
– Ó, elsüllyesztetted a rombolómat!

Ezt a játékot akár egy papírcetlin is lehetett játszani, amire mi rajzoltuk be a rácsot és a hajóinkat. Most ha régen elveszett barátod valami távoli vidéken rá tud kapcsolódni a Hálóra, újra megtapasztalhatod a játék örömeit. Sőt, ez a megoldás kicsit még élvezetesebb is. ***Ricardo Quesada*** a szerzője annak a Batalla Naval nevű hálózatban is működő programnak, amely új formába önti ezt a klasszikus játékot. Megszűnt a játékosok számának korlátja, kettőnél többen is beszállhatnak a csatába, így sokkal több az ok az aggodalomra, mint egyetlen ellenfél esetén. A kaland a Batalla Naval honlapjának felkeresésével kezdődik, a legfrissebb forrást a ➲ http://batnav.sourceforge.net címen töltheted le. A program lefordítása sokak számára ismerős lesz:

```
tar -xzvf
➥gbatnav-1.0.2.tar.gz
cd gbatnav-1.0.2
./configure
make
make install
```

Kezdődhet a játék? Először a kiszolgálóprogramot, a `gbnserver`-t kell elindítanod. A Világhálón lehetőséged nyílik egy már futó kiszolgálóhoz való csatlakozásra is. A kiszolgálót futtató gépen egy előugró ablak mutatja az ügyfélgépek kapcsolatának állapotát, 0–9-ig megszámozva. Ezen a ponton lehet a `gbnclient` alkalmazással csatlakozni a játékhoz. Az első ablak, ami megjelenik, a kiszolgálót és a kapu számát kérdezi meg. Kapcsoló nélküli indítás esetén a kiszolgáló az 1995-ös kaput használja. Ekkor választhatsz nevet magadnak, ami alapértelmezésben a bejelentkezési neved. Ha ez megvan, helyezd el a rácson a hajóidat (ebben a megvalósításban legfeljebb tízet) és kattints a ***Send ships*** feliratra. Legalább két játékosra van szükség, de robotokat is indíthatunk a játékban. Ha minden játékos elkészült, a csata a ***Start game*** gombbal indítható.

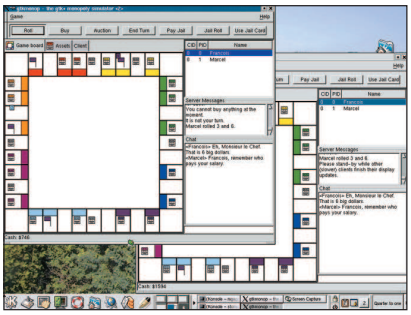
Most már tudunk a hálózaton keresztül játszani – akár egy szobán belül, akár földrésznyi távolságra egymástól. De mi van a baráti ugratásokkal, a játékosan bosszantó beszólásokkal? A korszerű hálózatok adta lehetőségeket kihasználó csevegőfelület a régi és az új találkozásának érdekes élményét nyújtja. A játék menete alatt a résztvevők az ügyfélablak alján lévő szövegmezőbe gépelve küldhetik el az üzeneteiket. Minden üzenet előtt megjelenik a feladójának neve. Az első kép a Batalla Navalt mutatja működés közben.



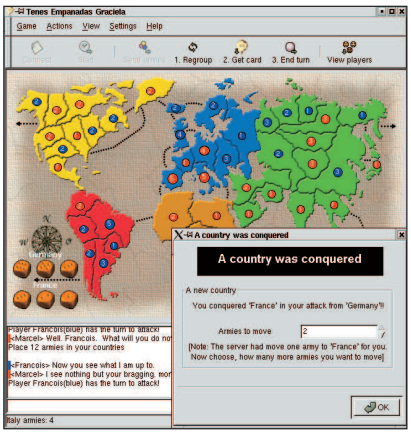
1. kép Egy kegyetlen játék: a Batalla Naval

A következő bemutatandó tétel a mindannyiunkban megbúvó kapitalista ingatlanbárót csábítja el. Minden idők egyik legnépszerűbb játékáról, a Monopolyról van szó. Az évek során rengeteg formája látott napvilágot, például a világ egy-egy nagyvárosát reprezentáló táblával. Atlantic City csak egyetlen helyszín a sok közül a virtuális ingatlanmogul számára.

Ennek a klasszikus játéknak több változata is forgalomban van. A Hálón már futó kiszolgálóhoz is csatlakozhatsz, de amikor én vizsgáltam, egyetlenegyet sem találtam, úgyhogy az a legjobb, ha saját kiszolgálót használunk (különösen ha a saját csapatunkkal szeretnénk játszani). Ilyen alkalmazás ***Rob Kaper*** `monopd` nevű programja is. A `monopd` démon a ➲ http://sourceforge.net/projects/monopd címről tölthető le. A démon összeállításához még egy alkotóelemre, a ***libcapsi_network*** könyvtárra van szükség (szintén Rob Karper alkotása), ami ugyancsak a ***SourceForge*** oldalán érhető el, de más címen: ➲ sourceforge.net/projects/libcapsinet/projects/libcapsinetwork



2. kép A gtkmonop és az ügyletek



3. kép Marcel visszanyeri a fennhatóságot Franciaország felett

A `libcapsi` forrásának kicsomagolásával kezdjük:

```
tar -xzvf libcapsinetwork
      ➥-0.0.13.tar.gz
cd libcapsinetwork-0.0.13
./configure
make
make install
```

A szükséges könyvtár birtokában hasonló lépéseket követve állíthatjuk elő a `monopd` kiszolgálódémont:

```
tar -xzvf monopd-0.2.0.tar.gz
cd monopd-0.2.0
./configure
make
```

A démon a `monopd` paranccsal indul. A program az 1234-es kaput használja, és működés közben kis üzenetek tájékoztatnak a tevékenységéről. Mostanra már valószínűleg futtatod a `monopd` démont, ami boldogan végzi a feladatát a háttérben, de az igazi játék megkezdéséhez még szükségünk van valamire. Oui, mes amis, egyetértek abban, hogy a démonok is remek szórakozást nyújthatnak, de nem kimondottan társasjátékok, igaz? Szükségünk van még az ügyfélprogramra is. A mai menühöz ***Eric Bourget*** gtkmonop-ját fogjuk használni, melyet a ➲ http://gtkmonop.sourceforge.net címről tölthetsz le:

```
tar -xzvf gtkmonop
➥-0.2.0.tar.gz
cd gtkmonop-0.2.0
./configure
make
make install
```

Amikor elindul, három fület vehetsz észre. Az első, a ***Game Board*** feliratú a játék tábláját mutatja. Az ***Assets*** (Tulajdonok) címkével ellátott fülön tájékozódhatsz az ingatlanok formájában felhalmozott vagyonodról. A harmadiknál kezdődik a játék, a becenév és a kiszolgáló kiválasztását saját gépen futtatva (valószínűleg a ***localhost***) teszi lehetővé. A ***Connect***-re kattintva csatlakozhatsz. Amikor a második játékos is belépett, indulhat a mulatság.
Ahogy a Batalla Naval, úgy a `gtkmonop` is lehetővé teszi, hogy a többi játékossal beszélgetést folytass.
A *2. kép* a `gtkmonop` képernyőképeinek egyikét mutatja játék közben.
A Linux világában sokszor tréfálkozunk a világuralmat emlegetve, ami (természetesen) a Linux uralmát jelenti a kiszolgálók világában, az asztali számítógépeken, a kézi számítógépeken és így tovább. Úgy tűnik, néhányunknál ez a rögeszme még a Linux születése előtti időkig nyúlik vissza. Sok évvel ezelőtt rengeteg időt töltöttem barátaim társaságában egy kis játékkal a világ meghódításán fáradozva. Mondanom sem kell, a legtöbbször igen rizikós vállalkozás volt. A játéknak is ez volt a neve: Rizikó. Sokszor gondolok nosztalgiával ezekre a pillanatokra, de nem kell tovább ezen rágódnom, mert régi barátunk, ***Ricardo Quesada***, a Tenes Empanadas Graciela (vagy egyszerűen a TEG) szerzője, úgy gondolta, neki is hiányoznak ezek az idők. Hála neki és elhivatott fejlesztő- és játszócsapatának, a klasszikus világhódító társasjáték fennmaradt.

```
tar -xzvf teg-0.8.0.tar.gz
cd teg-0.8.0
./configure
make
make install
```

A hálózati játék folytatásához a `tegserver` kiszolgálót kell futtatnod. Ne futtasd rendszergazaként (a program nem is engedi ezt). Ha már fut a kiszolgáló, a `tegclient` ügyfél futtatásával indíts új munkafolyamatot. A kiszolgáló nevét kell még megadnod, ami (alighanem) a kiszolgálót futtató gép neve lesz.
A kiszolgálóhoz való csatlakozás után színt kell választanod (piros, kék, rózsaszín, sárga, fekete vagy zöld). Egy játékban legalább kettő, de legfeljebb hat játékos vehet részt. Ha elegendő számú játékos csatlakozott, a játék a ***Start*** gombbal indítható. Stratégiai játékról van szó, és a feladat egyszerű: hódítsd meg a világot! Észre fogod venni, hogy a játék módosított célokkal is játszható, ezek az úgynevezett titkos küldetések. Lehetőség nyílik megfigyelőként is – figyelve a többiek tevékenységét – csatlakozni.
A TEG három térképpel rendelkezik: a klasszikussal, az érzelmessel és a modernnel – ez a kedvencem. Az egyikről a másikra való váltás a játék folyamán a ***Preferences*** (Beállítások) fülre kattintva lehetséges. A *3. kép* egy köztem és hűséges pincérem között folyó játék egyik pillanatát mutatja.
Ahogyan a többi bemutatott játék, a TEG is lehetőséget biztosít arra, hogy játék közben ellenfeleinknek rövid szöveges üzeneteket küldjünk. Majdnem olyan, mint nagyszüleink vidéki házában egy asztal körül üldögélni, nem? A szórakozás, a játékos versengés – minden együtt van, a távolság pedig legyőzetett. A következő alkalomig au revoir, mes amis! Az asztalotok várni fog. A votre santé! Bon appétit!

Marcel Gagné
(mggagne@salmar.com)
Mississaugaban (Ontario, Kanada) él, a Salmar Consulting Inc. rendszerépítéssel és hálózati tanácsadással foglalkozó cég elnöke. Pilóta és sci-fi író egy személyben. A Világhálón elérhető honlapján sok hasznos dolgot találhatunk.
➲ http://www.salmar.com/marcel/

Kapcsolódó címek

Batalla Naval
➲ http://batnav.sourceforge.net
gtkmonop (GTK+ monopd ügyfél)
➲ http://gtkmonop.sourceforge.net
TEG (Tenes Empanadas Graciela)
➲ http://teg.sourceforge.net
A borok főhadiszállása
➲ http://www.winehq.com

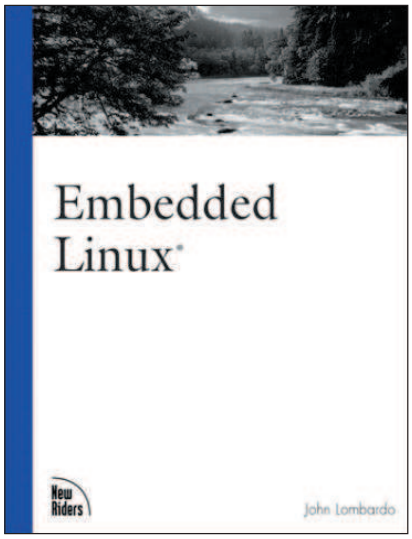
© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Beágyazott Linux: egy új, jókor megjelent könyv

Amerikában a boltok könyvespolcaira frissen kerülő, a beágyazott Linuxról szóló első könyv ismertetése következik, mely kötelező olvasmány mindazok számára, akiket érdekel a téma.

Mikor ***John Lombardóval*** először 2000 augusztusában találkoztam az Embedded Internet Conference-en, John elmondta, hogy egy beágyazott Linuxról szóló könyvön dolgozik, amely körülbelül egy éven belül megjelenik.
Számos program kiadásának elkerülhetetlen késésével szemben Lombardo ***Beágyazott Linux***-a (New Riders, ISBN: 073570998X) határidőre jelent meg a könyvesboltok polcain. Tudomásom szerint ez az első, és e cikk írásának idejében az egyetlen, a témában megjelentetett könyv. Egyből reflektorfénybe került és kötelező olvasmánnyá vált minden fejlesztő (és természetesen bárki más) számára, aki érdekesnek találta a beágyazott Linux témájának felderítését.
A ***Beágyazott Linux*** nagy területet fed le. Négy fő részre osztható: programokra, alkatrészekre, alkalmazásfejlesztésre és kiegészítő hivatkozásokra. E négy fő rész mindegyike tíz fejezetből és négy mellékletből áll, beleértve az olyan fejezeteket is, mint a beágyazott Linuxszal ellátott eszközök indítása, alkatrész-szempontok, ellenőrzés és hibakeresés.
A teljes, részletes tartalomjegyzék elektronikus formában a
➲ http://www.newriders.com/tocs/073570998X.pdf helyen érhető el.
Az Embedded Linux Journal 2001. május–júniusi számában egy példafejezet is található a könyvből.
Ezt a 192 oldalas könyvet egy nap alatt kényelmesen el lehet olvasni, nem bocsátkozik kézzelfogható Embedded Linux Workshop feladatokba (erről lásd bővebben később).
A Linux használata beágyazott rendszerekben és „okos eszközökben" téma alapos feltárása természetesen sokkal több oldalt foglalna el. Nyilvánvaló, hogy számos adatból kiválasztani a legérdemesebb mintegy kétszáz oldalnyit és megjelentetni egy érdekes, hasznos és jó stílusban megírt könyvben hatalmas kihívást jelenthetett a szerzőnek – aki kiválóan oldotta meg a feladatot.
Egy ilyesféle új és gyorsan változó technológiával foglalkozó könyv megírása mindazonáltal versenyfutás az idővel. Először is sürgősen szükség volt már

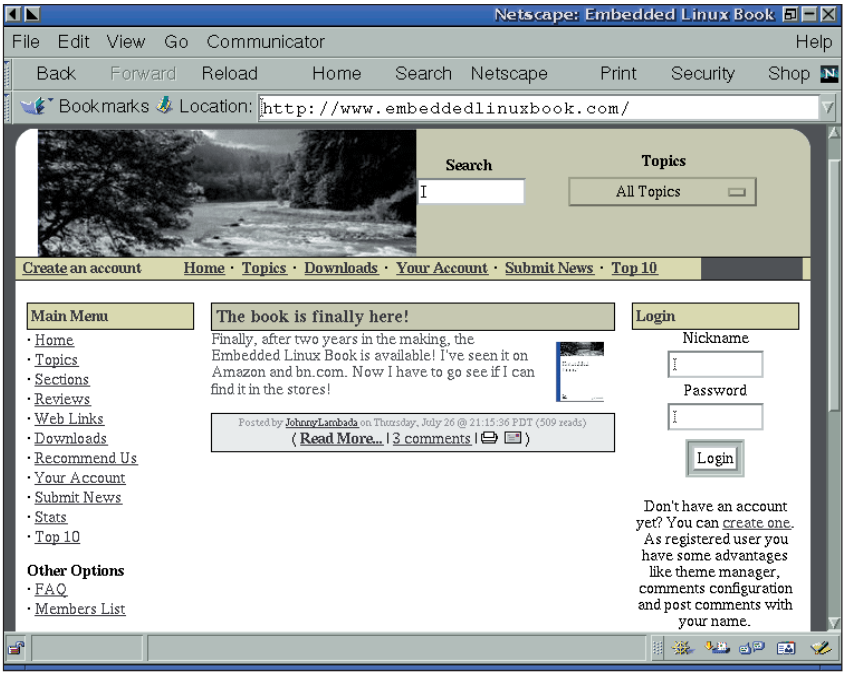


egy a beágyazott Linux témájával foglalkozó könyvre. Bár számos beágyazott linuxos műhely és tanfolyam vehető igénybe, tulajdonképpen csak kevés fejlesztő tud figyelmet fordítani rá – ezért a beágyazott Linuxról szóló technikai irányultságú könyv nagyon keresett. A diákok, a szakemberek és az érdeklődő szemlélők kiéhezetten vártak valamire, ami kiműveli őket ebben a témakörben. Egy ilyen könyvnek a kiadásánál az idő kritikus voltának másik oka az, hogy e tárgy nagyon gyorsan változik. Cégek virágoznak fel és mennek tönkre, terjesztések jönnek-mennek, új technológiák és képességek jelennek meg a színen majd' minden nap. Egy ilyen típusú könyvnél, mint ez is, a piacra dobás időtartamának oly rövidnek kell lennie, amennyire csak lehetséges.
A könyv a Linux beágyazott operációs rendszerként való felhasználásának bemutatásával indul, majd felvázolja a beágyazott Linux lehetőségeit, és fejest ugrik a beágyazott rendszer Linuxon való fejlesztésének érdekfeszítő tárgykörébe, beleértve a programokat, az alkatrészeket és a rendszerszintű szempontokat is. A könyv zsúfolva van jó és hasznos tanácsokkal az általános beágyazott rendszereket illetően, különösen a Linux-fejlesztők számára.
Miután a nélkülözhetetlen program- és alkatrészalapokat bemutatta, Lombardo nekilát taglalni a könyv valódi témáját: az Embedded Linux Workshopot (ELW). Az ELW egy nyílt forrású Linux-programcsomag, amelyet Lombardo a Linux-alapú beágyazott alkalmazások fejlesztési folyamatának egyszerűsítése céljából könyvének társprojektjeként hozott létre.
Több mint 35 oldalt (beleértve az egész hetedik és nyolcadik fejezetet) szentel az ELW-nek. Ebbe beletartozik az ELW programcsomag letöltésének menete lépésről lépésre, és felhasználása egy egyszerű, de valódi beágyazott terv – a beágyazott Minicom – megírására.
Ha valóban meg akarod tanulni a beágyazott Linux-rendszer használatát, az ELW fejezetei kiváló útmutatásokat tartalmaznak egészen az alapoktól.
A teljes ELW-csomag nyílt forrású, és bármilyen asztali PC-re letölthető, beállítható és kipróbálható. Az ELW felépítése olyan, hogy az egész fejlesztési folyamat és a kapcsolódó programok teljes mértékben megtanulhatók és alaposan megvizsgálhatók. Micsoda tökéletes projekt egy ilyen könyv számára, mint ez!
Lombardo ízig-vérig beágyazottrendszer-fejlesztő, és nem marketinges fickó, ezért könyve annyira mentes a marketinges blablától, amennyire csak szeretnéd. Sőt, léteznek fejezetek „A nyílt forráskód hátrányai" és „Amikor a Linux alkalmatlan" címmel, amelyeket az egyensúly megtartása végett írt bele. Ezért biztosan nem fogod úgy érezni, hogy egy vallási fanatikus szavait olvasod.
A könyv hátlapján ez olvasható: *„**A Beágyazott Linux** című könyv beágyazott rendszerek és informatikai eszközök fejlesztői, valamint általános Linux-programozók számára íródott."* Ez ter-

mészetesen nemcsak a programfejlesztőket, hanem a rendszerfejlesztőket is magába foglalja. Sőt, az alkatrészmérnökök is találhatnak benne értékes dolgokat – köszönhetően a számos fejezetnek, melyeket a rendszertervezés témájának és a hibakeresésnek szenteltek. Ezenkívül rengeteg hasznos adat található különböző módszerekről és témákról az általános beágyazott rendszerek fejlesztésével és támogatásával kapcsolatban. Egy szó, mint száz, azt hiszem, hogyha bárki otthonosan szeretné érezni magát a beágyazott rendszerek világában, hasznot húz e könyv elolvasásából, bár a nem programozók azon kaphatják magukat, hogy jó néhány programkódokban bővelkedő részt átugornak (lévén rendszermérnök vagyok, én is becsuktam a szemem az ijesztő programkódrészeknél).

Mindent összevetve, a ***Beágyazott Linux***-ot könnyű és élvezetes olvasmánynak találtam (kivéve a fent említett ijesztő részleteket), és rengeteg hasznos adatra leltem benne. Bepillantást nyújt a beágyazott rendszer fejlesztésének témájába – jól lefedi a beágyazott rendszer felépítésének tervezésének alapjait, beleértve ebbe az alkatrészek és programok kiválasztását, a rendszerfelépítés kompromisszumait stb. A beágyazott program fejlesztési tippjeinek, illetve módszereinek és trükkjeinek kifogyhatatlan forrása is egyben – a szerző tekintélyes mennyiségű tapasztalatot oszt meg az olvasóival, tanácsokkal, trükkökkel, és óvatosságra intéssel segíti a fejlesztőket, hogy munkájukat egyszerűsíthessék és felgyorsíthassák.

Kedvezőtlen apróságként említeném meg, hogy a könyv szűkölködik a forrásokban. Bár általánosságban jól bemutatja a beágyazott Linuxot, egy picit szűkszavú, ha beállításokat vagy a lehetőségeket kell felsorolni. Talán kevésbé kellett volna középpontba állítani egy egyszerű kereskedelmi beágyazott Linux eszközkészletet (LynuxWorks BlueCat), és helyette inkább féltucat főbb vetélytárs programot kellett volna bemutatni a különböző jellegzetességeikkel együtt (egy terjesztéseket összehasonlító táblázat is elkélt volna).



A könyvben könnyen összekeverhető a „beágyazott Linux-változat” és az „eszközkészlet” fogalom – véleményem szerint, a szerző összemossa a különbségeket a beágyazott Linux-változatok (Lineo Embedix, MontaVista Hard Hat és LynuxWorks BlueCat) és a beágyazott Linux-eszközkészlet között. Az eszközkészletet „a készüléket működtető bináris állomány fejlesztésének egyszerűsítésére írt program”-ként határozza meg. Az olyan termékek (Embedix, Hard Hat and BluCat), melyeket a könyvben eszközkészletnek nevez, tartalmaznak egy Linux operációs rendszert és egy eszközkészletet a céloperációs rendszer bináris állományának létrehozására is. Az Embedded Linux Workshopra helyénvaló az a kifejezés, hogy eszközkészlet, mert nem tartalmaz olyan dolgokat, mint például Linux-rendszermag, értelmező, könyvtárak, GNU-segédeszközök. Ebből kifolyólag szívesen látnék beágyazott Linux-változatnak nevezett kereskedelmi termékeket, melyek tartalmazzák az eszközkészleteket és a GNU/Linux operációs rendszert is.

Sőt, az ELW anyagát át kellene dolgozni – ezáltal adva neki rendkívüli használhatóságot, és okos dolog lenne az ELW-ről szóló fejezeteket szétválasztott részekre bontani és részletesebben taglalni. Gyanítom, hogy az ELW önmagában kitenne egy teljes 192 oldalas könyvet, különösen mióta a SourceForge projektet befejezték.

Összegezve, John Lombardónak én 4-es mércén „3,5-es Tux”-osztályzatot adnék. A piacon elsőként megjelent, jól felépített, sok adatot tartalmazó munka, a kiváló Embedded Linux Workshopért és a hozzákapcsolódó lépésről lépésre bemutatott példákért elismerést érdemel.

Kapcsolódó címek

Beágyazott Linux weboldal ➲ http://www.embeddedlinuxbook.com
Friss hibajegyzékek, továbbfejlesztések, kiegészítések, valamint más hasznos beágyazott Linux-forrásokra utaló hivatkozások találhatók ezen az oldalon.
Wookey and Paul Webbútmutató az ARMLinuxhoz fejlesztők számára.
(Aleph One, 2001. április) ez a könyv sokkal kevésbé összpontosít a témára, mint Lombardóé. Második kiadás, elektronikus formában a ➲ http://www.aleph1.co.uk/armlinux/thebook.html helyen olvasható.

Rick Lehrbaum (rick@linuxdevices.com) hozta létre a LinuxDevices.com „beágyazott Linuxok portálját”, amely nemrég visszatért alapítójához, miután az LLC a Device Forge-ot. Rick 1979 óta foglakozik beágyazott rendszerek fejlesztésével. Társalapítója az Ampro Computersnek, alapító tagja a PC/104 Consortiumnak, és fontos szerepet játszott abban, hogy elindulhatott az Embedded Linux Consortium.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

Visszaélés a hálózattal

David a hálózati alapokról értekezik, valamint visszatér néhány korábban felfedezett programhoz.

Linux-felhasználóként számunkra teljesen természetes, hogy hálózatot használunk. A Linux-rendszermag felépítésének köszönhetően mindezt könnyen meg is tehetjük, csakhogy a lehetőségekkel némi rosszindulattal nemcsak élni, de visszaélni is lehet. Ugyan a unixos és a linuxos rendszergazdák az átlagosnál jobban értenek a hálózatok kezeléséhez, azonban szeretnék egy olyan esetre utalni, amely bár windowsos hálózaton történt és oka a hiányos hálózati ismeretekben gyökeredzett, akár egy linuxos hálózaton is megeshet.

Nagyjából másfél éve egy linuxos tűzfal telepítése előtt átnéztem az egyik ügyfelem hálózatát. A hálózat 10 Mb sebességgel működött, és találtam benne néhány különösen hosszú kábelszakaszt (néhány helyen nem is ötös kategóriájú, hanem telefonkábelt használtak), nem megfelelő lezárásokat, amit csak súlyosbított, hogy összesen négy jelelosztót (hub) kötöttek sorba. A hálózaton három protokollt is használtak (IP-t, IPX-et és NetBEUI-t). Talán mondanom sem kell, hogy ilyen körülmények között – 15 százalékos csomagveszteséggel – a hálózat teljesítménye meglehetősen gyengének bizonyult. Néhány hete az ügyfél úgy határozott, hogy lassú, 64 KB-s kapcsolt vonali csatlakozását egy valamivel gyorsabb, 128 KB sebességű vezeték nélkülire cseréli le. Több mint 26 százalékos csomagveszteség elérésekor azonban a hálózat összeomlott. Ma már minden jobban megy, köszönhetően az újratervezett hálózatnak: 100 Mb sebességű rendszert használnak egyetlen protokollal (IP), eltávolították a sorba kötött hálózati elosztókat, és jó minőségű kábeleket szereztek be. A rendszergazdák sajnos nem értették, mi történt. Gúnyolódhatnánk azon, hogy „csak" MCSE minősítésük volt, unixos hálózati ismereteket nem szereztek, de ugyanez megtörténhet bárhol, ha a rendszergazdák nem rendelkeznek megfelelő hálózati tudással. Rengeteg hálózat létezik a világon, és közülük sok eléggé rossz állapotú. Kár tehát nevetni, bármikor belefuthatsz egybe.

Mielőtt belekezdenék mai válogatásomba, szeretném megemlíteni, hogy ezzel a számmal csemegézéseim negyedik évfolyamát kezdem meg. Az eltelt idő alatt az itt bemutatott programok közül sok jelentős fejlődésen ment keresztül, míg mások – legalábbis látszólag – eltűntek. Ezentúl minden hónapban előveszek egy-egy három évvel ezelőtti programot; ha közöttük van a kedvenced, tudasd velem.

Az e havi visszatekintéshez több régi programot is elővettem, közöttük volt például a GTK+ Equation Grapher (geg), a gtkfind (amely a jelek szerint teljesen eltűnt a Webről) és az X Northern Captain, választásom mégis a PySolra esett.

PySol

Amikor a rendszeremet frissítem, egyúttal mindig megpróbálom kicsit tisztogatni is – általában akad elegendő törölnivaló. Az egyik ilyen alkalommal a családom tagjai közül ketten is panaszkodtak, hogy a PySol eltűnt a gépről. Kevés programot használtunk annyit, mint ezt, így annak ellenére, hogy játék, kapott még egy esélyt. Felpumpált Windows Pasziánsznak csúfoltam három éve, azóta azonban rengeteget fejlődött – hangja, sőt zenéje van, és sokféle játékot tartalmaz. Számos kereskedelmi kártyajáték messze elbújhat mögötte. A futtatásához Python szükséges.

➲ http://wildsau.idv.uni-linz.ac.at/mfx/pysol.html

Netdude

A keményvonalas hálózati guruk talán szeretnek ***tcpdump*** állományokat olvasgatni, vélhetőleg könnyen meg is birkóznak velük. Amennyiben az ilyesmiben még kezdő vagy, a Netdude hasznos kis segédprogram. Beolvassa a ***tcpdump*** fájlokat, a kimenetet pedig rendkívül könnyen olvashatóra formázza. Segítségével módosíthatod, majd mentheted a fájlokat. Futtatásához szükséges: libgtk, libgdk, libgmodule, libdl, libXext, libX11, libm, libglib, libpcap és glibc.

➲ http://netdude.sourceforge.net

ifmonitor

Az ifmonitor arra szolgál, hogy a megadott felületet figyelje, amihez adatait a */proc* fájlrendszerből gyűjti, majd egy SQL-adatbázisba illeszti be őket. Ezután az adatokat egy PHP-parancsfájl segítségével érheted el, és egy grafikonon megjelenítheted őket a böngészőben. Mind a telepítése, mind a használata egyszerű. A futtatáshoz szükséges: MySQL, */proc*, Perl, DBD::MySQL Perl-modul, PHP gd-támogatással, MySQL, webkiszolgáló PHP-támogatással, webböngésző.

➲ http://ifmonitor.preteritoimperfeito.com

Manhattan Virtual Classroom

Rendkívül egyszerű, könnyen használható rendszer hallgatók és oktatók számára. Igazi virtuális osztályterem, szerzője a fejlesztés során a biztonságot is mindvégig szem előtt tartotta. Telepítése ugyan nem a legegyszerűbb, de a szerző érthető, tömör leírást mellékelt. Csupán ezt kell követni, és el sem ronthatod. Ha oktató, illetve tanácsadó vagy egy iskolában, a program megérdemel egy bemutatót. A futtatáshoz szükséges: glibc és Apache webkiszolgáló.

➲ http://manhattan.sourceforge.net

Celestia

Rendkívül elragadó háromdimenziós csillagnézegető program, amellyel számítógépedről a teljes ismert Világegyetemet bejárhatod. A grafika egész jó, és a programot rengeteg adattal töltötték fel. Igaz, hogy a látvány lenyűgöző, de az sem mindennapi, amit a program a géppel művel. Lehet, hogy a gépem nem a legújabb gigahertzes csúcsmasina, de nem gondoltam volna, hogy ennyire lassú, míg meg nem próbálkoztam a Celestia futtatásával. Régebbi Pentium I gépen meg sem próbálnám elindítani. A futtatáshoz szükséges: libpng, libjpeg, libGLU, libGL, libSM, libICE, libXmu, libXi, libXext, libX11, libstdc++, libm, libz, libpthread, libdl, libXt és glibc.

➲ http://www.shatters.net/celestia

David A. Bandel