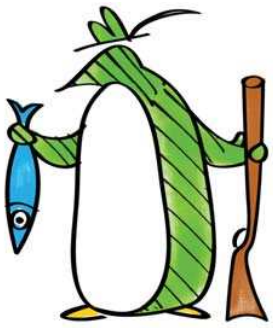
Beköszöntő

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

Ebben a hónapban két fő témakört ölelünk fel lapunk hasábjain, az egyik sarkalatos téma a mindenkit érintő biztonság kérdésköre. A legfontosabb írás a Zorp tűzfalról szóló sorozatunk első része, mely nemcsak azért nagy horderejű, mivel magyar fejlesztésű programról van szó, hanem azért is, mert egy világszínvonalon kivitelezett és elismert programcsomag. A Zorp előtt bemelegítésként néhány olyan programról is olvashatunk, melyek nagymértékben elősegítik rendszerünk feltérképezését. Ha a felméréssel végeztünk, máris nekiláthatunk a Zorp beállításának. Ezenkívül szót ejtünk a biztonságos programozás alapjairól is: mikor melyik függvényt célszerű használni, és melyek alkalmazásáról jobb, ha megfeledkezünk.

A Linux számos üzletágat alakított át – utat mutatva, mely szerint nem csak a régi kereskedelmi minták helyesek és használhatók –, így jött létre a Magnatune internetes lemezkiadó vállalat. A cég érdekessége abban rejlik, hogy szakított a hagyományokkal, nem ad ki kézzelfogható lemezeket, hanem interneten keresztüli letöltést kínál többféle minőségben, amivel a felhasználók kedvük szerint (és természetesen a felhasználói szerződés feltételeinek betartásával) járhatnak el.
További multimédiás alkalmazásokról szóló cikkeink közül ***Komáromi Zoltán*** írását ajánlom, mely az XMMS médialejátszó program lelkivilágába nyújt betekintést.
Olvasóink kívánságára útjára bocsátunk egy új cikksorozatot, mely a Gimp pixelgrafikus képszerkesztő program rejtelmeibe enged bepillantást.
Kellemes időtöltést kívánok!

Programvadászat



A linuxos programok szerencsére gyorsan fejlődnek, ennek következtében a minőségük is folyamatosan javul. A fejlesztők hétről hétre programok igen széles kínálatát vonultatják fel. Ezúttal két, a linuxosok életében igen fontos szerepet játszó, program kapott helyet a korongon: az egyik az Xfree86 projekt 4.4.0-s anyaga, a másik pedig a Gnome 2.6.0-s munkafelület.

Gnome 2.6.0

E kiadás hatalmas előrelépés a Gnome életében, ugyanis ez a változat már a 2.4-es GTK+-ra épül. A mostani megjelenés nem múlt el izgalom nélkül

– bár gondolom, hogy alkotói akkor is aggódnak, ha semmi különleges esemény nem szól közbe –, most viszont a március 24-ei megjelentetést el kellett halasztani 30-ra, mivel feltörték a Gnome.org kiszolgálóját. E kellemetlenség elhárítása után örömmel jelenthetjük, hogy a Gnome 2.6.0-s változat megjelent.
Korongunkra a teljes forráskódot feltettük, ennek segítségével bárki telepítheti, bár nem a legegyszerűbb feladat. A Debiant használók abban a szerencsés helyzetben vannak, hogy `apt` forrásban hozzá tudnak férni, és ilyen módon a teljes környezet egyszerűen telepíthető. (Azt azonban érdemes megjegyezni, hogy ez az egyszerűség csak SID használata esetén érvényes.)
A következő sort adjuk hozzá a */etc/apt/sources.list* fájlhoz:

```
deb http://ftp.hu.debian.org/
➥debian/../project/experimental
➥main
```

Ezután adjuk ki az alábbi parancsokat:

```
apt-get update
apt-get remove libgtop2 (ha esetleg
telepítve van)
apt-get install -t experimental
➥gnome-desktop-environment
```

Azoknak is könnyű dolguk lesz, akik Slackware-t használnak, először fel kell telepíteni a Dropline Installert (GNOME_2.6.0/Slack):

```
installpkg dropline-installer
➥-2.4.6-i686-1dl.tgz
```

Ezt követően adjuk ki a `dropline-installer` parancsot, majd a menük segítségével máris telepíthetjük a programokat.

XFree86 4.4.0

Az Xfree86 projekt ettől a kiadástól fogva megváltoztatta a felhasználói szerződés feltételeit, ennek következtében kisebbfajta ellenállás ütötte fel fejét a linuxosok köreiben, ugyanis felröppent a hír, hogy ez a felhasználói szerződés már nem GPL-megfelelő. Ennek ellenére, például a Slackware is beépítette current ágába. Az Xfree86 oldalán található felsorolásban a következő Linux-változatokat említik meg, melyek szintén tartalmazzák: Slackware, Conectiva, Ark Linux, ArchLinux, Buffalo Linux, JoLinux, Lycoris Linux, Magic Linux, OneBase Linux, OpenNa Linux, Plamo Linux, Rubyx Linux, Slax Live Linux, Source Mage és Sorcerer Linux.
Számuk folyamatosan nő.
Az XFree86 4.4.0 telepítése nagyon egyszerű: adjuk ki az `sh Xinstall.sh` parancsot a rendszerünknek megfelelő könyvtárban, fontos, hogy ne futtassuk működő X alatt (ha esetleg mégis így tennénk, a telepítő is figyelmeztetni fog), mert a grafikus rendszerünket teljesen tönkretehetjük.

PHP konferencia 2004

E lapszámunk 12. oldalán beszámolót olvashatnak a Második Magyarországi PHP Konferenciáról, és eme cikkünkhöz kapcsolódóan a tanácskozás teljes anyagát közreadjuk, pdf formátumban (lásd Magazin/PHPkonf könyvtárban).

Zorp

A kis hazánkban fejlesztett, igen nagy sikerű és tudású Zorp tűzfal GPL felhasználási szerződés alatti változatát is felraktuk a CD-re. Akinek cikkünk olvasása közben sikerült felkelteni a kíváncsiságát és kalandvágyát, az kedvére gyakorlatozhat vele. A Zorp tűzfalalkalmazás a Magazin/Zorp könyvtárban található meg.

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

Kedves naplóm!

Ha azt halljuk, hogy a Nokia 7610-es telefonja korunk elvárásainak megfelelően a telefonáláson kívül számos dologra használható, kamerája 1 megapixeles felbontással dolgozik, négyszeres digitális közelítésre és mozgókép rögzítésére is képes, akkor valószínűleg unottan legyintünk: melyik telefon nem tudja mindezt? Nagyjából így is van, néhány érdekesség mellett azonban kár volna szó nélkül elmenni. A 7610-es ugyanis nemcsak mintegy tízpercnyi film rögzítésére használható, de a felvételt szerkeszteni is tudjuk vele, valamint képi hatásokkal is bővíthetjük, mielőtt például MMS-ben elküldenénk. Ugyancsak egyedülálló a Lifeblog szolgáltatás, amelynek segítségével a trendi felhasználók multimédiás naplóba rögzíthetik életük minden fontosabb eseményét, majd a képeket, hangokat időrend szerint tölthetik át számítógépükre. A Java-támogatással ellátott, MP3 és AAC formátumú zenék lejátszására is alkalmas, Bluetooth-kapcsolatok létesítésére képes telefon tömege 118 gramm, végfelhasználói ára várhatóan 500 euró lesz.

➲ http://www.nokia.com

Bejegyezték

A Toshiba 0,85"-os, vagyis mindössze 2,16 cm-es merevlemezét, mint a világ legkisebb ilyen meghajtóját, hivatalosan is Guiness rekordként ismerték el, s a Guiness 2005-ös, idén ősszel megjelenő kiadványában szerepeltetni fogják. A rekordok könyve rendkívül népszerű, a Toshiba számára pedig bizonyára jól jön a termék népszerűsítése, hiszen a miniatürizálás nem öncélú volt, az év végén szeretnék megkezdeni a 2 és 4 GB-os kivitelben készülő, mobiltelefonokba, fényképezőgépekbe és hordozható adattároló készülékekbe szánt lemezek gyártását.

➲ http://www.harddrives.toshiba.com

256 tagú zenekar

A Silicon Graphics elsőként mutatott be olyan kiszolgálót, amelyben 256 Intel Itanium processzor futtat egyetlen operációsrendszer-példányt, jelen esetben Linuxot. A hatalmasra hizlalt SGI Altix megépítése nemzetközi próbaprogram eredményeképpen állhatott össze, eredetileg nem is terveztek, csak fele ennyi processzort belezsúfolni. Hasonló gépekre számos olyan területen van igény, ahol a nyers erő az első szempont: természettudományos kutatások, mérnöki tervezések, olaj- és gázmezők keresése stb. Az Altix rendszerek több TB megosztott memóriát és a tervek szerint az év végére – a fejlesztések folytatása nyomán – akár 512 processzort is tartalmazhatnak. A nagy számú processzor működtetésében, gyakorlati használhatóságának biztosításában fontos szerep jut az SGI Advanced Linux Environment környezetnek és az SGI ProPack legújabb, 2.4-es számot viselő változatának is.

➲ http://www.sgi.com/software/linux

Kettős látás

A /k/ Embedded Java Solutions és a Quest Innovations közösen fejlesztenek egy MIPS alapú, 32 bites processzormagot, amely Java-bájtkódok és más programok egyidejű futtatására lesz képes. A processzorban valójában két mag lesz, ezek egymással párhuzamosan végzik majd munkájukat, tehát míg a Java mag bájtkódot futtat, addig a másik magon egy különlegesen testreszabott Linux üzemelhet. A két mag – az ennek megfelelően átalakított Linux segítségével – képes lesz egymással együttműködni is, így például, ha a Java-mag túl összetett feladatot kap, akkor a MIPS-magtól kérheti a megoldását, míg ő maga egy másik szállal folytathatja a Java alapú program végrehajtását.
A kettős processzort idén a harmadik negyedévben kívánják megjelentetni, és elsősorban olcsó beágyazott készülékekbe szánják.

➲ http://www.kiffer.be/k/

➲ http://www.quest-innovations.com

Ide gyertek, kedveskéim

A CeBIT-en a Philips több tervezési alapnak szánt, a cég Nexperia médiaprocesszorára épülő szórakoztatóelektronikai készüléket is bemutatott.

A Philips fő szándéka érdekes módon nem az volt, hogy saját lapkáját népszerűsítse, hanem az, hogy rámutasson: az ilyen jellegű készülékek piacra való bevezetése sokkal eredményesebben folyhatna, ha létezne egy közös, az eszközök fejlesztésekor mintaként szolgáló alap. A szórakoztatóelektronikai készülékek közös jellemzője, hogy a vas összeállítása szinte rutinmunkának számít, a felhasználói felület és a programok elkészítése viszont egyre nagyobb hangsúlyt kap. Ugyanakkor a piac erősen szétszabdalt, nincsenek olyan egységes felületek, amelyeket minden vastípuson és operációs rendszeren meg lehetne valósítani. Ezt a gondot oldaná meg a Philips saját tervezésű alapkészülékeivel, míg várakozásaik szerint a programok terén az egységesítés jegyében a Linux juthat egyre növekvő szerephez. A fejlesztésekben résztvevő cégek szeretnék elérni, hogy a linuxos fejlesztői tábor megbarátkozzon a szórakoztatóelektronikai készülékekkel is, valamint az egységes felületek révén meg akarják könnyíteni a számítógépekre írt programok átültetését. Aki rászánja magát, hogy ezzel a területtel foglalkozzon, minden bizonnyal izgalmas kihívásokat talál majd, hiszen itt egészen más elvárásoknak kell megfelelni, a felhasználói felületeket rendkívül könnyen használhatóra kell alakítani, a rendszerindításra legfeljebb néhány másodperc szánható, kevesebb memóriával lehet gazdálkodni, mégis valós idejű szolgáltatásokat szükséges nyújtani.

➲ http://www.semiconductors.com

Kegyvesztett megahertzek

Az Intel a jövőben újfajta processzorszámozási eljárást fog alkalmazni. Az új jelölések, amelyek a Pentium és a Celeron lapkákra egyaránt vonatkoznak majd, nem annyira a processzorok órajelét, sokkal inkább teljesítményét és tudását fogják kifejezni. Egy processzor nem csak attól gyorsabb, hogy nagyobb az órajele, sebességét a gyorstár mérete és az adatok belső mozgatásának gyorsasága is befolyásolja, vélik az Intelnél. Hozzátehetnénk, utolsóként jöttek rá erre, hiszen például az AMD régóta nem – amúgy az Intel lapkáknál alacsonyabb – órajelükkel adja el termékeit.

Az Intel új jelölésében háromjegyű számokat találhatunk majd, ezek hármassal, ötössel vagy hetessel kezdődnek. Egy a 300-as sorozatba eső lapka értelemszerűen lassabb lesz, vagy kevesebb szolgáltatást nyújt majd, mint az 500-as vagy a 700-as sorozat tagjai. A Xeon és az Itanium processzorokra az újfajta jelölés – ez egyelőre inkább a fogyasztók félrevezetésére, mint hiteles tájékoztatására tűnik alkalmasnak – szerencsére nem terjed ki.

Néhány példa:

- Intel Pentium 4 / 520 processzor – 2,8 GHz órajel, 1 MB gyorstár, 533 vagy 800 MHz FSB
- Intel Pentium 4 / 530 processzor – 3,0 GHz órajel, 1 MB gyorstár, 800 MHz FSB
- Intel Pentium 4 / 540 processzor – 3,2 GHz órajel, 1 MB gyorstár, 800 MHz FSB
- Intel Pentium 4 / 550 processzor – 3,4 GHz órajel, 1 MB gyorstár, 800 MHz FSB

➲ http://www.intel.com

.mobil

A mobil készülékek iparágának jeles képviselői egy mobileszközök számára elkülönített legfelsőbb szintű internetes tartománynév (TLD) üzembe helyezését kérik az ICANN-tól. Az előterjesztést a 3, a GSM Association, a HP, a Microsoft, a Nokia, az Orange, a Samsung, a Sun és a Vodafone támogatja, és amennyiben sikerrel járnak, a cégek közös szervezetet alakítanak a TLD fenntartására. A cégek képviselői szerint az új tartománynév az internet és a hordozható eszközök világa közötti rést segítene szűkíteni, ez a felhasználók és a gyártók szempontjából egyaránt előnyös lenne. Az elképzelések szerint a bővítés után egyszerűbbé válna a mobil felhasználóknak szánt webhelyek és egyéb szolgáltatások indítása, elérési sebességük nőne, használatuk pedig egyszerűsödne. A mobil TLD pontos neve egyelőre ismeretlen, azonban a döntést az ICANN fogja meghozni.

➲ http://www.mtldinfo.com/

Videózás bővítőkártya nélkül

A Pinnacle Systems három, elsősorban otthoni felhasználóknak szánt PC-s videoeszközt mutatott be. Mindhárom egység külső, a számítógéphez kizárólag USB 2.0-s kapun keresztül csatlakoztatható.

A kisméretű, beépített vevőegységgel rendelkező PCTV USB2 tévéműsorok megtekintésére és felvételére alkalmas. Külső tápellátást nem igényel, a hordozható gépek tulajdonosai számára is könnyen használható, a gyártó időeltolásra alkalmazható programot is mellékel hozzá.

A Dazzle DVC (Digital Video Creator) 90 tenyérnyi méretű kiegészítő, segítségével digitális videokamerát lehet könnyedén csatlakoztatni a számítógéphez, majd – akár a mellékelt Pinnacle Studio QuickStart program segítségével – a felvétel átszerkeszthető, képi hatásokkal színezhető.

A MovieBox Deluxe analóg és IEEE 1394 szabvány szerinti be- és kimenetekkel rendelkezik, segítségével például digitális videokamera csatlakoztatható USB 2.0-s kapuval felszerelt számítógéphez. A MovieBox Deluxe vásárlói a Pinnacle Studio 9 és a Hollywood FX Plus alkalmazásokhoz is hozzájutnak.

Az újdonságok ára átszámítva sorrendben 20 ezer, 18 ezer és 60 ezer forint.

➲ http://www.pinnaclesys.com

Nem tűnik el, csak megvették

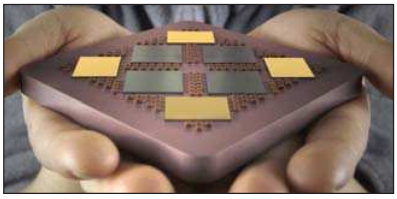
A Euroweb International 9,5 millió dolláros áron megvásárolta az Elender Üzleti Kommunikációs Rt. száz százalékos tulajdonjogát. Amennyiben a Gazdasági Versenyhivatal jóváhagyja a vásárlást, akkor az Euroweb Internet Szolgáltató Rt.-t és az Elendert összevonják, ezzel a honi piac vezető független internetszolgáltatója jön létre, illetve az új cég a vállalati szegmensben is első helyre lép majd. Az összevont vállalat vezetője az Elender eddigi tulajdonos-vezérigazgatója, ***Kóka János*** lesz.

A Euroweb további felvásárlásokkal mind hazánkban, mind a régióban további terjeszkedést tervez.

➲ http://www.euroweb.hu

Átszabható processzorok

A jövő IBM processzorai képesek lehetnek önmaguk módosítására, például több memória használatba vételére vagy egyes lapkarészek letiltására, használójuk elvárásaihoz igazodva. A cég reményei szerint



a Power processzorai ezzel a módszerrel sokféle területen lesznek alkalmazhatók, a csúcskategóriájú vállalati kiszolgálóktól kezdve egészen a játék- és a kézigépekig.

A mikroprocesszoroknál a fejlesztés évekig az egy lapkára zsúfolt tranzisztorok számának növelését jelentette, mára azonban fizikai korlátok szabnak határt az ilyen irányú továbblépésnek. Egyre átgondoltabb, okosabb szerkezeteket kell tervezni, ennek egyik megvalósítási módja az önműködő lapkák készítése – vallják az IBM-nél. A megoldás működése egy eddig inkább kellemetlennek tartott jelenségen alapul, mégpedig azon, hogy a vékony, folyamatosan áram alatt lévő fémvezetékek atomjait az elektromos áram idővel kiüti, így a vezetékek elvékonyodnak majd elszakadnak. Az IBM fejlesztői kidolgoztak egy módszert, amellyel a vezetékszakadás szándékosan is előidézhető, és bár a hagyományos tervezői szemlélettel már maga az ötlet is hátborzongató, a lapkák testreszabása mégis így válik lehetővé.

➲ http://www.research.ibm.com

Tölt és indul

Az NEC mérnökei a jelenlegi akkumulátoroknál lényegesen gyorsabban, mindössze 30 másodperc alatt feltölthető akkumulátortípust fejlesztettek ki. Az új akkumulátorok teljesítménye elődeikével egyező, a segítségükkel táplált készülékeknél az üzemidő csökkenésére nem kell számítani.

A gyors töltés kényelmi szempontból rengeteg előnnyel jár, és itt nemcsak a mobiltelefonokra, de az akkumulátoros borotvakészülékektől kezdve egészen a lemerült akkumulátorú autók elindításáig számos alkalmazási területre érdemes gondolni. Az NEC újfajta akkumulátora gyors kisülésre is képes, így olyan helyeken is használható, ahol viszonylag rövid ideig, de nagy árammennyiségre van szükség. Az NEC meglévő gyárait hamarosan az új áramforrások termelésére fogja átállítani, ezek előállítási költsége – köszönhetően annak, hogy semmilyen különösen drága anyag felhasználására nincs szükség – várhatóan nem lesz magasabb jelenlegi társaiknál.

➲ http://www.nec.com

Gimp 2.0

Az 1995-ben megindult fejlesztés nyomán immár elkészült a GNU Image Manipulation Program 2.0-s változata. A legújabb kiadás önkéntes programozók három éves munkájának eredménye, s egyben az első olyan üzembiztos változat, amely hivatalosan is támogatott nemcsak Unix alapú, de Mac OS X és Windows-rendszereken is. A Gimp 2.0-s újdonságai az átlagos felhasználó számára részben láthatatlanok, kódját ugyanis teljesen átszervezték, így a

korábbinál sokkal áttekinthetőbb és könnyebben karbantartható lett. Noha szolgáltatások tekintetében igazán hangsúlyos újdonságokat nem tartogat az új program, az alapszintű eszközök működését, használhatóságát számtalan ponton javították, illetve maga a felhasználói felület is – bár a kisebb módosítások ennél szintén inkább csak a jobb áttekinthetőséget szolgálják – az elődjénél több lehetőséget kínáló GTK+ 2 eszközkészletre épül. Az újdonságok listája a program szintén megújított weboldalán található leírásban olvasható.

➲ http://www.gimp.org

Medgyesi Zoltán
(mz@rettesoft.hu)
A Linuxvilág hírszerkesztője. Szabadidejét legszívesebben a barátnőjével tölti, szeret autózni és bográcsban főzni.

Zorp 3 a CeBIT-en

A hazai piacon már jól ismert BalaBit Kft. minden eddiginél biztonságosabb tűzfal-technológiát mutatott be az informatika egyik legrangosabb nemzetközi vásárán, a hannoveri CeBIT-en.

A BalaBit IT Security neve nem ismeretlen a hálózati informatikusok számára, ugyanis ez a magyar csapat fejleszti a világ legelterjedtebb választható unixos rendszernaplózó programját a `syslog-ng`-t is, amit többek között a világ legnagyobb bankjai, az amerikai Védelmi Minisztérium és a NASA is használnak. A cég tavaly már szerepelt a CeBIT-en, ennek eredményeképpen a német piacon is elérhető a Zorp nagy biztonságú tűzfal. Idén a termék harmadik változatát mutatták be Hannoverben, ez a nagyon összetett, éppen ezért testreszabott megoldásokat igénylő nagyvállalatok számára készült. A fejlesztés során több mint harmincezer felhasználó próbálta ki sikerrel az új technológiát. A korábbi változathoz képest a Zorp 3 lényegesen magasabb szintű ellenőrzési (monitoring) és hitelesítési (authentication) lehetőségeket kínál. A fejlesztői csapat az összetett hálózati környezethez igazodva jelentősen finomította telepszolgáltatásait (cluster function). Ergonómiai megfontolások alapján a közvetlen felhasználói vezérlés lehetőségét biztosító grafikus csatolófelület (GUI) is átalakult, ennek köszönhetően az eszköz kezelése könnyebbé vált.

Az fejlett tűzfal-technológia előnyeiből a következőket emelték ki:

- A technológia a csomagszűrés és a protokollelemzés kettősét egészíti ki a részletes adatelemzéssel. Különleges eljárással, a 16 leggyakrabban használt protokoll esetében, a teljes adatfolyam értelmezésére, elemzésére és módosítására képes.
- A rendszer proxyjai a tűzfal egy-egy modulját képezik, amelyek összeköthetők, így a módszer alkalmas a beágyazott protokollok (pl.: HTTPS, POP3S) kezelésére, ezért az egyre népszerűbb e-business alkalmazásoknak is biztonságot nyújt.
- A Zorp esetében nem kell a biztonsági szabályzatokat a tűzfal műszaki lehetőségeihez szabni, mivel a felhasználói felületen végrehajtható programozás lehetővé teszi a bonyolult és összetett beállításokat, ugyanis a protokollok jellemzőihez (parameters) eseményeket lehet rendelni. A módszerben rejlő különleges lehetőségek a szigorú biztonsági igényeket támasztó, szakaszolt nagyvállalati környezetben érvényesülnek legjobban.

Jelenleg Zorp tűzfal védi a magyar kormányzati informatikai rendszer több érzékeny pontját. A BalaBit Kft. ügyfelei között tudhat számos bankot, elektronikus médiával foglalkozó cégeket, az energiaszektorban tevékenykedő vállalatokat, több hazai múzeumot, valamint számos honi nagyvállalatot. Tavaly született meg az a megállapodás, amely az első lépés volt a BalaBit tevékenységének külföldi kiterjesztése során. A BSI (Bundesamt für Sicherheit in der Informations technik) minősítéssel rendelkező, a német piacon jelentős szereplőként számon tartott help AG-val együttműködés született a Zorp németországi értékesítésére. A tavalyi sikereket követően a CeBIT-en való idei részvétel célja, hogy a cég a Zorp forgalmazását nemzetközi szinten elősegítse, hosszú távon pedig a szakmával mintegy szabványként elismertesse a tűzfalak következő nemzedékének első tagjaként.

A BalaBit a magyar hálózati védelem piacának egyik legelismertebb szereplője. A cég a tevékenységét 1996-ban a `syslog-ng` naplófájlelemző program kifejlesztésével kezdte meg, amely később bekerült a Debian Linux-terjesztésbe és ma már megközelítőleg kétszázezer kiszolgálón fut. A folyamatos szellemi innovációra és az „üzleti angyalok" által megteremtett szilárd gazdasági háttérre támaszkodva a BalaBit Kft. hihetetlenül gyors fejlődésnek indult. A cég életében a következő mérföldkő a Zorp tűzfalrendszer kifejlesztése volt, amely a világ első olyan proxytűzfala volt, amely képes az általa ismert protokollok és beágyazott protokollok valamennyi parancsának felismerésére.

A technológia tavaly elnyerte a Informatikai és Hírközlési Minisztérium „Leginnovatívabb üzleti megoldás" díját.

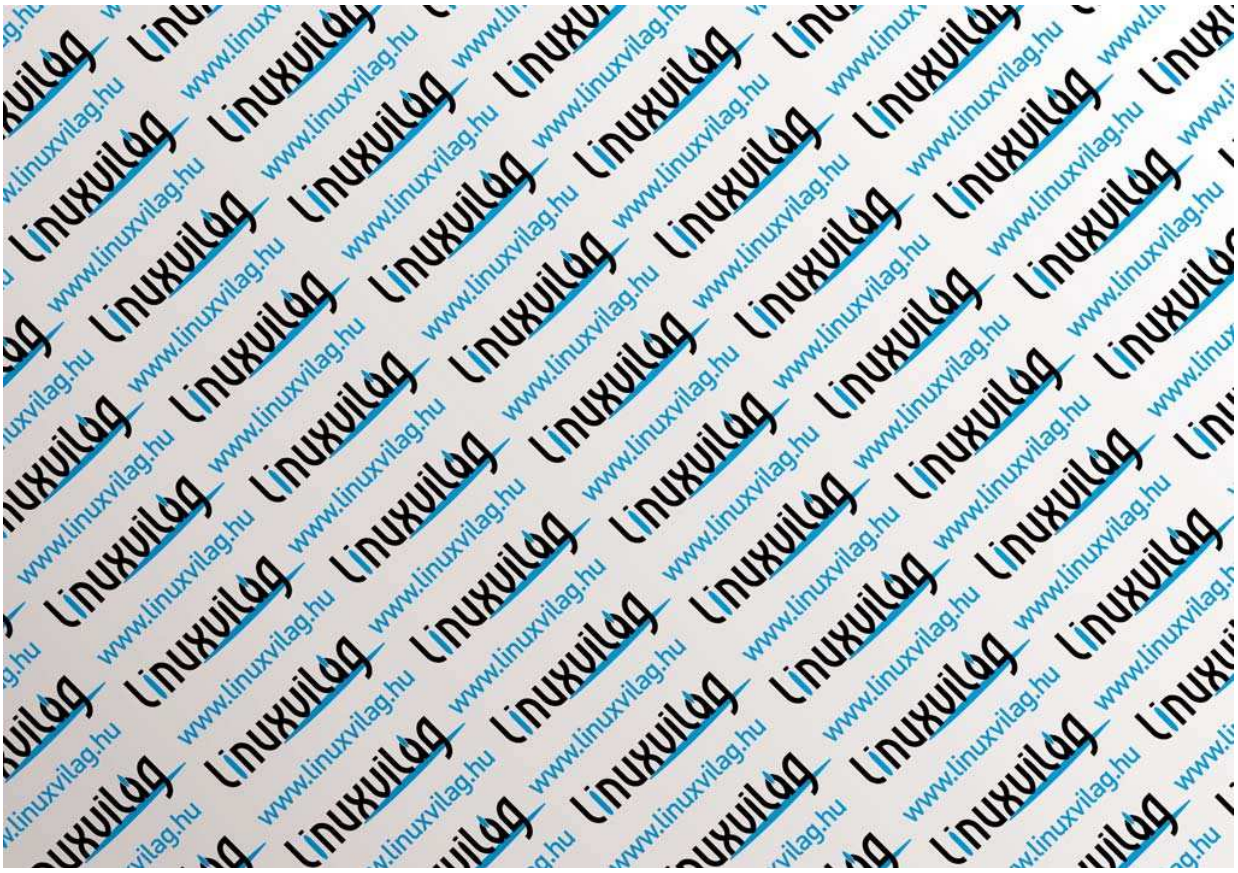
A fejlesztések eredményeképpen a Zorp moduláris szerkezete alkalmassá vált víruskereső modul integrálására. Ezzel a Zorp kereskedelmi változata máig az egyetlen módszer, amely a megfelelő kiegészítéssel képes az általa ismert protokollokban – például HTTP, FTP, POP3, IMAP és ezek titkosított változataiban – vírust keresni.

A BalaBit jelenleg is a leggyorsabban növekvő vállalatok között szerepel; viszonyításképpen a cég árbevétele 2003-ban a 2002. évi 280%-a volt, eközben az alkalmazottak száma nyolcvan százalékkal nőtt.

Linuxvilág

Linux-tábor 2004

Negyedik alkalommal vehetnek részt az érdeklődők a Linux-felhasználók Magyarországi Egyesületének támogatásával megrendezett Linux-táborban, Szerencsen.
Az immár hagyományosnak tekinthető Linux-tábor ismét megrendezésre kerül. Az egyhetes turnusokban 2004. július 4-től – július 10. és július 11-től – július 17. között lehet résztvenni. A tábor programjának középpontjában az érdeklődők Linux terén szerzett ismereteinek bővítése áll, emellett természetesen marad idő a kikapcsolódásra is. A Linux ismerete egyébként nem előfeltétele a jelentkezésnek.
A tanfolyam anyagának elsajátításában a legjobb linuxos szakemberek segítenek, a tematikát az igényeknek megfelelően alakítják ki. A résztvevőket a szerencsi gimnázium épületéhez közeli kollégiumban helyezik el. Itt természetesen lehetőség nyílik internetezésre és levelek olvasására is. A tudás bővítését a délelőtti oktatás szolgálja, a nap további részében pedig számos egyéb program várja az érdeklődőket: többféle sportolási lehetőség, túrázás és borkóstolás a tokajhegyaljai borospincékben. Az egyhetes programot a már hagyományos bográcsozás zárja.
A részvételi díj tartalmazza a szállást, a napi háromszori étkezést, a tanfolyamokat, a tábori pólót, valamint a borkóstolót. Amennyiben május 17-ig jelentkeznek az érdeklődők, 27 000 Ft-ért (LME-tagok 24 000 Ft) vehetnek részt a Linux-táborban. Akit csak a szakmai program érdekel, és a szállást, étkezést maga kívánja megszervezni, annak a szervezők május 17-ig mindössze 14 000 Ft-ért (LME-tagoknak 11 000 Ft) teszik lehetővé a tanfolyam látogatását. Szívesen várnak gyermekeket is, szülői kísérettel.
A táborban az oktatás három független csoportban (kezdő, haladó és profi) zajlik. A tanfolyam során elsősorban a Debian GNU/Linuxra alapoznak. Az első nap a Linux telepítésével kezdődik, mely a csoportok tudásszintjétől függően másképp alakul. A kezdőknél a leggyakoribb témák a grafikus rendszer telepítése és beállítása, valamint a különböző eszközök (pl.: hangkártya) beüzemelése. A haladó témák iránt érdeklődők megismerkedhetnek többek közt a rendszerbiztonság, az alapvető hálózati és belső biztonság, a levélkiszolgáló, a proxy-kiszolgáló témaköreivel, míg a profik e szolgáltatások rejtett trükkjeibe, valamint a Linuxon megvalósítható biztonsági megoldások magas fokú ismereteibe nyernek bepillantást.
További tájékoztatás és jelentkezés a ➲ http://linuxtabor.hu weboldalon lehetséges. A táborral kapcsolatos kérdéseket a linuxtabor@webhome.hu címre várják. A szervezők levelezőlistát és webes fórumot is indítottak, melyek a weboldalról érhetők el.



PHP Konferencia – Magyarországon másodszor

Immár másodszor rendeztek hazánkban PHP konferenciát a Neumann János Számítógép-tudományi Társaság támogatásával.

A társaság több mint harminc éve munkálkodik az informatikai társadalom aktív formálásán, a tudományos és szakmai területeken a kutatás–fejlesztés és a szervezés feladatából vállalva részt a hazai informatika életében. A tavaly megrendezett első PHP konferencián megalakult a Neumann János Számítógép-tudományi Társaság Webalkalmazások Fejlesztése Szakosztály. Részben ez a független szakmai közösség áll a méltán népszerű Weblabor című webes technológiákkal foglalkozó hírmagazin mögött is (➲ http://www.weblabor.hu). A hírmagazinhoz tartozó levelezőlisták egyike a PHP-val foglalkozik, a webes programozók ezen a listán beszélhetik meg a PHP-val kapcsolatos tapasztalataikat, kérdéseiket valamint gondjaikat.
A szakosztály a webszabványt kidolgozó World Wilde Web Consortium Magyar Irodájával is nagyon jó kapcsolatot tart fenn. Fenti szervezetek egymástól független szakmai közössége idén – a tavalyi rendezvény nagy sikerén felbuzdulva – ismét megrendezte a Második Magyarországi PHP Konferenciát.
2004. március 27-én, szombaton már kora reggel regisztrálhattak az érdeklődő programozók, rendszergazdák, szakmabeliek valamint a PHP nyelv iránt bármilyen okból vonzalmat érzők.
Azok, akik a konferencia honlapján a ➲ http://www.phphost.hu címen előre bejegyezték magukat és a visszaigazoló kérdőívet is kitöltötték, egy a PHP konferencia logójával ellátott papírszatyorban szakmai anyagot vehettek át, amelyben a PHP konferencia idei programfüzete is megtalálható volt. Ebből tájékozódhattak az egész napos rendezvény előadásairól (rövid vázlatuk alapján) és a konferencia támogatóit is megismerhették.

CMS Maraton

Az érdeklődők, a rövid megnyitót követően, egyből szakmai előadást hallgathattak „CMS Maraton" címmel.
Heilig Szabolcs, ***Hojtsy Gábor***, ***Illés Szabolcs***, ***Palócz István*** három tartalomkezelő rendszert (Content Management System – CMS), a PHP-Nuke-ot, a Drupalt és az eZ publisht hasonlították össze, nagy vonalakban. A három rendszer részletes összevetésére természetesen egy nap is kevés lett volna, azonban a hallgatóság így is támpontokat kapott a számukra megfelelő tartalomkezelő rendszer kiválasztásához.
A CMS Maraton végén az érdeklődők két csoportot alkottak. Az egyikben ***Kolman Nándor*** jóvoltából a PHP újdonságaival ismerkedhettek, a többiek két rövidebb előadást hallgathattak meg szintén a PHP 5 lehetőségeiről. Először ***Bóna László Márton*** „Objektumorientált programozás a gyakorlatban", majd ***Ercsey Balázs*** „Zenetár a webszerverünkön, avagy XML használata a PHP 5-ben" című gyakorlati bemutatóját kísérhették figyelemmel az érdeklődők.
Rövid szünetet követően a szervezők ismét több részre osztották a közönséget. A konferencia arany fokozatú támogatója, az Interware nemcsak anyagilag, de szakmailag is támogatta a rendezvényt. ***Kovács Zsolt***, az Interware Szerver Hotel igazgatója a beszédes „Segítség! Felnőttem!" címmel tartott előadást a nagy terhelhetőségű, magas rendelkezésreállású rendszerek építéséről, valamint üzemeltetéséről.
Ezzel párhuzamosan ***Mocsnik Norbert*** elméleti összefoglalóját („PEAR – a PHP gyümölcse") és ***Horváth Zoltán*** „Fejlesztés PHP-Nuke rendszerre" című gyakorlati előadását követhette végig a hallgatóság.
A program ***Bártházi András*** „Így készült a Weblabor" bemutatójával és ***Károly György Tamás*** „PHP és Perl – két dudás egy csárdában" című, a két nyelvet összehasonlító elemzésével folytatódott.
Ezután ismét hosszabb szünet következett, ami alkalmat adott arra, hogy az eddig csak interneten találkozó emberek különböző közösségeik találkozóit a konferencián megtarthassák. Például a PHP-Nuke-közösség, a weblabor levelezőlisták közössége és az Neumann János Számítógép-tudományi Társaság Webalkalmazások Fejlesztése Szakosztály tagjai is személyesen találkozhattak és beszélgethettek egymással.
A szünet elegendő időt adott arra is, hogy a konferencia résztvevői – támogatóink jóvoltából – az ingyenesen biztosított élelmet is magukhoz vegyék.

A délután

A délutáni program három különböző teremben párhuzamosan zajlott. Az egyik teremben ***Papp Győző*** „PHP Chemotox" című, a hibakereséssel és a programtervezéssel kapcsolatos előadását, ***Kolman Nándor*** útmutatója követte az „Általános űrlapkezelés"-ről. A másik teremben ***Nováki Szilárd*** a „Gomba for PHP fejlesztési környezet"-ről beszélt,

majd ***Szalai Ferenc Attila*** „PHP a grid technológiában – egyszerűen és célratörően” előadását és végül ***Forstner Bertalan*** „Elterjedt technológiákra építő webes fejlesztőrendszer”-ről szóló elméleti fejtágítóját hallgathatta meg a nagyérdemű.
A harmadik helyiségben szintén három előadást hallgathattunk meg: először ***Mocsnik Norbert*** a „Webes alkalmazásfejlesztés PEAR csomagokkal” gyakorlati bemutatójából tudhattunk meg többet a PEAR-ről (PHP Extension and Application Repository), később ***dr. Baranyai László*** a „Webszabványok hazai alkalmazásának statisztikai elemzése” címmel tartott ismertetője után Mocsnik Norbert folytatta előző előadását „Hatékony fejlesztést segítő további PEAR csomagok” címmel.
Az előadások kivitelezését sajnos néhány nem kellő gondossággal kiválasztott program és rakoncátlankodó gép nehezítette meg – főleg, amikor a rendszer megmakacsolta magát és nem volt hajlandó életre kelni –, ám a közönség érdeklődése és az előadók szakmai rátermettsége a feledés jótékony homályába burkolják az elsötétült képernyőt, illetve a Lanka.bmp csodásnak épp nem nevezhető látványát…
A konferencia vége felé közeledve az értő közönségben is kérdések halmozódtak fel, hiszen ennyi információ befogadása a szakmabeliek számára sem könnyű feladat. Rövid szünet után a fórumon lehetőség nyílt a megfogalmazódott kérdések felvetésére, amelyeket az előadók és szervezők készségesen meg is válaszoltak.

5k compo

A konferencia zárásaként az „5k compo” eredményhirdetése és a regisztrált résztvevők közötti ajándéksorsolás következett. Az 5k compókról (5k netto – freestyle compo) érdemes tudni, hogy ezek 5 kB méretű PHP-programok, amelyeket lelkes programozók küldtek be a PHP konferencia honlapjának felhívására. A beküldött programok megtekinthetőek a weblap galériájában, szavazni azonban csak a konferencia előtt lehetett, hiszen az eredményhirdetés – mint fentebb már említettük – a konferencia záróeseménye volt. A legjobbak díjakat is kaptak, köszönet érte támogatóinknak, akiket a teljesség igénye nélkül említenék meg:

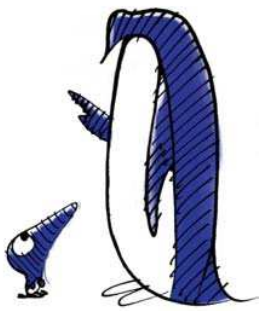
- Interware Rt.,
- Senorg Hungary Rt.,
- Colossal Rendezvénytechnikai Szolgáltató Kft.,
- G'Roby Élelmiszerdiszkont Hálózat,
- Kiskapu Kft.,
- Pentaschool Oktatási Központ Kft.,
- PHP Magazine,
- Sprint Computer Kft.,
- valamint a Weblabor.

Jövőre ismét lehetőség nyílik rá, hogy a bátrabb kóderek megmérettessenek egy igazán szép küzdelemben…
A rendezvény honlapján az 5k compo mellett megtalálható az előadások anyaga, illetve az előadókról is olvashatunk. Ezen az oldalon egyébként a tavalyi előadások anyagai között is böngészhetnek az érdeklődők.
A konferencia nem jöhetett volna létre, ha anyagilag és szakmailag nem támogatják azok a cégek, illetve magánszemélyek, akik szívükön viselik a szabadon terjeszthető, ingyenes, nyílt forrású programok térnyerését.
A konferencia támogatója volt többek között a Linuxvilágot is kiadó Kiskapu Kiadó.

Összegzés

Összességében egy jól megszervezett és szakmailag kerek konferencián vehettek részt azok, akiket érdekelt a PHP és a hozzá kapcsolható szabadon és ingyenesen terjesztett, nyílt forrású fejlesztések világa. A szervezők és előadók nem titkolt szándéka, hogy az informatikai társadalmat a szemléletformálás és az oktatás eszközével tereljék a szabad programok (PHP, PEAR, Perl, Apache, Linux, BSD stb.) és a szabványok (XML, XHTML, CSS) használata felé.
A konferencia szervezői jövőre is várják az érdeklődőket.

Károly György Tamás (➲ http://kgyt.hu)
Szabadúszó webdesigner. 1993-ban kezdett hobbiból weblapokat tervezni, ma már ez a fő tevékenysége. A PHP-val viszonylag friss a barátságuk, 2003 előtt szinte kizárólagosan Perlben dolgozott. A tavalyi PHP konferencián döntött úgy, hogy komolyabb feladatok megoldására is kipróbálja ezt a nyelvet, azóta szinte kizárólag PHP-t használ webes munkáihoz.



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

A hónap szakmai tanácsai

ISA hangkártya támogatása

Akadt egy kis nehézségem a hangkártyám beállításával a Red Hat 7.2-es rendszeren. Amikor az `sndconfig`-ot futtatom, azt az eredményt adja visza, hogy van egy ISA PNP kártyám, pontosabban egy SoundBlaster 32 Wavetable. Amikor kipróbálná a kártyát, akkor viszont az alábbi hibaüzenetekkel tér vissza:

```
/sb.o : init_module: no such device
/sb.o : insmod
/sb.o : failed
/sb.o : insmod sound-slot-0 failed
```

A `dmesg` parancsot futtatva az alábbi üzeneteket kapom:

```
sb: No IsaPnP cards found, trying standard
➥ones...
sb: I/O, IRQ, and DMA are mandatory
No detected device
```

A listából más SoundBlaster kártyákat kiválasztva is ugyanezeket az üzeneteket kapom.

Joseph Helton, hteam1@mindspring.com

Úgy tűnik, a kártyád olyan be-kivitel (I/O) címre, IRQ-ra (megszakításkérésre) és DMA-ra van beállítva, amelyeket a rendszermag magától nem tud felismerni. Próbáld meg az alábbi sort hozzáadni a */etc/modules.conf* fájlhoz:

```
option sb io=0x220 irq=5 dma=1 dma16=5
➥mpu_io=0x330
```

E sorokban a számokat a saját kártyád által használt értékekkel kell behelyettesítened.

Jim Dennis, jimd@starshine.org

Az internetelérés költségeinek alacsonyan tartása

Széles sávú ADSL internet-kapcsolatomnak állandó a havi díja, egészen addig, amíg nem haladok meg egy megadott legnagyobb letöltési határt, ami jelenleg 3 GB. PPPOE-kapcsolatot használok az internet elérésére. Azt szeretném tudni, hogy létezik-e olyan alkalmazás, amivel nyilván lehet tartani a forgalmazott bájtok mennyiségét?

Guillermo Gimenez de Castro, guigue@craam.mackenzie.br

Sok ilyen alkalmazás van: például az ipac IP-számlázó csomag, az MRTG multirouter forgalommérő, ami nemcsak egyszerűen összegzi a forgalmat, hanem grafikonon ábrázolja is a használatát. Az `ifconfig` parancsot szintén használhatod a fogadott (RX) és elküldött (TX) bájtok számának megnézésére. Mindent összevetve, valószínűleg az ipac csomag tűnik a legegyszerűbb és célszerűbb megoldásnak. Letölthető a ➲http://www.daneben.de/ipac.html címről.

Jim Dennis, jimd@starshine.org

A `sar` parancs számos statisztikai adattal szolgál a hálózati csatolóról. Rendszergazdaként próbáld ki az alábbi utatasítást:

```
sar -n FULL
```

Ez egy időbeosztásos táblázatban írja ki más adatok mellett az elküldött és fogadott bájtok számát. A `man sar` paranccsal lehet több útmutatást kapni a használatához.

Felipe Barousse Boué, fbarousse@piensa.com

Debian-telepítés SATA-meghajtókon?

Létezik olyan Debian alapú terjesztés, amelyet SATA merevlemezre lehet telepíteni? Merevlemez-vezérlőm, egy MSI RAID Bus Controller, amelyhez vagy a YaST eszközfelismerője által ajánlott `ata_via` modult, vagy a VIA 8237-et használom az MSI KT6 Delta kézikönyvének megfelelően. Vissza akarok váltani Debianra, de a telepítő nem ismeri fel, hogy merevlemez van a rendszeremen. Ugyanez történik a Knoppix `knx-hdinstall` használatakor is. Azt értem, hogy a Linux-rendszert egy IDE-mevlemezere kell telepíteni, majd hozzá kell adni azokat a modulokat, amelyek a SATA használatához szükségesek, és az egészet át kell másolni az SATA-meghajtóra, azután futtatni kell a LILO-t, hogy a rendszer el tudjon indulni a SATA-meghajtón –, de ez egy kicsit bonyolult egy olyan lusta személy számára, mint amilyen én is vagyok.

A helyzet az, hogy a Debiant szinte bármire lehet telepíteni, ha nem a normál telepítőt, hanem a `debootstrap` csomagot használod. Azért akad hozzá néhány trükk és fogás is. Wiki-oldalaimon leírtam egy módszert, amivel a Debiant LVM alatt lévő meghajtókra lehet telepíteni, nem használva mást, csak egy LNX-BBC-t (➲ http://www.lnx-bbc.org), és az én saját hálózati kapcsolatomat. Szóval, ha van egy olyan mentőlemezed, mint az LNX-BBC, ami látja és eléri a SATA-merevlemezeket, akkor egyszerűen kövesd azt az eljárást, amit a ➲ http://www.starshine.org/sysadmin/DebootsraplnstallationDebootsraplnstallation címen, az oldalamon leírtam. Figyelmeztetlek, hogy nem lesz könnyű. Eléggé munkaigényes, és az én lépésről lépésre leírásom sem magyaráz túl sokat. Feltételezi, hogy szakértő vagy a lemezrészek felosztásában, a fájlrendszerek létrehozásában és befűzésében.

Jim Dennis, jimd@starshine.org

A Xandros Desktop OS (➲ http://www.xandros.com), a LindowsOS (➲ http://www.lindows.com) és a Libranet GNU/Linux (➲ http://www.libranet.com) mind Debian alapú, valamint karbantartott eszköz-együttműködési listájuk is van. A Serial ATA eszközének a gyártó webhelyén is utánanézhetsz.

Don Marti, dmarti@ssc.com

Új termékek

Roku HDTV SDK

A Roku, a HD1000 jelzésű nagyfelbontású digitális médialejátszó készítője, nemrég egy programfejlesztői készletet (SDK) adott ki, amellyel nagyfelbontású televíziókhoz lehet médiakezelő alkalmazásokat készíteni. A fejlesztők a Roku SDK segítségével C és C++ nyelven írhatnak programokat a Roku HD1000 készülékhez. Az SDK révén a tévés felhasználói felület elemei, a médiafolyamok, a hálózat és a memóriakártya egyaránt könnyen elérhetők, valamint az eszközvezérlés is megoldott. Emellett az egyedi telepítéseknél a Roku HD1000 az SDK nélkül, egyszerű ASCII parancsokkal is vezérelhető.
A HD1000 soros vagy ethernetkapun keresztül, vagy parancsfájlok futtatásával teljes körűen vezérelhető. Az SDK az alábbi címről tölthető le.
➲ http://www.rokulabs.com

LIPZ4 telefon

A Zultys Technologies LIPZ4 terméke szabadon elérhető szoftveres telefon, melynek segítségével hagyományos kézibeszélő nélkül, a számítógépen keresztül lehet telefonhívásokat indítani és fogadni. A nyílt szabványokra épülő LIPZ4 bármely Session Initiation Protocol (SIP) alapú IP telefonrendszerrel képes együttműködni. Négyféle hívásmegjelenítést támogat, képes az azonnali üzenetek kezelésére, valamint tartás, átadás, átirányítás, újrahívás és egyéb szolgáltatásokat kínál. A LIPZ4 az utolsó 32 bejövő vagy kimenő hívás számát tárolni is tudja. A Zultys a további szolgáltatásokat – például konferenciabeszélgetést, tömörítést (G.729), tartalék kiszolgálót – díjfizetés ellenében teszi elérhetővé. A program a hálózati biztonság növelésére a fejlett titkosítási szabványt (AES) használja.
A LIPZ4 a ➲ www.lipz4.com címről tölthető le.
➲ http://www.zultys.com

Sun Java Desktop – Talin hordozható gépen

A Tadpole Computers bejelentette, hogy legújabb hordozható gépeit, a Talin sorozat tagjait, Sun Java Desktop Systemmel szállítja. A Tadpole hordozható gépeire a felhasználók könnyen áttérhetnek, a gépek beépített biztonsági megoldásokat tartalmaznak, egyaránt alkalmasak a meglévő Unix, Java és Linux alapú alkalmazások

megtartására, valamint az irodai dokumentumokkal és háttérszolgáltatásokkal való együttműködésre. A Java Desktop System tartalmazza többek közt a Gnome, a StarOffice, a Mozilla, az Evolution, a Java 2 Platform programokat és a Linux operációs rendszert. A Talin 15 típus akár 3,0 GHz órajelű P4 processzorral, 128 MB – 1 GB SDRAM-mal, 15"-os SXGA+ kijelzővel, 3D grafikai támogatással és beépített Wi-Fi-csatolóval kapható. A közeljövőben további Talin modellek megjelenése is várható, köztük a 2 kg alatti tömegű 100X és egy 17"-os kijelzővel ellátott típus.
➲ http://www.tadpolecomputer.com

FogBUGZ 3.0 for UNIX

A FogBUGZ 3.0 webalapú programfejlesztési tervezetkezelőrendszer. Lényegében esetek adatbázisa, amely új szolgáltatások megvalósítására vonatkozó kérések, hagyományos hibajelentések és elektronikus levélben érkezett visszajelzések tárolására egyaránt használható. Minden kezelendő kérdés egy-egy személyhez rendelhető hozzá, az ő feladata, hogy a megoldást kidolgozza vagy a megfelelő személyt felkérje rá. Az esetek sorba rendezhetők, szerkeszthetők, becsülhetők, kereshetők stb., illetve leírást is lehet csatolni hozzájuk. Az esetek elektronikus levélben vagy webes felületen keresztül vihetők be. Kötelezően kitöltendő mezők nincsenek, és a hibajelentéseket bárki szerkesztheti. A FogBUGZ-esetekhez képernyőképeket, mintafájlokat és egyéb típusú dokumentumokat lehet csatolni, valamint Unicode támogatásának köszönhetően a hibák tetszőleges nyelven beírhatók. A FogBUGZ emellett olyan forráskódkezelő megoldásokkal is egybeépíthető, mint a VSS, a CVS, a Vault vagy a Perforce. A javítások és a hibajelzések között kétirányú összerendeléseket lehet létesíteni. A FogBUGZ 3.0 for UNIX futtatható Red Hat, SuSE, Mandrake, Debian és FreeBSD rendszereken.
➲ http://www.fogcreek.com/FogBUGZ

Escalade 8506-MI SATA-vezérlők

Az Escalade 8506 sorozatba tartozó többutas belső RAID-vezérlők egyesített csatlakozórendszerrel rendelkeznek, amely négy SATA kaput fog össze egyetlen csatlakozóba a vezérlőnél és a hátoldali aljzatnál egyaránt. A többutas vezérlők – PCI kártyánként – legfeljebb 12 SATA-meghajtót képesek vezérelni, így a meghajtók kapacitásától függően, egyetlen félhosszúságú kártyával akár 3 TB-os tárolóhely is kezelhető. Az Escalade StorSwitch kapcsolt felépítését alkalmazó 8506-MI a SATA végpontok között kapunként akár 1,5 GB/s sebesség elérését teszi lehetővé. Az MI-vezérlő RAID 0, 1, 10, 5 és JBOD-módban használható, továbbá megfelel a 64 bites, 66 MHz-es órajelet alkalmazó PCI-szabvány előírásainak.
➲ http://www.3ware.com

Linux Jounal 2004, 119. szám

Treo 600 Communicator

Akik valaha használtak már valamilyen zsebtitkárt, pontosan tudják mekkora gondot jelent, hogy a tenyérgépet (PDA) nem építették egybe a mobiltelefonnal, hiszen a tenyérgépeket összehangoljuk a számítógéppel, a mobiltelefont pedig a tenyérgéppel, vagyis elmondhatjuk ez a mindent mindenhez esete. Vegyünk egy egyszerű példát a saját életemből: mindig nagyon ügyelek rá, hogy mindenütt pontosan jelenjek meg. A számítógépemen egy idő-összehangoló (`ntpd`) fut, ami a legközelebbi atomórához igazítja a munkaállomásom óráját. A Palm-gépem óráját egy egyszerű perles programmal a számítógéphez igazítom, és mivel Siemens telefonomhoz – grafikus és karakteres felületen egyaránt – teljes értékű Linux-támogatás érhető el, ezért ezt is az asztali gépemmel hangolom össze. Ez csupán egy egyszerű időigazítási feladat volt, de ha a telefonszámokat, címeket, találkozókat vesszük tekintetbe valós hibalehetőséget rejt, hogy mit mihez, mikor és hogyan hangoljunk össze. E feladatra régóta elérhető megoldás, például a Nokia Communicator vagy az Ericsson P900-as telefon. Összevetettem a piacon elérhető összes eszköz jellemzőit, de amikor a leírásban elértem odáig, hogy az adott eszköz miként használható Linux alatt, igen elkeseredtem. Az Ercisson nagy tudású készülékei ugyan XML-t használnak, vagyis az adatokat ki lehet nyerni belőlük, de ez nagyon bonyolult foglalatosság. Nekem egy olyan GUI-ra lenne szükségem, mint a Jpilot, amellyel Palm-készülékemet hangolom össze a telefonommal (minden adatot nem csak a SIM-en tároltakat).

A készülék jellemzői

Tavaly novemberben a Handspring (ma már PalmOne) cég bejelentette a Treo 600-as készüléket, amely egy szabványos Palm tenyérgépet és egy GSM modult tartalmaz egyben. A készülék rendkívül megnyerő külleműre sikeredett és kiváló jellemzőkkel bír, de ezek ismertetésével most nem untatom olvasóinkat, hiszen majd' mindegyik tenyérgépekkel foglalkozó weboldalon megtalálhatóak, ugyanakkor megosztanám a rövid használat alatt felhalmozódott tapasztalataimat.

Várakozásaimnak megfelelően a Treo 600-asra egy mozdulattal fel lehetett tölteni az összes adatomat. Eddig egy Palm Tungsten T3-ast használtam, erről egy teljes mentést készítettem, ezt felmásoltam a Treóra, és ugyanonnan folytathattam a munkát, ahol megkezdtem. A Linux tekintetében semmilyen új meghajtót vagy grafikus programot nem igényelt, ugyanúgy a Jpilot és a PilotLink programmal lehet menteni, adatokat küldeni és fogadni. A készüléket úgy kell felfogni, mintha egy Zire71 gépet és egy GSM kártyát összegyúrtak volna, így sajnos a kijelző felbontása meg sem közelíti a Tungsten T3-asét, de sebaj majd a következő modellé. Rögtön az adatok áttöltése után beállítottam GSM szolgáltatóm GPRS net adatait (nagyjából 30 másodperc), és máris fenn voltam az interneten. SSH-kapcsolat kezdeményezése csak önsanyargatók számára ajánlott, ugyanis gyötrelmes feladat a kis képernyőn kínlódni vele, bár a lehetőség adott. A beépített böngésző szinte hibátlanul töltögeti be (meglepően gyorsan) a különböző oldalakat, bár érdemesebb a WAP böngészőt használni, ha a keresett oldal létezik wap formátumban is, hiszen jóval gyorsabb és kisebb adatmozgással jár. Az SMS-eket kezelő program képes a szálkövetésre (Thread), ha például valakivel egymás után több SMS-t váltunk, szálszerűen (egymás alá) rendezi őket. A beépített kamera sajnos igen gyengére sikeredett, legfeljebb MMS küldözgetésére való, de van benne, és ha elég fényt kap a célobjektum, akkor 640×480-as felbontású képet tud róla készíteni. A Treo, amíg nálam tartózkodott jelentősen megkönnyítette az életemet, hiszen mindent az egyben (GSM+Palm) készülék. A legjobb tulajdonsága talán az volt, hogy a péntek esti sörözések alkalmával, amíg a barátaimra vártam, Assimov „Alapítvány és Birodalom" című művét olvasgattam rajta. Mindezek ellenére nagyon zavart, amikor a Palm rendszer „dobott egy hátast" és újraindult, minden egyes alkalommal a telefon is kikapcsolt, és újra és újra kézzel kellett visszakapcsolni.

Összegzés

Mindent egybevetve a készülék hibái ellenére is rendkívül rokonszenves. Remélhetőleg, hamar kinövi gyermekbetegségeit (a Palm gyorsan és megbízhatóan frissíti saját gépeinek programjait), azonban a legnagyobb hiányosságán sajnos csak a vastag pénztárca segíthet, ugyanis itthon megközelítőleg 230 000 forintba kerül.

Varga S. Csaba

Nehogy elfeledjük, miért győz a nyílt forrás

Az ügyfelek tapasztalata azt mutatja, hogy a nyílt forrás a legjobb biztosítás, amit egy cég informatikai osztálya köthet.

Rövid ideig a VA Research volt a legjobb hely, ahol eddig dolgoztam. Kevéssel azután, hogy munkába álltam, ***Larry Augustin***, az akkori ügyvezető azzal állt elő, hogy többek között nekem kellene ápolnom a cég kapcsolatait a linuxos közösséggel. A munkát azzal kezdtük, hogy régi gépeket gyűjtöttünk be, és az egyes fejlesztői csoportoknak hozzáférést adtunk.

Legyen szó a Free Software Foundationről, a Debianról vagy a Stampede-ről, azt akartuk, hogy a felhasználók tudják, létezik egy hely, ahová fordulhatnak, ha projektjük sávszélesség-igénye meghaladja az internetszolgáltatójuk által kínált kereteket. Ez mindaddig egészen jól működött, amíg a projekteknek otthont adó gépek száma el nem érte a negyven körülit, ez azonban már jelentős többletterhet rótt rendszergazdáinkra. Etájt történt, hogy ***Tony Guntharp***, ***Tim Perdue***, ***Uriah Welcome*** és mások bezárkóztak egy szobába néhány raklap SCSI merevlemez társaságában, és 68 nappal később bemutatták a világnak a ***SourceForge.net***-et.

Tony és kollégái az év végéig körülbelül ezer érdeklődőre számítottak, de már az első hónap után több mint ötezer felhasználójuk akadt. A növekedés mértéke a későbbiek során sem csökkent. A vezetés egyre nagyobb nyomást gyakorolt rájuk és a magas költségekre hivatkozva be akarták záratni a webhelyet. A webhely és a VA szerencséjére – a Larry Augustinhoz és a ***Steve Westmoreland***-hoz hasonló emberek munkájának köszönhetően –, ez a törekvés háttérbe szorult. A SourceForge webhely kódját, a webhelyen található többi kezdeményezéshez hasonlóan, GPL alatt fejlesztették. Nem sokkal ezután a VA új igazgatót kapott, és kivonult a számítógépes üzletből. A kód GPL-es kiadásai hirtelen megszűntek és a SourceForge programjának telepítése lett a cég fő tevékenységi köre. Ezzel a változással egyidejűleg a VA lezárta a forrást.

A GPL felhasználói szerződésből fakadóan azonban az eredeti kiadás elérhető maradt. Körülbelül három hónap elteltével Tim Perdue, az eredeti csapat egyik tagja nagy munkába fogott, és kiadta a ***GForge***-ot, egy új változatot, azok számára, akik frissíteni vagy karbantartani akarták saját vagy ügyfeleik SourceForge-szerű webhelyeit. Ahhoz, hogy teljes képet alkothassunk, meg kell említenem, hogy a Tony Gunthrap, Steve Westmoreland és jómagam által alapított Konstrux Technologies cég szintén ilyen webhelyek telepítésével és karbantartásával foglalkozik.

Cégünket a GForge-ra alapoztuk. Tim Perdue és mások ugyancsak hasonló vállalkozásokba kezdtek.

Amióta a SourceForge és a GForge forrása elvált, a GForge-hoz egy mindenre kitérő rendszerfelügyeleti és telepítési leírást adtunk hozzá, sokat tisztítottunk a kódon és a felhasználói felületen, megvalósítottuk az XML-csatolófelületet, továbbá telepítőprogram is készült hozzá.

Akár az ügyfél, akár a szolgáltató nézőpontjáról legyen is szó, túl gyakran feledkezünk meg a nyílt forrású programok előnyeiről. Az egyik ügyfelünk szerepkörökön alapuló azonosítási rendszert szeretett volna fejlesztetni a GForge-hoz. Miközben a bővítés részleteit rögzítő szerződést írtuk, egy programozó (a Mitre-től) hasonló rendszert dolgozott ki és beküldött egy javítást is, amely az elvégzendő munka kilencven százalékát lefedte. Ilyen módon újabb alkalmazottak kényszerű foglalkoztatása, illetve túl kicsi árrés alkalmazása vagy túl magas ár kiszabása nélkül is teljesíteni tudtuk ügyfelünk elvárásait. Lehetőségünk nyílt versenyre kelni nálunk sokkal nagyobb, zárt forrással dolgozó cégekkel. Abban a valószínűtlen esetben, ha a Konstrux csődbe menne, az ügyfeleknek nem kell tartaniuk tőle, hogy a csoportmunkát támogató webhelyük karbantarthatatlanná lesz. Ezt fent említett versenytársainkról nem lehet elmondani. Amennyiben ők mennének csődbe, a programjuk forrását a felszámolási eljárás keretében elárverezik és ügyfeleik könnyen pórul járhatnak. A nyílt forrás segítségével az ügyfelek megvédhetik informatikai befektetéseiket. Ez magától értetődő azok számára, akik régen foglalkoznak a nyílt forrással, de az emberek többségét mai napig emlékeztetni kell rá.

A Free Software Foudation az egyén szabadságjogairól beszél, de az eszményük jól kiterjeszthető az üzleti életre is, hiszen a cégek számára is biztosítja a szabadságot, hogy saját informatikai szakértőket alkalmazzanak, és az üzleti rendszereiket megőrizzék anélkül, hogy a barátságtalan program-viszonteladók vagy a bíróság túszaivá válnának.

Chris DiBona
(➲ http://dibona.com) a Konstrux Technologies társalapítója, a Linux Journal 2000 Book of the Year és az Open Sources társszerkesztője, valamint a Slashdot.org webhely egyik szerkesztője.

A hét vezér: biztonsági eszközök

Attól, hogy újabb és újabb programokkal tömjük meg, a gépünket nem lesz nagyobb biztonságban. Az ajánlott hét csomag segítségével megtanulhatjuk, hogyan állíthatunk össze, például biztonsági házirendet.

Linux alá számtalan kiváló, ingyenes és nyílt forrású biztonsági eszköz érhető el; akár minden hónapban szólhatna valamelyikükről egy cikk. Általában egy vagy két eszközzel szoktam mélyrehatóbban foglalkozni, most azonban inkább áttekintő jelleggel szeretném megismertetni kedvenc linuxos biztonsági eszközeimet. Aki csak most ismerkedik a Linuxszal vagy a hálózati biztonság témakörével, talán először találkozik e programcsomagokkal. Sebaj, reményeim szerint sikerül megtalálniuk azt az irányvonalat, ami mentén haladva elmélyültebb tudásra tehetnek szert. Aki egy vagy több eszközt ismer, de nem mindegyiket, ez alkalommal bővítheti ismereteit és eszközkészletét. Biztosan akadnak olyanok is, akiknek az égvilágon semmi újat nem fogok mondani – ők legfeljebb kikapcsolódnak egy kicsit, miközben azt is megtudják, hogy a saját hordozható gépem processzora miféle dolgokon gondolkodott a legtöbbet az elmúlt időszakban.

Netfilter, illetve iptables

Körsétánkat a legalapvetőbb programmal, a Netfilterrel kezdjük, a Linux-rendszermag beépített tűzfalával. Pontosítsunk egy kicsit: a kérdéses modulok összességének hivatalos neve a Netfilter, az `iptables` pedig a felhasználói térben érvényes parancs, amit a Netfilter magmodulok beállításainak megadására használunk. A két nevet nyugodtan összekeverhetjük, különösebb félreértést nem fogunk okozni vele, kivéve, ha `iptables` parancsokat adunk ki vagy a rendszermag fejlesztőivel társalgunk.

A Netfilter tavaly megnyerte Szerkesztőségi díjunkat a „Legjobb biztonsági eszközök” kategóriában. Azóta is tartom a véleményemet, mely szerint a Netfilter fontos szerepet játszott abban, hogy a linuxos tűzfalak végre túllépjenek az állapotmentes csomagszűrés (stateless packet filtering) némileg meghaladott módszerén, és belépjenek az állapotalapú csomagszűrés korába. Mit is jelent mindez a biztonsággal napi 24 óránál kevesebbet foglalkozók számára? A Netfilter segítségével a linuxos tűzfalak egymással fennálló kapcsolataikat is figyelembe véve vizsgálhatják a hálózati csomagokat, vagyis a létrejött kapcsolatokkal társítják őket; például adott csomagot egy átvitel kezdeteként jelölnek meg. A 2.4-esnél régebbi Linux-rendszermagok a csomagokat különálló egységekként kezelték, a szűrést pedig kizárólag a csomagok forrása, valamint célja alapján végezték.

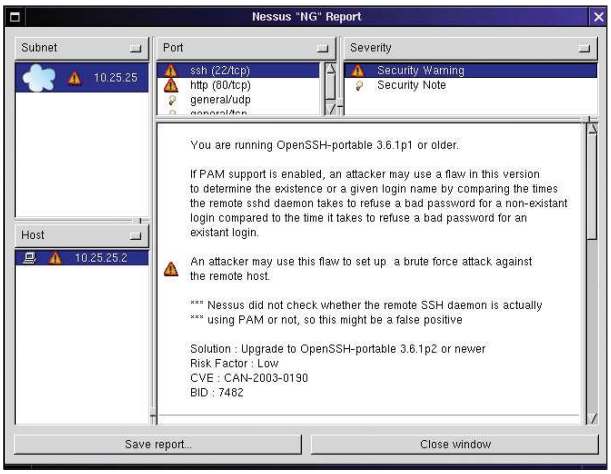
Egy HTTP-kapcsolat csomagjait például a tűzfalak egyenként vizsgálták és szűrték, nem egymáshoz tartozó elemekként kezelték őket.

Az újfajta csomagszűrési eljárás nem csak arra alkalmas, hogy linuxos tűzfalakat állítsunk össze, a Netfilter helyi biztonsági eszközként, kiszolgálókon és munkaállomásokon ugyanolyan jó szolgálatot tehet, mint a hálózati tűzfalakon. (Arról, hogy mindezt miképpen cselekszik a Linuxvilág 2002. decemberi számának 55. oldalán „Biztonság a helyi hálózaton: IP Tables” címmel írtam részletesebben.)

Az `iptables` parancs használata nem nagy ördöngösség, csak egy kis időt kell szánnunk az `iptables(8)` súgóoldal tanulmányozására. Ezenkívül e témában ***Robert Ziegler*** kiváló „Linux Firewalls”című könyvét tudom ajánlani. Az `iptables` kiemelkedő parancsfájl-készítési lehetőségeket kínál a már említett forrásokban, és az interneten rengeteg példát lehet találni, amelyeket könnyedén saját igényeinkhez igazíthatunk.

Mit ajánlhatunk azoknak, akik nem érdeklődnek a csomagszűrés rejtelmei iránt, és inkább ékes angol nyelven beszélő grafikus felületen keresztül szeretnének dolgozni? Semmi gond, a Netfilterhez számos külső fejlesztésű felület létezik. Az egyik legjobb a Firewall Builder (➲ http://www.fwbuilder.org), ami varázslókkal és újrafelhasználható objektumokkal segít bennünket a tűzfalszabályok létrehozásában. A Firewall Builderrel a Linuxvilág hasábjain egy kétrészes sorozatban foglalkoztam 2003. májusában és júniusában.

Másik népszerű `iptables` kiegészítő a Mason, amely az `iptables` parancsfájlokat önműködően, a rendszer mindennapi használatát figyelemmel kísérve készíti el. Elsősorban egyedi, munkaállomásokat védő tűzfalszabályok létrehozására használják. A Mason a ➲ http://users.dhp.com/~whisper/mason címről tölthető le. Szintén érdekes és elterjedt eszköz a Shorewall, amely a ***/etc/shorewall*** könyvtárba helyezett egyszerű szöveges fájlok alapján állítja be az összes `iptables` parancsfájlt. Honlapja a ➲ http://shorewall.net címen található.



1. kép Nessus-példajelentés, amely sebezhető SSH-démon jelenlétéről tudósít

Több Linux-változat saját, egyedi eszközt kínál az `iptables` használatának segítésére. A SuSE 8.2-esben ez a SuSEfirewall2, amely a */etc/sysconfig/SuSEfirewall2* fájlban megadott egyszerű beállítások alapján önműködően hozza létre és futtatja az `iptables` parancsokat. Ha saját terjesztésünk rendelkezik ilyen programmal, akkor érdemes kipróbálni. Valószínűleg már fent csücsül a gépünkön, így a telepítéssel sem lesz gondunk.

Bastille

A Bastille, ***Jay Beale*** és ***Jon Lasser*** alkotása, önmagában is fogalom, külön kategóriát alkot. Lényegében egy parancsfájl, amely a használójának feltett kérdések alapján átfogó jelleggel képes védeni a linuxos rendszert. A többi védelmi parancsfájltól élesen megkülönbözteti, hogy minden kérdéséhez bőséges magyarázat tartozik. Soha nem láttam még olyan biztonsági programot, amely olyan sokat tenne a felhasználók oktatásáért, mint a Bastille. Újoncoknak éppen ezért ajánlom, hogy elsőként mindenképpen a Bastille-t ismerjék meg.
Amikor pár éve a Bastille-ról írtam („Hogyan erősítsük rendszerünk biztonságát?”, Linuxvilág, 2001. április), néhány kérdésemmel megkerestem Jay Beale-t. A Bastille nagyszerűsége jelentős részben Jay nyitott személyiségéből fakad, közvetlen és szórakoztató stílusban segíti hozzá programjának használóit rendszerük biztonságának javításához.
A Bastille hivatalosan Red Hat, Mandrake és Debian GNU/Linux alatt használható. Átültetése HP-UX és Mac OS X alá is elérhető; és a ➲ http://www.bastille-linux.org címről szerezhető be.

Nmap

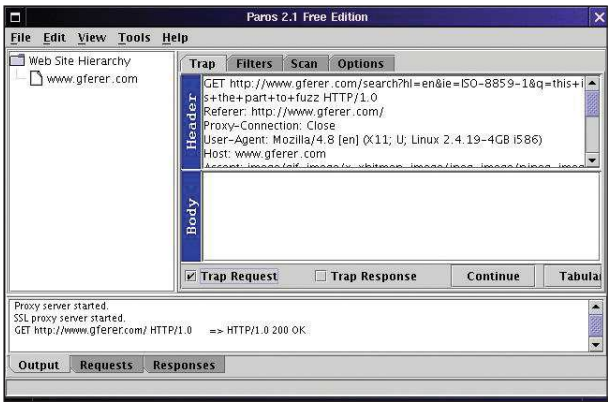
A Netfilter és a Bastille kizárólag védekezésre szolgálnak. Mivel próbálhatjuk ki, hogy valóban biztonságban van-e linuxos gépünk? Az egyik lehetőség, hogy egy kapupásztázót (port scanner) futtatunk le, összesítjük mely kapuk állnak nyitva, és ennek alapján határozzuk meg, mely hálózati alkalmazások futnak.
Ha egy egész telephely vagy iroda biztonságát szeretnénk felmérni, akkor csupán önműködő kapupásztázással nem tudjuk megállapítani, hogy biztonsági szempontból mennyire végzett gondos munkát a vele megbízott személy. A pásztázás mégsem haszontalan, ha mást nem is, legalább a tűzfalak beállításait ellenőrizni tudjuk. Egy jó kapupásztázóval pontosan meghatározhatjuk, hogy a kívülről érkező betyárok (hacker) milyen bejutási pontokat láthatnának rendszerünkön.
Az Nmap (lásd 1. széljegyzetünket az 59. CD-melléklet Magazin/biztonság könyvtárában) kétségbevonhatatlanul a kapupásztázók királya: gyors, kis méretű, ingyenes és nagy tudású. Az Nmap többféle pásztázási módszert ismer, a gyors és feltűnő TCP-csatlakozási próbától egészen a csendes és aljas karácsonyfa-pásztázásig. Az Nmap – bár parancssorból sem nehéz használni – grafikus felülettel is bír. Legújabb változata a ➲ http://www.insecure.org címről tölthető le, de valószínűleg saját Linux-változatunk is tartalmazza valamely változatát. Nem kell mást tennünk, mint a megfelelő korongot elővenni, és pillanatok alatt feltehetjük gépünkre az Nmapot.

Nessus

A kapupásztázók csak végignézik a nyitott kapukat, a biztonsági pásztázók viszont megpróbálnak csatlakozni is hozzájuk, és a lehető legtöbb adatot kísérlik meg beszerezni a kapuk mögött várakozó alkalmazásokról. Ennek legegyszerűbb változata a hirdetménygyűjtés, amikor az alkalmazások által a sikeres kapcsolódást követően megjelenített üzeneteket naplózzuk. Számos alkalmazás névvel, és nem egy esetben, változatszámmal azonosítja magát.
Természetesen a magasabb szintű biztonsági pásztázók jóval többet tesznek a hirdetmények begyűjtésénél. Miután meghatározták, hogy az adott kaput milyen alkalmazás figyeli, és az ismert sebezhetőségeit megpróbálják kihasználni, általában olyan módon, hogy a közismert betörési módszereket kezdik alkalmazni – természetesen nem viszik végig őket. A Nessus (*1. kép*) egy profi, kiválóan testreszabható, ennek ellenére ingyenes biztonsági pásztázó.
Az Nmaphez hasonlóan, a Nessus is felbecsülhetetlen értékű eszköz a biztonsági mérnökök számára. A két program haszna a hétköznapok során is megmutatkozik. Például, ha például a Nessus segítségével egy hobbi-webkiszolgálót akarunk ellenőrizni. A Bastille-hoz hasonlóan a Nessus tervezése során is a célkitűzések között szerepelt a felhasználók oktatása. Ha a jelentéseket gondosan átolvassuk, akkor nemcsak a sebezhetőségekről, de javítási módjaikról is egy csomó dolgot megtudhatunk.

Paros

Mielőtt a biztonsági ellenőrzések témakörén továbblépnénk, szenteljünk némi figyelmet a webes alkalmazások biztonságának is. A webes alkalmazások az interneten keresztül elérhető szolgáltatások bővülésében és a kívülről kihasználható sebezhetőségek számának növekedésében egyaránt elsődleges szerepet játszanak. Vajon hogyan tudnánk ellenőrizni webes alkalmazásaink biztonságosságát? Jó gondolat, hogy a Nessus kezdésnek megfelel, de inkább az általános webdémonok ellenőrizhetők vele. A Nessus jelentései inkább a kiszolgálóra, például az Apache démonra, és nem az általa szolgáltatott webes tartalomra vonatkoznak. A Nessusszal sosem tudjuk megállapítani, hogy vajon saját webes alkalmazásaink kellően ellenőrzik-e a bemeneteket, nem sebezhetők-e a webhelyek között futó parancsfájlokkal



2. kép A Paros ingyenes eszköz a webes alkalmazások ellenőrzésére

végrehajtott támadásokkal szemben (cross-site scripting), esetleg az általuk várt átadott értékek módosításával nem lehet-e megbolondítani őket, és így tovább. A Paros pontosan az ilyen kérdésekre segít megtalálni a választ.

A Paros (*2. kép*) egy ingyenes, Java nyelven írt eszköz, amely Clarified Artistic License (letisztult művészi felhasználói szerződés) hatálya alatt jelenik meg. A Paros használatához telepíteni szükséges a Java futtatási környezetet. A Paros futtatható JAR állománya és teljes forráskódja a ➲ http://www.proofsecure.com címről tölthető le.

A Paros a webes biztonsági eszközök új nemzedékének működési elvét követi. Az ellenőrzésre használt munkaállomáson helyi proxyként fut, a helyi böngésző és a célwebkiszolgáló közötti párbeszédeket bonyolítja le. Ezzel a módszerrel a kimenő kérések a kiszolgálónak való továbbküldés előtt elfoghatók, módosíthatók és összezavarhatók. Tegyük fel, hogy a webes alkalmazásunk lenyíló menüket tartalmazó űrlapot alkalmaz, és meg akarunk győződni arról, hogy alkalmazásunk megfelelően ellenőrzi-e a bemenetet. A Paros segítségével a lenyíló menük pontjait véletlenszerű karakterláncokra cserélhetjük le – így lesz a hétfőből blablabla –, majd megvizsgálhatjuk, hogyan válaszol az alkalmazásunk.

A Paros sokféle pásztázási módszert támogat, például a könyvtárak végigjárását is el tudja végezni. A JRE futtatása a régebbi rendszereket ugyan erősen leterhelheti, de maga a Paros rugalmas és felhasználóbarát program. Mivel Java alapú, nemcsak Linux alatt futtatható, jómagam is többféle Windows-változaton használtam már.

A Paros nem az egyetlen létező összezavaró proxy, ***Dave Aitel*** kiváló Spike nevű proxyprogramját is érdemes megemlíteni. Szintén helyi proxyként működik, de miután a köztünk és a távoli webhely között zajló párbeszédet megfigyelte, önműködő összezavarásos támadások indítására is képes. A Spike Python nyelven készült, vagyis kisebb számítási teljesítményt és kevesebb memóriát igényel, mint a Paros.

F.I.R.E.

Rövidke áttekintésemet egy oknyomozó programmal, ***William Salusky*** F.I.R.E. (Forensics and Incident Response Environment, azaz törvényszéki és esetkezelő környezet) alkalmazásával szeretném zárni. Bármennyire is szeretnénk elhessegetni a gondolatot, mégis tudomásul kell vennünk: hiába vagyunk körültekintőek, hiába próbálunk előre gondolkodni, mégis előfordulhat, hogy egy nem túl szép napon rendszerünket megtörik. Ha tudni szeretnénk a betörés hogyan és miért történt, a F.I.R.E. lehet segítségünkre. A F.I.R.E. egy CD-s Linux-változat, amit feltört rendszerek elemzésére és adataik helyreállítására állítottak össze.

A gépet szükség szerint a F.I.R.E. CD-jéről is indíthatjuk, de ha a rendszerünk még működőképes, azt is megtehetjük, hogy a lemezt befűzzük, és a szükséges programokat róla futtatjuk. Az utóbbi főleg akkor lehet előnyös, ha a gépen található futtatható állományokban nem bízunk meg, mert úgy gondoljuk, hogy a betörő támadócsomagot (rootkit) vagy trójaival fertőzött változatokat telepített fel. A F.I.R.E. a szükséges elemzések elvégzése mellett arra is használható, hogy segítségével a megtört gép adatait más hálózati állomásra másoljuk át. A F.I.R.E. az X Window Systemet is tartalmazza, valamint számos parancssori és X alapú biztonsági programot is magában foglal, többek közt az Nmapet és a Nessust. A F.I.R.E. segítségével akár egy egyszerű windowsos hordozható gépből is félelmetes betörésérzékelő hadigépezetet varázsolhatunk. A F.I.R.E. főbb szolgáltatásai a menürendszerből is elérhetők, ilyen módon még a betörések felderítésében kevésbé jártasak számára is könnyen használható. A F.I.R.E. program a ➲ http://fire.dmzs.com címről tölthető le.

Összegzés

Az volt a szándékom, hogy a teljesség igénye nélkül megemlítsek néhányat a számos kiváló linuxos biztonsági eszköz közül. Az írásban szereplők listája jóval rövidebb, mint a kimaradottaké: Tripwire, AIDE, Nikto, GnuPG, FreeS/WAN, Snort, PSAD, Stunnel, OpenSSL és még sokan mások. Mindössze annyit sikerült elérnem, hogy kedvenc munkaeszközeimet bemutattam. Remélem, mindenki hasznosnak találja majd őket. Ne feledjük, az említett eszközök jókora hatalmat adnak használójuk kezébe, ezért alkalmazásuk során felelősséggel, gondosan és erkölcsösen kell eljárni. Aki így cselekszik, annak jó szórakozást kívánok!

Linux Journal 2004. február 118. szám

KAPCSOLÓDÓ CÍMEK

- ➲ http://www.fwbuilder.org
- ➲ http://shorewall.net
- ➲ http://www.bastille-linux.org
- ➲ http://www.insecure.org
- ➲ http://www.proofsecure.com
- ➲ http://fire.dmzs.com

Mick Bauer (mick@visi.com)
Biztonsági szakember, a Linux Journal biztonsági témákkal foglalkozó szerkesztője, biztonsági tanácsadó a Minnesota állambeli Minneapolisban található Upstream Solutions LLC Inc.-nél.

Alkalmazásszintű proxyzás a Zorppal (1. rész)

A különféle támadási formák ellen az alkalmazásszintű proxyk nyújtják a legjobb védelmet, viszont bonyolultabbak is a csomagszűrőknél.

Első ránézésre úgy tűnik, hogy az állapotalapú csomagszűrés teljesen uralma alá vonta a tűzfalak világát, nemcsak a piaci részesedés, de gondolkodásmódunk tekintetében is. Az állapotalapú csomagszűrést végző termékek listája meglehetősen hosszúra nyúlik, a kereskedelmi fejlesztésű Check Point Firewall-1 és a Linux kiváló Netfilter magkódja egyaránt ide sorolható.
De vajon mi a helyzet az alkalmazásszintű proxykkal? A tűzfalakkal foglalkozó mérnökök régóta bizonygatják, hogy semmi sem képes olyan sokféle hálózati támadást megállítani, mint az alkalmazásokat pontosan ismerő proxy. ***Scheidler Balázs***, a ma már nélkülözhetetlennek számító naplózóprogram, a `syslog-ng` készítője, megalkotta a Zorpot, ami egy nyílt forrású, egész egyszerűen zseniális proxytűzfal. Ebben a hónapban azt szeretném elmesélni, hogy az alkalmazásszintű proxytűzfalak kérdésében micsoda megváltásnak éreztem a Zorp megjelenését, és e program hogyan lehet mindazok segítségére, akiknek fontos és érzékeny adatokat kezelő hálózatok biztonsága felett kell őrködniük.

Frissítsük fel emlékezetünket

Talán többen is feltették már a kérdést, mi a csuda az az alkalmazásszintű proxyzás és az állapotalapú vizsgálat? Miért jobb az egyik vagy a másik?
A tűzfal egy számítógép vagy egy beágyazott programot futtató eszköz, amely képes a hálózatok egymástól való elkülönítésére és a közöttük folyó forgalom szabályozására. Azokat a szabályokat, amelyek megmondják, mely hálózati csomópontok milyen típusú hálózati csomagokat és hová továbbíthatnak, tűzfalszabályoknak, összességüket pedig tűzfal-házirendnek nevezzük.
Ezek a szabályok különböztetik meg a tűzfalakat a normál útválasztóktól. Az útválasztókon a csomagok különféle hálózatok közötti mozgatására vonatkozó szabályokat nyilván meg kell adni, de azt már nem feltétlenül kell előírni, hogy milyen forgalmat hova szabad továbbítani. Ezzel szemben a tűzfal képes a kifinomult válogatásra.
A csomagok szétválogatásának egyik legegyszerűbb szempontja az internetprotokoll (IP) fejrészének tartalma. Az IP-fejrész alapszintű adatokat tartalmaz, a legfontosabbak a protokolltípus, a forrás és a célállomás címe, illetve – ha van ilyen – a forrás- és a célkapu. A kapuszámokat valójában a csomag következő, az UDP vagy a TCP protokollhoz tartozó fejrésze tartalmazza. Azokat a tűzfalakat, amelyek csak az alapszintű adatokat vizsgálják, egyszerű csomagszűrőknek nevezzük. Mivel az egyszerű csomagszűrők nem néznek bele a csomagok mélyébe, működésük nagyon gyors. Ugyanakkor az IP-fejrész és a TCP- vagy az UDP-kapuszám semmit nem árul el a csomag más csomagokkal fennálló viszonyáról. Ha például megvizsgáljuk egy HTTP-csomag IP-fejrészét, akkor az IP-fejrészbeli protokollmezőnek köszönhetően tudni fogjuk, hogy TCP-csomagról van-e szó, a forrás- és a cél-IP-címéből meg tudjuk állapítani, honnan érkezik és hova küldték, a célkapu száma alapján pedig el tudjuk dönteni, milyen típusú alkalmazás küldte. Az *1. táblázat* egy egyszerű csomagszűrő egyik szabályát szemlélteti. Ilyen szintű vizsgálatnál azonban fontos, hogy a HTTP-kapcsolattal összefüggő adatokat figyelmen kívül hagyjuk. Nem nézzük, hogy a csomag új HTTP-kapcsolatot hoz-e létre vagy már meglévő kapcsolat csomagja, esetleg véletlenszerűen, adott esetben rossz szándékkal elküldött, más csomagokkal semmilyen viszonyban nem lévő csomagról van-e szó. Annak oka, hogy mindezt figyelmen kívül hagyjuk, hogy a kapcsolatra vonatkozó adatok, mint a TCP-zászlók (TCP flags), a TCP-sorozatszámok (TCP sequence numbers) és az alkalmazásszintű parancsok a csomag belsőbb részeiben helyezkednek el, ahova a csomagszűrő már nem lát be. Itt lép a képbe az állapotalapú csomagszűrés. Az állapotalapú csomagszűrők, ahogy az egyszerű csomagszűrők is, az egyes csomagok forrás- és cél-IP-címének, illetve forrás- és célkapujának vizsgálatával kezdik munkájukat. Ennél azonban mélyebbre is beásnak a csomagba, és annak UDP- vagy TCP-fejrésze alapján megvizsgálják, hogy a csomag tartalma egy új kapcsolat létrehozására irányul-e. Ha igen, a tűzfal létrehoz egy új bejegyzést az állapottáblában. Ha nem, akkor megvizsgálja az állapottábla tartalmát, és megállapítja, hogy a csomag már fennálló kapcsolathoz tartozik-e. Az állapotalapú csomagszűrő képes azoknak a csomagoknak a kiszűrésére, amelyek valótlanul állítják magukról, hogy meglévő kapcsolathoz tartoznak.
Itt kell megemlítenünk, igaz ugyan, hogy az UDP kapcsolat nélküli protokoll, ám egy jó állapotalapú tűzfal tudja, hogy például egy az 53-as UDP-kapura küldött, kimenő DNS-kérés hatására érkezni fog egy válasz a kiszolgáló 53-as kapujáról. Az állapotalapú csomagszűrés két fontos előnnyel bír az egyszerű csomagszűréshez képest.

1. táblázat *Egyszerű csomagszűrő szabályok a HTTP-forgalomhoz*

Forrás-IP (Source IP)	Cél-IP (Destination IP)	Protokoll	Forráskapu (Source Port)	Célkapu (Destination Port)	Művelet
Bármely	192.168.1.1	TCP	Bármely	80	Engedélyezés
192.168.1.1	Bármely	TCP	80	Bármely	Engedélyezés

2. táblázat *Állapotalapú csomagszűrő szabály a HTTP-forgalomhoz*

Forrás-IP	Cél-IP	Protokoll	Forráskapu	Célkapu	Állapot	Művelet
Bármely	192.168.1.1	TCP	Bármely	80	Új	Engedélyezés

Először is a tűzfalszabályok egyszerűbbek lehetnek. Nincs szükség rá, hogy minden egyes kétirányú – például HTTP – kapcsolat mindkét irányát pontosan leírjuk, a tűzfalszabályokban csak a megengedett kapcsolatok kezdeményező pontját kell megadnunk. A megengedett és már létrehozott kapcsolatokhoz tartozó további csomagok kezelése a tűzfal állapottáblája alapján történik, az általunk megadott tűzfalszabályoktól függetlenül. A 2. *táblázat* alapján látható, hogy egyetlen szabállyal is engedélyezni tudtuk ugyanazt a HTTP-kapcsolatot, amelynek leírásához az 1. táblázatban még két szabályra volt szükségünk.

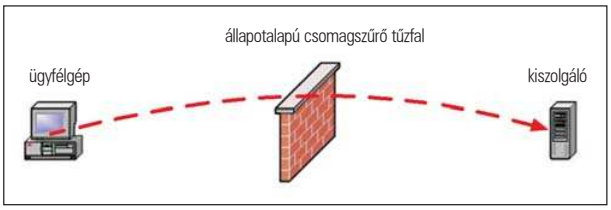
Az állapotalapú csomagszűrés második fontos előnye, hogy nem kényszerülünk olyan, a nem állapotalapú megoldásoknál időnként szükséges lépésekre, mint az 1024-nél magasabb célkapuszámmal rendelkező, bejövő TCP- és UDP-csomagok beengedésére az internetről a belső hálózatra. Ilyesmire akkor lehet szükség, ha nincs más lehetőségünk a megengedett kapcsolatok és a csomagok összerendelésére. Nem szaporítom tovább a szót, a lényeg az, hogy az állapotalapú csomagszűrés jobb védelmet nyújt, mint az egyszerű csomagszűrés.

„Nagyszerű – állapíthatjuk meg – az állapotalapú csomagszűrők tehát hatékonyabbak és biztonságosak!" – valóban így van. De vajon vannak-e olyan dolgok, amelyeket az állapotalapú csomagszűrők sem vesznek figyelembe? Mi történik például a hibásan megformált HTTP-kérésekkel és a szándékosan átlapolt IP-töredékekkel (overlapping IP fragments)? Létezik olyan tűzfal, amely minden egyes csomagot teljes egészében képes megvizsgálni, vagy valamilyen más módon tud segíteni abban, hogy a lehető legkevesebb ártó szándékú csomag érje el a hálózatunkat? Igen, létezik ilyen, ezt nevezzük alkalmazásszintű proxynak, más szóval alkalmazásrétegbeli átjárónak. Míg a csomagszűrők – egyaránt állapotalapúak és egyszerűek – minden csomagot megvizsgálnak, majd az engedélyezetteket továbbítják, addig az alkalmazásszintű proxyk minden létrehozandó kapcsolatot két részre osztanak, önmagukat pedig az összeköttetés egyenrangú résztvevőjeként iktatják be középre. Az ügyfél, vagyis a kapcsolat kezdeményezője számára a tűzfal kiszolgálóként látszik, míg a megjelölt cél, vagyis a tényleges kiszolgáló számára ügyfélként jelenik meg. Az *1.* és a *2. ábra* a két megoldás közötti különbséget szemlélteti. Az 1. ábrán egy állapotalapú csomagszűrőt láthatunk, amely „csupán" engedélyezi és tiltja a kapcsolatokat. Amolyan megfigyelőként foghatjuk fel, amely az engedélyezett csomagokat többé-kevésbé változatlan formában továbbítja – kivéve azt az esetet, ha például hálózati címfordítást (Network Address Translation, NAT) használunk. Ezzel szemben a 2. ábrán azt láthatjuk, hogy a tűzfal a megengedett kapcsolatok végpontjaként önmagát jelöli meg, majd új, proxyzott kapcsolatot kezdeményez az eredetiek célállomásaival.

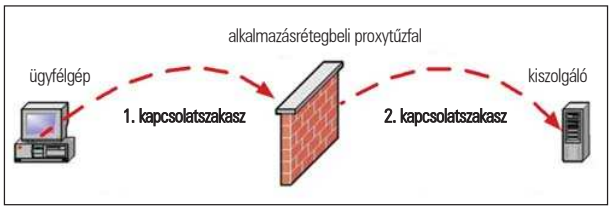
A proxyzás kétféle típusú lehet: átlátszó (transparent) és nem átlátszó. Átlátszó proxy használatakor egyik fél sem veszi észre, hogy kapcsolatuk valójában proxyn keresztül jött létre. Az ügyfélrendszer úgy címezi meg a csomagjait, mintha nem is lenne tűzfal a hálózatban, vagyis a valódi célállomás IP-címét adja meg. A nem átlátszó proxynál viszont az ügyfélnek a saját csomagjait a tűzfalnak szükséges címeznie, és nem a valódi célállomásnak. Mivel az ügyfélnek ilyenkor valamilyen módon közölnie kell a tűzfallal, hogy valójában kivel szeretne kapcsolatba lépni, nem átlátszó proxy használatakor proxyképes alkalmazásokat (proxy-aware applications) kell futtatnunk. A legtöbb webböngésző és FTP-ügyfél ugyan képes a nem átlátszó proxyk használatára, egy átlátszó proxy azonban kevesebb bosszúságot okoz a felhasználóknak. A korszerű alkalmazásszintű proxyk, mint a Zorp is, átlátszók.

Átlátszó vagy sem, a proxyzás számos előnnyel kecsegtet. Az egyik ilyen, hogy az alacsonyabb szinten hibás csomagokat, például a fejrészükben rendellenes jelzőkkel ellátott IP-csomagokat a tűzfal nem küldi tovább. A tűzfal a másodlagos kapcsolatot olyan módon hozza létre, amit ő maga tart helyesnek, és nem az ügyfélgép. Másik előnye abból fakad, hogy a tűzfal teljes egészében új kapcsolatokat hoz létre, nem csupán továbbküldi vagy átírja a hozzá beérkező csomagokat, és így kiváló helyzetben van ahhoz, hogy alkalmazásszinten vizsgálhassa meg a kapcsolatot. Ne gondoljuk, hogy ez magától értetődő képesség, ha például a tűzfal csak SOCKS tűzfal, és nem valódi alkalmazásrétegbeli proxy, akkor megteheti azt, hogy az ügyfelektől beérkező csomagok adatait egyszerűen átmásolja az új, proxyzott csomagokba. Ha a tűzfal képes az alkalmazásszintű működésre – ilyen például a Zorp –, akkor arra is alkalmas, hogy az ügyfelek által küldött csomagok tartalma alapján bizonyos döntéseket hozzon meg.

Nézzünk egy példát! Tegyük fel, hogy van egy nyilvános webkiszolgálónk, ezt eredményesen lehet támadni egy hibás formátumú `HTTP GET` paranccsal, amely például túlságosan hosszú URL-t tartalmaz, és ezzel átmenetitár-túlcsordulást



1. ábra Állapotalapú csomagszűrő használatakor a csomagok az ügyfél és a kiszolgáló között közvetlenül utaznak, feltéve, hogy továbbításuk valamelyik szabály vagy állapottábla-bejegyzés alapján megengedett



2. ábra Alkalmazásszintű proxy használatakor a kapcsolat két részre osztható: az ügyfélnek a tűzfal kiszolgálóként látszik, a kiszolgálónak pedig ügyfélként viselkedik

(buffer-overflow) idéz elő. Alkalmazásszintű proxytűzfalunk fogadja az ügyféltől érkező kérést, ám amikor látja, hogy túlságosan hosszú URL-t tartalmaz, akkor az ügyfélnek hibaüzenetet, a kiszolgálónak pedig `reset` parancsot küld, és a támadócsomag továbbítása nélkül megszakítja a kapcsolatot.
Az előnyök mellett azonban hátrányokkal is számolni kell. A proxyzás eleve több erőforrást köt le, mint a csomagszűrés, és az alkalmazástudatos proxyzásra ez még inkább igaz. Az alkalmazásszintű proxyknak ezt a hátrányát sokszor túlhangsúlyozzák. Például a Zorp egy 700 MHz-es Celeron processzorral és 128 MB memóriával felszerelt gépen 88 Mb/s sebességgel képes a HTTP-forgalom vizsgálatára, ami közel kétszerese egy T3 WAN-kapcsolat sebességének. Ha a Zorp alá egy kétprocesszoros Pentium III gépet, 512 MB memóriát és SCSI RAID merevlemezeket teszünk, akkor körülbelül 480 Mb/s sebességre lesz képes, legalábbis a ➲ http://www.balabit.hu címen elérhető Zorp Professional v2 leírása szerint.
Az alkalmazásszintű proxyk tehát képesek rá, hogy a hálózati kapcsolatokba beékelődve minden csomagot nulláról indítva újra létrehozzanak, illetve intelligens döntéseket hozzanak arról, mely alkalmazásszintű parancsokat és adatokat szabad továbbítani – vagyis magasabb szintű védelmet nyújtanak. Munkájukat az egyes alkalmazások feltételezett működésének ismeretében, és nem csupán a hálózati csomagok kinézete alapján végzik. A legfőbb ellenérv az alkalmazásrétegbeli proxyk ellen a teljesítményre vonatkozik, ám – hála Moore törvényének – az egyre olcsóbbá és nagyobb teljesítményűvé váló gépeknek köszönhetően ez is könnyedén elhárítható.
Természetesen nem szabad elmenni egy másik, az alkalmazásszintű proxyk kapcsán sokak által felvetett tényező mellett, azaz jóval összetettebbek. Az alkalmazásszintű proxyk értelemszerűen sokkal kifinomultabbak, mint a csomagszűrők, és sokkal bonyolultabb a beállításuk is, ami nem meglepő, hiszen egy sportautó vezetése is nagyobb tudást kíván, mint egy taxi leintése. Egy Zorpot futtató tűzfalat vagy egy Secure Computing Sidewinder készüléket beállítani mindig is nehezebb lesz, mint egy Check Point Firewall-1 vagy Linux Netfilter/IP Tables alapú megoldást.
Vajon a nagyobb fokú biztonság nem éri-e meg azt a kis többletmunkát? Mint máskor is, mindenki maga döntse el, hogy mivel éri meg foglalkoznia. Lesz, aki úgy gondolja, érdemes dolgoznia a magasabb szintű védelemért, és lesz, aki nem öl bele több energiát. Bármi is legyen a végső döntés, remélem, meghozásához sikerült némi segítséget adnom. Írásom fennmaradó részében röviden bemutatom a Zorp telepítését és beállítását.

A Zorp változatai

A Zorp proxydémon a Linux-rendszermag felett, a normál Netfilterrel és a Balabit Kft. által készített Tproxy rendszermag-modulokkal párhuzamosan fut. A Zorpot Debian Linuxon fejlesztik, és a hozzá mellékelt leírás túlnyomó részében is feltételezik, hogy a használó Debiant futtat.
A Zorp három változatban érhető el: a Zorp GPL az ingyenes, GPL szerződéssel elérhető változat; a Zorp Unofficial a Zorp GPL újdonságokkal bővített próbaváltozata; a Zorp Professional, vagy csak egyszerűen Zorp Pro pedig kereskedelmi termék, amely számos fontos szolgáltatást nyújt.
Ha megvásároljuk a Zorp Prót, akkor egy rendszerindításra alkalmas CD-ROM-ot kapunk, amely nemcsak a Zorp Prót, de az ZorpOS-t is telepíti. Utóbbi egy lecsupaszított, a Zorp igényeihez igazított Debian-változat. A Zorp Pro egy teljesen szűz gépre negyed óra alatt felrakható, természetesen a beállítások megadása ebben nincs benne.
A Zorp Pro az új Zorp Management Servert (ZMS) is tartalmazza, amelynek segítségével több Zorp tűzfalat is kezelhetünk egyetlen központi felügyeleti állomásról. Az állomás távoli működtetésére a ZMC szolgál, ez egy grafikus felületet kínáló ügyfélprogram, amely Debian Linux és Windows alá egyaránt elérhető. A ZMS szolgáltatásait tekintve egyenértékű a Check Point Firewall-1 felügyeleti moduljával, amely a legfontosabb tényező volt abban, hogy a Check Point meghódíthatta a vállalati tűzfalak piacát. A ZMS-nek köszönhetően a Zorp kiváló megoldást jelent a nagy számú tűzfallal rendelkező nagyvállalatok számára is.
A Zorp Próval ellentétben a Zorp GPL és a Zorp Unofficial telepítéséhez már működő Linux-telepítésre van szükség, amely tartalmazza a következő összetevőket: glib 2.0, Python 2.1, libcap 1.10 és OpenSSL 0.9.6g. Működéséhez IP, tűzfal és átlátszó (transparent) proxytámogatással fordított 2.2-es, vagy `iptables`, `iptables/netfilter` kapcsolatkövetés (conntrack), `iptables` NAT és `iptables/netfilter tproxy`-támogatással – ez a Balabit Tproxy rendszermag-foltjával érhető el (➲ http://www.balabit.com/products/oss/tproxy) – fordított 2.4-es rendszermag szükséges. Az említett szolgáltatások mindegyikét modulba kell fordítani. Ha az operációs rendszer készen áll, a Zorp GPL-t bináris `deb` csomagokból tudjuk telepíteni, de a forrásból való fordítást is választhatjuk.
A következő alkalommal arról lesz szó, hogyan kell beállítani a Zorp GPL-t egy átlagos internet – szabad zóna (DeMilitarized Zone) – megbízható hálózati topológia (Trusted Network Topology) védelmére.

Mick Bauer

Linux Journal 2004. március 119. szám

Biztonságos programok készítése

Az új kódok, amiket te írtál, bizonyára biztonságosak, de mi legyen az olyan nagyobb tervezetekkel, amit mástól kell átvenned?

A biztonság kapcsán két fő terület jut eszünkbe, a hálózati és a fizikai biztonság. A biztonságos programok írására sokkal kevesebb figyelem jut. A biztonságos programok készítéséhez szükséges eljárások alkalmazása nem lenne ördöngösség, ám a szűkös idő és a tervezési szakasz lerövidítése miatt ritkán sikerül követni őket. Minden jó programozó ismeri az elveket, de nincs ideje alkalmazni őket – túl nagy nyomás nehezedik rá: nagymennyiségű kódot kell elkészítenie, a tervezettel pedig muszáj időben végezni.

Az 1970-es évek elején virágkorát élte a strukturált programozás. Akkoriban nemcsak a program, de az egész tervezet gondosan összeállított szerkezettel bírt, megvoltak a szükséges műszaki elvárások, részletes tervezési előírások, valamint a tervezést és a kódot is gondosan végigrágták. Természetesen a tervezetek nagyobbak lettek, befejezésükhöz több időre volt szükség, ám amikor elkészültek, a kód hibáinak megkeresése könnyen ment, és sokszor csak apróbb módosításokra volt szükség, hogy tökéletesen működjenek. Némelyik tervezet befejezése akkoriban több évig is eltartott.

Napjainkra a strukturált fejlesztési eljárások túlnyomó része eltűnt. Csakhogy a biztonság megteremtése a program tervezésénél kezdődik, és a munka színhelyéül szolgáló szervezet által felállított programírási szabványoktól függ. Senki nem gondol arra, hogy a kódja százszázalékosan biztonságos lesz, ugyanis ilyen kód nincs is. Arra azonban törekedni kell, hogy a kód üzembiztos és biztonságos legyen. Mit lehet tenni ennek érdekében? Írásomban néhány ötletet és három biztonságos programírást segítő eszközt szeretnék ismertetni. Beágyazott rendszerek tervezésekor és megvalósításakor különös gondossággal kell eljárni. A biztonságos programok írása végső soron a kód készítőjének feladata, az ő képességeinek függvénye, hogy megérti-e mit is kezelhet biztonságosként.

Hibakeresés

Minden függvény visszaad valamilyen típusú állapotjelzést, akár közvetlenül, akár az `errno` kimeneten keresztül. Ezek ellenőrzése egyszerű. A C++ alatt a kivételkezelés könnyen használható, ám annál nehezebb lehet a beállítása. Az utóbbi évek során, a C++ szabvány befejezése óta a kivételkezelés rengeteget fejlődött. Ha lehetséges, használni is kell őket.

A korábbi gyakorlat szerint a hibákat figyelmen kívül hagyták, mert mindenki úgy gondolta, maguk az átadott adatok helyesek. Mint később bebizonyosodott, ez a hozzáállás hibás. Az adat-átmenetitárak túlcsordulásai (data buffer overflows) miatt az elmúlt években rengeteg hibát kellett kijavítani. Ha beágyazott rendszerre fejlesztünk, a hibajelzések áttekintése nem mellőzhető. Ezután még mindig dönthetünk úgy, hogy a hiba jóindulatú, és ezért figyelmen kívül hagyhatjuk. Ha mégsem, meg kell kísérelni a hiba kijavítását. Amennyiben erre nincs mód, a rendszer alapállapotba állítását vagy a gép újraindulását kell eredményeznie. Sok esetben elegendő a program alapállapotba állítása is, ennek során a hibához vezető művelet újrakezdődik. Ez minden hibatűrő rendszer alapja. Az adott készülék típusától függően a gép újraindítása is megfelelő és elegendő megoldás lehet, míg akadnak olyan esetek, amikor elengedhetetlen valamilyen helyreállítási lehetőség biztosítása.

Karakterláncok kezelése

Az `sprintf`, az `strcpy` és az `strcat` helyett inkább az `strncpy` és az `strncat` függvényeket kell használni. Ezek szavatolják, hogy az átmeneti tár nem fog túlcsordulni, a túlnyúló részeket pedig elhagyják a karakterláncokból. Az `fgets` függvényt adatok beolvasására ne alkalmazzuk, mert átmenetitár-túlcsordulást okozhatunk vele. Ezek egészen egyszerű módosításoknak tűnnek, de könnyű róluk megfeledkezni, és éppen ezért a legtöbb kihasználható hiba forrása a helytelen karakterlánc-kezelés a programokban. Az önműködő ellenőrző programok ugyan elég jól szűrik az ilyen hibákat, de sokszor félrevezethetik használójukat. Gyakorta egy-egy karakterlánc-kezelő függvény használatát hibásnak minősítik, annak ellenére, hogy az adott környezetben a programozó megfelelő megoldást választott. Ezek azok a helyzetek, amikor a programozó képességei és tudása kerülnek előtérbe, ugyanis az ő feladata, hogy az ellenőrzőprogram naplóját átvizsgálja és megállapítsa, hogy mely esetekben szükséges módosításokat végezni.

Memóriaszivárgások és átmenetitár-túlcsordulások

A memóriaszivárgások önmagukban nem feltétlenül jelentenek biztonsági kockázatot. Ha viszont a memóriaterület több eljárás és adatszerkezet között van megosztva, kiaknázható hibát jelenthetnek.

A átmenetitár-túlcsordulások messze a leggyakoribb biztonsági kockázatforrások. Ha egy átmeneti tár a veremben található, akkor túlcsordulását előidézve törölhető vagy megváltoztatható egy függvény visszatérési címe. Amikor a függvényhívás visszatér, az eredeti helyett immár az új címmel teszi meg. Bizonyos átmenetitár-támadásokat (buffer attacks) a kupacon (heap) is végre lehet hajtani. Igaz, hogy egy ilyen támadást nehezebben lehet kivitelezni, de egyáltalán nem lehetetlen. A C nyelven írt programok sebezhetőbbek az ilyen támadásokkal szemben, de gyakorlatilag minden alacsonyszintű memóriakezelési és mutatóhasználati lehetőséget kínáló nyelv esetében jelentkezhetnek gondok. A mutatókezelés jó példa azokra a területekre, amelyeken nem szabad elfeledkezni a határok ellenőrzéséről.
A GNU C fordítóhoz létezik olyan kiterjesztés – a fordító fordításakor kell beépíteni –, amely a határok ellenőrzését elvégzi. Kiegészítő jelleggel használható, a programot külön kódrészekkel bővíti. A kipróbálás időszakára ezek a részek engedélyezhetők és használhatók, majd az alkalmazás telepítésekor le kell tiltani őket. Ennek oka az, hogy ezek a kódrészek a határok átlépésekor üzeneteket jelenítenek meg a képernyőn. Ha a program egy munkaállomáson fut, az üzenetek nem okoznak gondot, de egy beágyazott rendszeren, ahol általában nincs konzol, értelmüket vesztik.
E téma kapcsán felvetődhet az átmeneti tárak statikus lefoglalásának ötlete, ezzel a hibalehetőség azonnal megszűnne. Sajnos a szabott méretű átmeneti tárak jelenléte is kihasználható, a beléjük másolt adatok mérete ugyanis továbbra is nagyobb lehet az átmeneti tárénál. Ha az adatmásolás megtörténik, ugyancsak túlcsordulásra kerülhet sor. A kockázat csak olyan módon csökkenthető, ha az átmásolt adatok mérete nem haladhatja meg az átmeneti tárét. A karakterláncok helyét dinamikusan lefoglalva a programok tetszőleges méretű bemeneteket képesek kezelni. A gond ekkor csak az, hogy a program kifuthat a szabad memóriából. Beágyazott rendszeren az ilyen hiba végzetes lehet, munkaállomáson pedig a virtuális memóriarendszer kezdhet dolgozni serényen, ami viszont a számítógép teljesítményét veti vissza. C++ alatt az `std::string` osztály alkalmas a dinamikus szemlélet gyakorlatba ültetésére. Ha az osztály adatait egy `char *` mutatóval kezeljük, átmenetitár-túlcsordulás léphet fel. Az egyéb karakterlánc-kezelő könyvtáraknál ilyesmi nem fordulhat elő, de a programozónak mindvégig tekintettel kell lennie e korlátokra.

A bemenet ellenőrzése

Ha egy program kívülről kapja a további működése alapjául szolgáló adatokat, érdemes ellenőrizni, hogy az adatok mérete nem haladja-e meg a megengedettet, helyesek-e és nem tartalmaznak-e meg nem engedett típusú elemeket. Például, ha a bevitt adatsor csak A és Z közötti nagybetűkből állhat, akkor a függvénynek minden mást vissza kell utasítania. Ugyancsak ellenőrizni kell, hogy az adatok hossza érvényes-e. Nem is olyan rég volt, amikor az adat fogalma alatt mindenki 80 karaktert értett, azaz egy lyukkártya tárolási méretét. Manapság az adatmennyiség szinte tetszőleges lehet, típusára nézve pedig szöveges, bináris és titkosított egyaránt előfordulhat. Határok azonban még ma is léteznek. Az adatok ellenőrzését el kell végezni, és ha az eredmény nem megfelelő, a bemenetet vissza kell utasítani.
Sokszor nem elég a bemenet méretét a felső határhoz viszonyítani, meg kell vizsgálni azt is, hogy van-e olyan hosszú, mint amit alsó határként megszabtunk. A karakterláncokat abból a szempontból is ellenőrizni kell, hogy érvényes értékeket és karaktersorozatokat tartalmaznak-e. Ha az ellenőrzés alá vont adatok bináris formátumúak, és valamiért így is kell maradniuk, akkor a legjobb valamilyen általános kilépési karaktert használni a bináris formátum jelzésére. Ha szám típusú adatokkal dolgozunk, az értéktartomány betartására kell ügyelnünk. Ha pozitív egész számot várunk, mindig ellenőrizzük, hogy nullánál nem kisebbet kaptunk-e. Ha van valamilyen legnagyobb érték, ellenőrizzük, hogy nem léptük-e át. A ***limits.h*** fájlban a legtöbb típus maximum- és minimumértéke meg van adva, így ennek segítségével könnyedén ellenőrizhető a rendszer határértékeinek betartása.

Segédeszközök

A legtöbb fejlesztő került már olyan helyzetbe, hogy készen állt a kód, de csak nagyon kevés idő és energia maradt a lehetséges biztonsági hiányosságok felkutatására. Végül is a kód működik, miért kellene javítani rajta? Nagyon sok helyen tartja magát ez a felfogás. Ha mégis fény derül a kód valamilyen hibájára, azonnal fontos lesz a kijavítása, miként a felelőskeresgélés sem maradhat el.
Mit lehet tenni, ha rövid idő alatt kell meglelni a lehetséges hibaforrásokat? Három olyan eszközt említenék meg, amelyek képesek a lehetséges hibák felismerésére és jelentésbe gyűjtésére. Beágyazott rendszereken is használhatók, bár a valóságban a legtöbb fejlesztői munka vegyes környezetben zajlik. A munka nagy részét a munkaállomásokon szokták elvégezni, a célrendszeren már csak a finomhangolást szükséges végrehajtani.
A Flawfinder, a RATS és az ITS4 három olyan csomag, amely a forrásfát végigvizsgálja és a lehetséges hibákat jelenti. Kimenetük a forrásmodult és a kódsort is pontosan megjelölő hibalista. A hibajelzésekhez súlyozás is társul, amellyel a sebezhetőség mértékét adják meg.
Kódrészletünkben egy mintakódon futtatott Flawfinder kimenetének részlete látható. A hibák komolyságát 0 és 5 közötti számokkal jellemzi, ahol a 0 a csekély, az 5 pedig a komoly kockázatot jelenti.
A program akár jelentős számú hibát is jelezhet, ezt követően már a fejlesztő döntésén múlik, hogy kijavítja vagy figyelmen kívül hagyja-e őket. Sok fejlesztő tart tőle, hogy ezek a segédeszközök módosítsák is a kódot, de gondoljunk bele, még mindig jobb a módosítandó részeket kiválogatni, mint tömeges változtatásokat segédeszköz nélkül elvégezni. A Flawfinder program saját adatbázist és szabályhalmazt alkalmaz, ez a leggyakoribb biztonsági hiányosságok listáját tartalmazza.

Összegzés

Biztonságos kódot írni könnyű. Annak mérlegelése, hogy milyen szolgáltatásokat kell megvalósítani, és ezek hogyan használhatók támadások kivitelezésére, még a tervezési időszakban szükséges eldönteni és megvizsgálni. Az ellenőrzési eljárásoknak a különféle támadások és helytelen

használati módok kipróbálására is ki kell terjednie. Ezeket a próbákat teljesen önműködővé tenni fényűzés lenne, és a felhasználó is jóval később jutna hozzá a – kétségtelenül jobb minőségű – termékhez. Az itt említett módszerek és eszközök csak segítséget nyújtanak mindehhez. A programok biztonságának sorsa továbbra is a fejlesztők kezében marad, életútjuk az ő fejükben kezdődik.

Linux Journal 2003. november 115. szám

Cal Erickson (cal_erickson@mvista.com) jelenleg a MontaVista Software vezető Linux-tanácsadója. Mielőtt csatlakozott volna a MontaVista csapatához, a támogatást vezető mérnök volt a Mentor Graphics beágyazott alkalmazások fejlesztési részlegénél. Cal több mint harminc éve dolgozik a számítástechnikai iparágban, tapasztalatait számítógépgyártóknál és végfelhasználói termékeket fejlesztő cégeknél szerezte.

A Flawfinder kimenete

Flawfinder version 1.21, (C) 2001-2002 David A. Wheeler.
Number of dangerous functions in C/C++ ruleset: 127 (A C/C++ szabályhalmazban található veszélyes eljárások száma: 127)

Examining ../../pelda_kod/msgqueue/mksem.c (Az mksem.c fájl vizsgálata.)
../../pelda_kod/msg_queue/msgtool.c:73 [4] (buffer) strcpy:
Does not check for buffer overflows when copying to destination. (Nem végez átmenetitár-túlcsordulás vizsgálatot a célba másolás alatt.)
Consider using strncpy or strlcpy (warning, strncpy is easily misused). (Fontolja meg az strncpy vagy az strlcpy használatát. Vigyázat, az strncpy használata könnyen elrontható!)
../../pelda_kod/msgqueue/mksem.c:34 [4] (shell) system:
This causes a new program to execute and is difficult to use safely. (Új program futtatását okozza, biztonságosságát nehéz garantálni.)
Try using a library call that implements the same functionality if available. (Próbáljon könyvtári hívást használni erre a célra, ha lehetséges.)
../../pelda_kod/pipes/fifo/fifo_out.c:28 [4] (race) access:
This usually indicates a security flaw. If an attacker can change anything along the path between the call to access() and the file´s actual use (e.g., by moving files), the attacker can exploit the race condition. (Általában biztonsági hiányosságot jelent. Ha a támadó bármit meg tud változtatni az access() hívás és a fájl tényleges használatának ideje között (például fájlok átmozgatásával), akkor kihasználhatja a versenyhelyzetet.) Set up the correct permissions (e.g., using setuid()) and try to open the file directly. (Állítson be helyes engedélyeket (például a setuid() segítségével), és próbálja meg közvetlenül megnyitni a fájlt.)
../../pelda_kod/process_control/proc_mem_info/proc_mem_info.c:139 [4] (buffer) sscanf:
The scanf() family´s %s operation, without a limit specification, permits buffer overflows. (A scanf() család %s formátumú hívása korlát nélkül alkalmazva átmenetitár-túlcsorduláshoz vezethet.) Specify a limit to %s, or use a different input function. (Adjon meg korlátot a %s formátumú híváshoz, vagy válasszon másik beviteli függvényt.)
../../pelda_kod/msg_queue/sender/snd_thread.c:70 [3] (random) srand:
This function is not sufficiently random for security-related functions such as key and nonce creation. (A függvény nem biztosít elegendő véletlenszerűséget ahhoz, hogy biztonsági eljárásokban – például kulcsok vagy egyszer használatos véletlen adatok előállításához – használjuk.) Use a more secure technique for acquiring random values. (A véletlen értékek előállításához használjon biztonságosabb módszert.)
../../pelda_kod/dlopen/dltest.c:30 [2] (misc) fopen:
Check when opening files – can an attacker redirect it (via symlinks), force the opening of special file type (e.g., device files), move things around to create a race condition, control its ancestors, or change its contents? (Fájlok megnyitásakor ellenőrizze, vajon a támadó nem tudja-e átirányítani a műveletet – például közvetett hivatkozások segítségével –, nem tudja-e különleges fájl – például eszközfájl – megnyitására rávenni a rendszert, nincs-e módja versenyhelyzet létrehozására vagy a szülőelemek, vagy a tartalom módosítására.)
../../pelda_kod/msg_queue/receiver/rcvr.c:51 [2] (buffer) char:
Statically-sized arrays can be overflowed. (A statikus méretű tömbök túlcsordulhatnak.) Perform bounds checking, use functions that limit length, or ensure that the size is larger than the maximum possible length. (Végezzen határellenőrzést, használjon hosszkorlátozó függvényeket vagy ellenőrizze, hogy a tömb mérete nagyobb-e, mint a lehetséges leghosszabb bemeneté.)
../../pelda_kod/dlopen/another_dlopen_test/obj.c:15 [1] (buffer)
strlen:
Does not handle strings that are not \0-terminated (it could cause a crash if unprotected). (Nem kezeli a nem \0 végződésű karakterláncokat. Kellő védelem hiányában összeomlást okozhat.)

...

Number of hits = 139 (Találatok száma = 139)
Number of Lines Analyzed = 5491 in 2.67 seconds (2527 lines/second) (Elemzett sorok száma = 5491, idő: 2,67 másodperc (2527 sor/másodperc)
Not every hit is necessarily a security vulnerability. (Nem minden találat jelent feltétlenül biztonsági hiányosságot.)
There may be other security vulnerabilities; review your code! (Ellenőrizze a kódot, további biztonsági hibákat is tartalmazhat!)

Biztonságos levelezés az LDAP segítségével (1. rész)

Célunk egy biztonságos, méretezhető levélkiszolgáló üzembe helyezése, amely a meglévő LDAP-kiszolgálónk segítségével hitelesíteni tudja az IMAP-felhasználókat.

Nemrég végeztünk egy OpenLDAP-kiszolgáló építésével. Mostani és következő írásomban az LDAP egyik leghasznosabb alkalmazási területével, a hitelesítéssel és a telefonkönyv-szolgáltatással foglalkozom. Természetesen nem az LDAP témát fogom továbbvinni. A központi téma az IMAP (Cyrus) lesz, illetve az, hogy miként tudunk a tudására és az LDAP szolgáltatásaira támaszkodva biztonságos távoli levelezési lehetőséget nyújtani. Értelemszerűen feltételezzük, hogy már üzembe helyeztünk egy LDAP-kiszolgálót – vagy legalábbis ismerjük ennek módját –, és feltöltöttük a szükséges felhasználóadatokkal.

Szállító és továbbító ügynökök

Elsőként tekintsük át az IMAP szerepét az elektronikus táplálékláncon belül. Az IMAP, vagyis az Internet Message Access Protocol (internetes üzenet-hozzáférési protokoll, leírása az RFC 3501 dokumentumban található) a levélszállító ügynökök (Mail Delivery Agent, MDA) egyik protokollja. Míg a levéltovábbító ügynökök (Mail Transport Agent, MTA) – például a Postfix és a Sendmail – hálózatok között továbbítják a leveleket, addig az MDA-k az MTA-k és a megcímzett levelesládák közötti szállításáért felelősek. A „Building secure servers with Linux” című könyvemből kiragadva a példát, az MTA a postahivatalok között ingázó teherautóhoz, az MDA pedig a küldeményeket a hivatalból az otthonunkba hozó levélkihordóhoz hasonlítható. Egy IMAP alapú MDA-rendszer két részből áll: egy IMAP-kiszolgálóból, amely a felhasználók levelesládáinak ad helyet és valamilyen MTA-tól fogadja a leveleket, valamint az IMAP-ügyfélprogramot futtató felhasználók csoportjaiból. A három legnépszerűbb nyílt forrású IMAP-kiszolgáló a University of Washington IMAP (UW IMAP), a Carnegie Mellon Egyetem háza tájáról származó Cyrus IMAP, illetve az Inter7 Internet Technologies által fejlesztett Courier IMAP. A legelterjedtebb IMAP-ügyfélprogramok a Netscape/Mozilla Communicator, a Ximian Evolution, a Microsoft Outlook Express, a KMail, a `mutt`, a `pine` és az Apple Mac OS X Mail. Az IMAP-ügyfelekkel e helyen nem foglalkozunk, ami viszonylag könnyű beállításuk és használatuk miatt talán megbocsátható. Ráadásul mivel a legtöbb IMAP-ügyfél az IMAP-kiszolgálók túlnyomó részével gond nélkül működik, a dolognak ezen a részén kár lenne túl sokat rágódni.

Melyik IMAP-kiszolgálót válasszuk?

Az IMAP-rendszer építésekor a rendszergazda első döntése a megfelelő kiszolgáló kiválasztása. Milyen fontosabb különbségek fedezhetők fel az UW IMAP, a Courier IMAP és a Cyrus IMAP között? Mivel az utóbbi kettő szolgáltatásai elég gyakran bővülnek, a kérdés részletes megválaszolása alól kimenteném magam. Amit mondhatok:

1. A három megoldás közül az UW IMAP a legkevésbé rugalmas, mivel csak helyi felhasználói fiókokon alapuló levélszállítást támogat. Minden felhasználó bejövő leveleit egyetlen fájlban, a */var/mail/felhasználónév* név alatt tárolja. Ennek a megoldásnak két hátránya van: egyrészt minden levelező személynek rendszerfelhasználót kell létrehozni, másrészt egy-egy felhasználó postaládáját egyszerre csak egy folyamat tudja írni, s ez fájlzárolási hibákat okozhat.
2. A Courier IMAP-ot, amely jelenleg a Courier Mail Server része, a `qmail` levélkönyvtár rendszerének támogatására tervezték. Esetében a felhasználók saját levélkönyvtárakkal rendelkeznek, ezek a leveleket a teljesítmény és a fájlzárolási hibák elkerülése szempontjából egyaránt a kedvezőbb módon: külön fájlokban tárolják. A Courier adatbázisban is tárolni tudja a leveleket (lásd a 3. pontot), és az újabb változatai már támogatják az LDAP alapú hitelesítést.
3. A Cyrus IMAP telepítése a másik kettőjénél bonyolultabb, nagyrészt a működéséhez szükséges Cyrus SASL hitelesítő könyvtáraknak köszönhetően. A Cyrus IMAP viszont saját felhasználó- és levéladatbázist használ, mindkettő teljesen független az operációs rendszertől, így rendszerfelhasználók hozzáadása nélkül bővíthetjük a levelezéshez hozzáféréssel rendelkezők körét. A levelek egyszerű fájlok helyett adatbázisban való tárolásából fakadó teljesítménynövekedése magától értetődő.

Én legtöbbet a Cyrus IMAP-ot használtam, így a továbbiakban erről fogok bővebben szólni. Az UW IMAP, a Courier IMAP és a Cyrus IMAP weblapjára ellátogatva – lásd a *Kapcsolódó címeket* –, majd a szolgáltatások listáját áttekintve mindenki maga döntse el, hogy melyik program felel meg a leginkább az igényeinek. Írásom hátralévő részeiben remélhetőleg a tőlem eltérően döntők is találnak hasznos tudnivalókat, legalább az elgondolások szintjén.

A Cyrus IMAP letöltése és telepítése

Jómagam a jobb csomagkezelők nyújtotta változat- és javításkezelő szolgáltatások végett nagyon szeretem a futtatható állományokból álló csomagokat. Mielőtt valaki rosszra gondolna, a nagyobb terjesztések csomagkezelőit kivétel nélkül a jobbak közé sorolom. Azt javaslom mindenkinek, hogy a Cyrus IMAP telepítését saját terjesztésének frissítőszolgáltatása segítségével vagy valamilyen adathordozóról végezze el, ha ez lehetséges. A Cyrus SASL-csomagra is szükség lesz, ez adja a Cyrus IMAP számára a hitelesítő háttérrendszert. Az `SMTP AUTH` úgyszintén ezt használja, ezért lehetséges, hogy már fel is van telepítve.

A SuSE 8.2-es rendszerhez a `cyrus-imapd` és a `cyrus-sasl2` RPM-ekre lesz szükség, míg például Debian 3.0-s alatt a `packages cyrus-common`, a `cyrus-imapd`, a `libsasl2` és a `sasl2-bin` csomagokat kell beszerezni. SuSE és Debian futtatásakor egyaránt figyelembe kell venni, hogy a korábbi rendszereken régebbi – 2.0-s változat előtti – Cyrus SASL-csomagok lehetnek feltelepítve; a Cyrus IMAP által végzett, a későbbiekben ismertetett LDAP alapú hitelesítéshez viszont a SASL 2.0-s vagy újabb változat szükséges. Ha a futó terjesztésre 2.0-s előtti SASL-csomag van telepítve, akkor Cyrus SASL forráskódját le kell tölteni és le kell fordítani (ez az ➲ ftp.andrew.cmu.edu/pub/cyrus-mail címen érhető el).

A Red Hat 9.0-s rendszer alatt picivel több munkára lesz szükség, mint a legújabb SuSE- vagy Debian-terjesztések esetében, mivel a 7.1-es változat óta a Cyrus IMAP-csomagok kikerültek a Red Hat-terjesztésből. Elsőként a `cyrus-sasl`, a `cyrus-sasl-plain` és a `cyrus-sasl-md5` csomagot kell telepíteni, ezek a normál Red Hat 9.0-s terjesztés részei. Ezt követően egy SRPM-csomag formájában le kell tölteni magát a Cyrus IMAP-ot, amit a ***home.teleport.ch/simix*** címről tehetünk meg. (Az oldal karbantartásáért köszönet illeti a svájci ***Simon Matter***-t.)

Ha még sosem akadt dolgunk forrás-RPM-mel, vagyis SRPM-mel, ne essünk kétségbe; futtatható RPM-et az SRPM-ből a következő paranccsal készíthetünk:

```
rpm --rebuild [--target geptipus] srpm.nev.src.rpm
```

Ebben az ***srpm.nev.src.rpm*** az SRPM fájl neve, a „geptipus" pedig a gép típusát adja meg (például i386, i586, i686). Nekem például egy Pentium III alapú kiszolgálóm van, ezen az `rpm --rebuild --target i686 cyrus-imapd-2.1.12-7.src.rpm` utasítást kellett kiadnom. Ugyan a `--target` átadott érték elhagyható, de ha nagyobb IMAP-felhasználó adatbázist fogunk kezelni, akkor a Cyrus IMAP-ot érdemes a saját gépünk processzortípusának megfelelően fordítani, mert így az alapértelmezett i386 fordításhoz képest érzékelhető sebességnövekedést fogunk tapasztalni. Az `rpm` ezt követően önműködően több új futtatható RPM-et is elkészít, kifejezetten a helyi rendszerhez testreszabva. Ezek az RPM-ek a ***/usr/src/redhat/RPMS/*** könyvtárba kerülnek, amelyben az alkönyvtár pontos neve a `--target` átadott értékétől függ; az i386/ az alapértelmezett. A keletkező RPM-ek a következők: `cyrus-imapd`, `cyrus-imapd-utils`, `cyrus-imapd-devel` és `perl-Cyrus`. Telepítésüket az *`rpm -Uvh fájlnév`* paranccsal végezhetjük el.

A SASL beállítása

A továbbiakban még két dolgot végzünk el: egyrészt az IMAP-felhasználók hitelesítése terén kiaknázzuk meglévő LDAP-kiszolgálónk képességeit, másrészt Cyrus IMAP-kiszolgálónkat úgy állítjuk be, hogy csak SSL feletti, titkosított kapcsolatokat fogadjon a végfelhasználóktól. Korábbi írásaimban kellően felmagasztaltam a központi hitelesítést, így az LDAP által kínált értékes lehetőségek remélhetőleg senkinek nem jelentenek újdonságot. Korábban már ejtettem szót arról, hogy az elektronikus levelek nyílt szövegben folytatott lekérése milyen veszélyeket rejt magában. Normál POP3- és IMAP-kapcsolatoknál a felhasználónév, a jelszó és a levelekben található adatok egyaránt titkosítatlanul haladnak keresztül a hálózaton. Minden hallgatózás jellegű támadásnak ki vannak téve, ami különösen a vezeték nélküli hálózaton vagy az interneten – amely a világ legmegbízhatatlanabb hálózataként kezelhető – keresztül végzett lekérdezésekre igaz. Térjünk vissza a SASL témájához! Mivel a Cyrus IMAP és a Cyrus SASL egyaránt a Carnegie Mellon Egyetemről származik, és a Cyrus fejlesztői nyilván nem akartak valami teljesen újat alkotni, a Cyrus IMAP a hitelesítés tekintetében a Cyrus SASL szolgáltatásaira támaszkodik. Álljunk csak meg! Nem éppen erre a célra akarunk LDAP-ot használni? De igen. A SASL abból a szempontból is feleslegesnek tekinthető, hogy saját felhasználó-adatbázis használatára tervezték.

A SASL azonban nemcsak saját adatbázisát tudja használni, de a hitelesítési műveleteket más hitelesítési források – például PAM vagy LDAP – felé is tovább tudja adni. A legegyszerűbb eljárás a `saslauthd`, a SASL hitelesítési démon beállításainak módosítása, ezt a ***/etc/saslauthd.conf*** fájl tartalmának átírásával tehetjük meg. Az *1. kódrészletben* egy ***saslauthd.conf*** mintafájl tartalma látható.

Az `ldap_servers` kulcsszó után szóközzel elválasztva az LDAP-kiszolgálók URI-jait kell felsorolni. Az 1. kódrészletben nyílt szövegalapú LDAP-kapcsolatot adtam meg a helyi LDAP-folyamathoz. Természetesen az `ldap` helyett titkosított kapcsolatokkal dolgozó `ldaps` protokollt is választ-

1. kódrészlet A /etc/saslauthd.conf mintafájl

```
ldap_servers: ldap://localhost/
ldap_search_base: dc=proba,dc=org
ldap_bind_dn: cn=kiszolgalok,dc=proba,dc=org
ldap_bind_pw: ide_kell_irni_a_jelszot
```

2. kódrészlet A /etc/imapd.conf példafájl

```
configdirectory: /var/lib/imap
partition-default: /var/spool/imap
admins: cyrus bauer
sievedir: /var/lib/imap/sieve
sendmail: /usr/sbin/sendmail
hashimapspool: true
sasl_pwcheck_method: saslauthd
sasl_mech_list: PLAIN
tls_cert_file: /var/lib/imap/slapd3.pem
tls_key_file: /var/lib/imap/slapd3key.pem
tls_cipher_list: HIGH:MEDIUM:+SSLv2
```

hattam volna, illetve távoli gép teljesen minősített tartománynevét vagy IP-címét is megadhattam volna, mint például ➲ ldaps://ldap.proba.org.
Az `ldap_search_base` a felhasználók megkülönböztető neveinek (DN) az alapja, vagyis a közös része. Az `ldap_bind_dn` és az `ldap_bind_pw` az a DN és jelszó, amelyet a `saslauthd` az LDAP-kiszolgáló elérésekor fog használni. Erre a célra mindenkinek egy külön LDAP-bejegyzés létrehozását javaslom. Az 1. kódrészletben a „kiszolgalok” egy különleges LDAP-fiók neve, ennek `objectClass` jellemzője `simpleSecurity` `Object`, vagyis a DN és az `objectClass` mellett egyetlen további jellemzővel rendelkezik, és ez a `userPassword`.
A kifejezetten a kiszolgálók számára létrehozott LDAP-fiók segítségével az LDAP naplóiban könnyen meg lehet különböztetni a kiszolgálófolyamatok és a felhasználók által indított lekérdezéseket. Mindezt sokkal nehezebb volna elvégezni, ha az IMAP például személyes LDAP-fiókok azonosító adatait használná. Ha még kifinomultabb naplózást szeretnénk végezni, akkor minden LDAP-lekérdezéseket végző kiszolgálófolyamat számára külön LDAP-fiókot hozhatunk létre, mint például Cyrus és Postfix.
Jómagam ezekkel a lehetőségekkel éltem a saját */etc/saslauthd.conf* fájlom megírásakor, de természetesen továbbiak is léteznek. A Cyrus SASL csomagban található egy `LDAP_SASLAUTHD` fájl, ami a *saslauthd.conf* fájlban használható beállítások leírását tartalmazza. Ez a fájl a forráskódcsomag *saslauthd* könyvtárában található, ha pedig a SASL telepítését futtatható csomagból végezzük, akkor oda kerül, ahová az adott terjesztés a leírásokat helyezi; valószínűleg a */usr/share/doc* könyvtár valamely alkönyvtárába.
A */etc/saslauthd.conf* átírásán túl arról is meg kell győződni, hogy a `saslauthd` indítása a `-a ldap` kapcsolóval történik-e. Red Hat alatt ehhez a */etc/sysconfig/saslauthd* fájlt kell módosítani úgy, hogy a MECH átadott érték `ldap` legyen.
A SuSE-terjesztésnél ugyanezt a fájlt kell átírni, de az átadott érték neve `SASLAUTHD_AUTHMECH`. Ismétlem, a kívánt végeredmény az, hogy a `saslauthd` a `-a ldap` kapcsolóval induljon.
Ha végeztünk a `saslauthd` beállításainak módosításával és újraindítottuk a démont, akkor készen állunk az IMAP-szolgáltatás üzembe helyezésére. Ezzel már nem lesz gond.

A Cyrus IMAP beállítása

A Cyrus IMAP működését befolyásoló beállítások túlnyomó része a */etc/imapd.conf* fájlban található. A *2. kódrészlet* egy *imapd.conf* mintafájl tartalmát szemlélteti.
Mint valószínűleg feltűnt, az *imapd.conf* beállításainak jelentős része a Cyrus IMAP által igényelt programok és egyebek elérési útját adja meg. Nem fogok részletesebben foglalkozni velük, leírásuk az ***imapd.conf(5)*** súgóoldalon található meg –, de a 2. kódrészlet nem alapértelmezett értékkel ellátott beállításain gyorsan végigszaladnék:
`admins` – az IMAP-rendszernek a `cyradm` segédeszközzel való felügyeletére meghatalmazott Cyrus IMAP-felhasználók listája. (A `cyradm` használatáról egy későbbi írásban lesz szó.) Ha a `sasl_pwcheck_method` beállításnak a `saslauthd` értéket adjuk, és ha a `saslauthd` már be van állítva az LDAP használatára, akkor már rá is vettük a Cyrus IMAP-ot az LDAP alapú hitelesítésre. Ilyenkor hiába létezik `cyrus` felhasználó a helyi Linux-rendszeren, vagyis a */etc/passwd* fájlban, a `cyrus` egy külön LDAP-bejegyzést is igényel.
A `cyradmin` futtatásakor meg kell adni a `cyrus` jelszavát, ekkor az adatbázisban lévő Cyrus jelszavára és nem a linuxos `cyrus`-éra van szükség. Másként fogalmazva, az `admins` kulcsszó után felsorolt fiókoknak mindig abban az adatbázisban kell jelen lenniük, amelyet a `sasl_pwcheck_method` beállításnál kiválasztottunk.
Amikor a Cyrus IMAP-ot – akár forrásból, akár futtatható csomagból – telepítettük, egy új felhasználót (`cyrus`) kellett létrehoznunk, és átadni neki a legtöbb Cyrus IMAP-fájl tulajdonjogát. Mint a jobb szolgáltató démonok, a Cyrus IMAP is külön felhasználóként fut, és feladatainak túlnyomó részét nem rendszergazdaként látja el.
A 2. kódrészletben további három általam testreszabott beállítás látható, ez a `tls_cert_file`, a `tls_key_file` és a `tls_cipher_list`. Szerepük rendre megegyezik az OpenLDAP *slapd.conf* fájljában található `TLSCertificateFile`, `TLSCertificateKeyFile` és `TLSCipherSuite` beállításokéval. Azért akartam erre külön felhívni a figyelmet, mert az itt megadott tanúsítvány- és kulcsfájl megegyezik a rendszer OpenLDAP-kiszolgálója által használttal. Ennek az az oka, hogy rendszeremben a Cyrus IMAP és az OpenLDAP ugyanazon a kiszolgálón fut, nincs tehát semmi okom arra, hogy mindkét szolgáltatás külön kiszolgáló-tanúsítványokat használjon. A */etc/openldap* könyvtárból viszont – a tulajdonlások és az engedélyek kezelésének egyszerűsítése végett – mindkét fájlt a */var/lib/imap* könyvtárba másoltam.
Ha az LDAP-szolgáltatás külön állomáson futna, akkor a 2003. augusztusi cikkemben ismertetett eljárást követve az alábbi parancsokkal külön TLS-tanúsítvány – kulcspárost kellett volna létrehoznom az LDAP-kiszolgáló számára:

```
openssl req -new -x509 -nodes -out
slapdtanusitvany.pem
➥-keyout slapdkulcs.pem -days 365
```

Bármelyik megoldást választjuk is, a tanúsítvány- és a kulcsfájlt egyaránt a cyrus tulajdonába kell adnunk, és csak a tulajdonosuk kaphat rájuk olvasási jogot.
Ha a Cyrus IMAP telepítését forrásból végezzük el, akkor alapértelmezett SSL-kulcsokat fog használni, ebből viszont hiba származik, ha egy IMAP-ügyfél TLS- és nem SSL-titkosítás használatával próbál csatlakozni. Nemcsak megbízhatósági, de minden egyéb szempontból is rossz ötlet az alapértelmezett tanúsítványt vagy kulcsokat használni, bármihez is. Mindig használjuk a meglévő kiszolgálótanúsítvány – kulcspárosunkat, vagy hozzunk létre újat. Így az IMAP-kiszolgálónk megbízhatóbb és biztonságosabb lesz.
Nincs más hátra, mint a Cyrus IMAP újraindítása (`/etc/init.d/cyrus-imapd restart`), és a felhasználók hozzáadása a cyradm segítségével. Ezt a részt azonban, akárcsak a helyi MTA beállítását a levelek átadására az IMAP-nak, hagyjuk a következő alkalomra.

Mick Bauer

Linux Journal 2003. november 115. szám

Tanuljuk meg használni az ioctl rendszerhívásokat!

Eszközvezérlés ioctl hívásokkal.

Néhány évvel ezelőtt volt egy laptopom, amit a munkában és otthon egyaránt használtam. A hálózati csatlakozás megkönnyítése végett, illetve hogy ne kelljen helytől függően kézzel állítgatnom a dolgokat, úgy döntöttem, hogy mindkét helyen DHCP-t fogok használni. A munkahelyemen már megvolt, így hát otthon is felállítottam egy DHCP-kiszolgálót. A dolog jól működött egészen addig, amíg meg nem próbáltam hálózat nélkül elindítani a gépet. Ilyenkor ugyanis a gép rengeteg időt töltött el azzal, hogy – sikertelenül – megpróbálta megtalálni a DHCP-kiszolgálót, és csak azután folytatta az indítási folyamatot. Arra gondoltam, az lenne az eszményi megoldás, ha a rendszer ethernetcsatoló nélkül indulna, és csak akkor indítaná el a hálózatot, ha a kábel egy jelelosztóhoz (hub) csatlakozik, vagyis ha az ethernetkártyán világít a „link" (kapcsolat) lámpa. A legjobbnak az tűnt, ha írok egy olyan parancsfájlt, amely egy programocska visszatérési kódjából meg tudja állapítani, hogy az adott hálózati eszközön a „link" jelzés aktív-e. A feladat annak megállapítása volt, hogy miképpen lehet lekérdezni a 10/100Base-T kártyám kapcsolati állapotát.

Túl nagy gyakorlatom az alacsony szintű Linux-programozásban nem lévén eltartott egy ideig, amíg rájöttem, hogy az eszközvezérlőkkel folytatott ilyesfajta kapcsolatot általában az `ioctl` könyvtárhívásokon keresztül lehet megvalósítani (az `ioctl` az I/O control rövidítése), amelynek prototípusait a ***sys/ioctl.h*** állományban találjuk:

```
int ioctl(int, int, ...)
```

Az első érték a fájlleíró. Linux alatt minden eszközt fájlként érünk el, általában a célként kiválasztott eszközhöz tartozó fájlleírót használjuk fel. Az ethernetkártyák esetében azonban a fájlleíró egyszerűen egy nyitott foglalat (socket). Tulajdonképpen a foglalatot nem szükséges a kérdéses kártyához rendelnünk.

Az ***ioclt.h*** állományban található második érték a meghatározott `ioctl`-kérelemnek megfelelő egész szám. A kérelmek eszközről eszközre változnak: a soros eszközön például beállíthatjuk a sebességet, de a nyomtatón már nem. Természetesen a hálózati csatolókhoz is különleges parancskészlet tartozik.

A további értékek elhagyhatók és az egyik vagy másik eszköz `ioctl` megvalósításától függően változhatnak. Amennyire én tudom, a harmadik érték mindig jelen van, ugyanakkor háromnál többet sem láttam eddig.

A harmadik érték, úgy tűnik, általában egy szerkezetmutató. Ezáltal mindkét irányba tetszőleges mennyiségű adatot adhatunk át – a szerkezetben tárolt adatot egyszerűen a mutató átadásával továbbítjuk.

Az `ioctl` működésének egyszerű példáját az 1. listán bemutatott programban (lásd 59. CD-melléklet Magazin/ioctl könyvtárában) láthatjuk, amely egy adott jel állapotát ellenőrzi a soros kapun.

A program megnyitja a tty-eszközt (soros kapu), majd a soros kapu fájlleírójával, a `TIOCMGET` (azaz a modem helyzetbitjeinek lekérése) paranccsal és az eredmény tárolására szolgáló egész értékre mutató pointerrel meghívja az `ioctl` függvényt.

Ezután ellenőrizzük az `ioctl` eredményt, hogy megtudjuk, nem történt-e a kérés feldolgozása során valamilyen hiba. Amennyiben nem volt gond, ellenőrzésképpen a `TIOCM_DTR` tartalma és a visszaadott értékek közt logikai `ÉS` műveletet végzünk. Ez a lépés igaz vagy hamis, azaz nem nulla vagy nulla értéket adhat vissza.

Ethernetmeghajtókhoz ugyanígy használhatjuk az `ioctl` hívásokat. A foglalat alapú `ioctl` hívások (amelyeknél a fájlleíró valamely foglalat kezelője) harmadik értéke gyakran egy `ifreq` (interface request) szerkezetmutató. Az `ifreq` szerkezet típusmegadását (type deceleration) a ***net/if.h*** fájlban találjuk.

Számos `ioctl`-csatoló leírása sajnos csak igen nehezen lelhető fel, ráadásul a hálózati csatolók eléréséhez legalább három eltérő API létezik. A programot eredetileg a MII (Media Independent Interface) módszerrel írtam. A cikk írása közben – ahogy frissebb rendszermag került a gépemre – felfedeztem, hogy az `ETHTOOL` eljárást is fel kell vennem. Miután az `ETHTOOL`-t beépítettem, átalakítottam a programot és minden csatolófelület-eljárást külön alprogramba raktam át. A módosított program kipróbálja az egyik módszert, és ha nem megy, sorra veszi a következőt.

A harmadik módszer még a MII API-nál is régebbi, nem is találkoztam még olyan géppel, amelyiknek erre lett volna szüksége, így ezt a kódot nem építettem be.

Az MII csatolófelület használatával kapcsolatos adatokhoz elsősorban ***Donald Becker*** `mii-diag` nevű programjának (➲ ftp.scyld.com/pub/diag/mii-diag.c) kódelemzésével jutottam hozzá, amit a Scyld Computing Corporation honlapján találtam. Ezen a helyen egy kiváló oldalra bukkan (➲ http://www.scyld.com/diag/mii-status.html), ahol az `ioctl` által esetlegesen visszaadott MII állapot-

szavakkal kapcsolatos ismeretekhez férhetünk hozzá. Most azonban az ETHTOOL csatolófelületre fogok összpontosítani, hiszen ez a frissebb módszer.
Az ETHTOOL API használatával kapcsolatos ismereteket szintén a különféle forráskódok elemzésével gyűjtöttem – volt eset, hogy magukba a hálózati csatolók forráskódjába kellett belenéznem; különösképpen sokat segített az ***eepro100.c***. Ezenkívül igen hasznos volt ***Tim Hockin*** elektronikus levele, amire google-izás közben akadtam rá.
A programomat úgy készítettem el, hogy hacsak másként nem utasítjuk, alapértelmezettként az eth0 csatolófelületet használja. A csatolófelület ID az ifname változóban tárolódik. Minthogy az általam használt ioctl-parancsok csak erre a csatolóra érvényesek, más csatolót használva valószínűleg „cannot determine status” (az állapot nem meghatározható) jelzést kapunk.
Az ioctl meghívása előtt szükségünk lesz a fájlkezelőre, ezért először is meg kell nyitnunk egy foglalatot:

```
int skfd;
if (( skfd = socket( AF_INET, SOCK_DGRAM, 0 ) )
      ➥< 0 )
   {
   printf("socket error\n");
   exit(-1);
   }
```

A szokásos „próbáljunk meg minden hibát ellenőrizni” C-kódolási stílust híven követve az utasítást egy if feltételbe helyeztem, így ha a foglalat nem működik megfelelően, egyszerűen jelzi a hibát, majd -1 értéket visszaadva kilép a programból. Céljainknak talán jobban megfelel, ha az állapot meghatározhatatlanságát (hibát) a kapcsolat hiányának és nem kapcsolatnak értékeljük, ezért ha megtaláltuk a kapcsolatot 0-t, ha nem, 1-et adunk vissza.
A meghajtó eléréséhez biztosított új ETHTOOL API segítségével sokkal könnyebb volt megállapítani a kapcsolat állapotát, mint a korábbi módszerrel. Az ETHTOOL csatolófelületben úgy oldották meg az ioctl-kezelést, hogy csak egyetlen ioctl parancs létezik, a SIOCETHTOOL (ami azt mutatja meg, hogy az ETHTOOL parancs következik), majd az azt követő adatblokk tartalmazza az ETHTOOL-felület pontos alparancsát.
A szabványos ioctl adatszerkezethez (type ifreq) két további elemre is szükségünk lesz: a csatolófelület nevére, amelyre a parancsot alkalmazni szeretnénk, és a szerkezet címére (type ethtool_value), amelyben az adott parancsot és a visszaadott adatot tároljuk.
A szerkezeteket és az egyéb adatokat (ideértve a használható parancsokat is) az ***ethtool.h*** fejlécfájlban találjuk. Nekem az ETHTOOL_GLINK parancsra volt szükségem, amelyet „get link status” leírással láttak el – ezt helyeztem az ***edata.cmd*** mezőbe: edata.cmd = ETHTOOL_GLINK;
A csatolófelület nevét és az edata szerkezet címét el kell helyeznünk ifreq szerkezetben:

```
strncpy(ifr.ifr_name, ifname,
➥sizeof(ifr.ifr_name)-1);
ifr.ifr_data = (char *) &edata;
```

Innen már csak az maradt hátra, hogy meghívjuk a ioctl rendszerhívást, ellenőrizzük a visszaadott értéket (hogy lássuk, vajon a parancstípus engedélyezve volt-e), majd megvizsgáljuk a visszaadott mutató által meghatározott szerkezetet, ahonnan végül megtudjuk, hogy a kapcsolat élő-e vagy sem:

```
if (ioctl(skfd, SIOCETHTOOL, &ifr) == -1) {
    printf("ETHTOOL_GLINK failed: %s\n",
           ➥strerror(errno));
    return 2;
}
return (edata.data ? 0 : 1);
```

Jelen esetben a kód 0-t ad vissza, ha a kapcsolat él, 1-et pedig akkor, ha nem; illetve 2-t ad vissza, ha nem lehet eldönteni, esetleg hiba történt. A kód segítségével a függvényt az ***rc.local*** állományomból hívhatom meg, és ilyen módon a csatoló indítása után csak akkor indítom a dhcpcd-t vagy próbálkozom az IP-cím megszerzésével, ha a rendszer valamilyen működő jelelosztó vagy kapcsolóeszközhöz (hub, illetve switch) csatlakozik. Az ***rc.local*** idevágó részei a következőek:

```
/root/sense_link/sense_link | logger
if /root/sense_link/sense_link > /dev/null; then
  logger "No link sense -- downing eth0"
  /sbin/ifconfig eth0 down
else
  logger "Sensed link - dhcping eth0"
  /sbin/dhcpcd eth0
fi
```

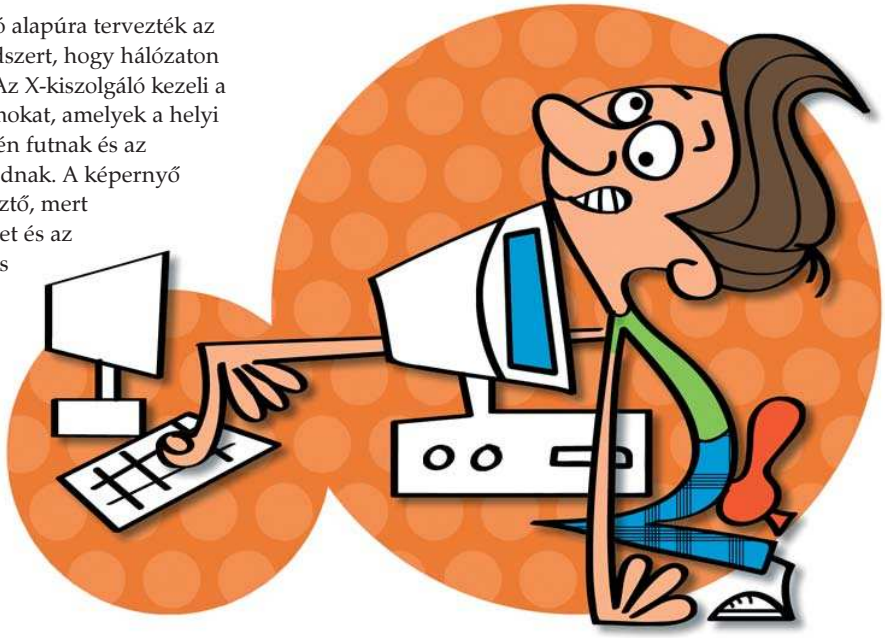
Először is a sense_link kimenetét elküldjük a rendszernaplóba. Azután, ha az eth0 eszközön nem találtunk kapcsolatot, netán nem tudtuk megállapítani, mi a helyzet, üzenetet küldünk a naplóba, majd leállítjuk az eth0-t. Amennyiben kapcsolatot érzékeltünk, végrehajtjuk a dhcpcd-t az eth0 csatolón.
Amióta ezt a rendszert összeraktam, ***rc.local*** állományom igen gyorsan lefut, akár hálózati kábel nélkül, akár működő DHCP-kiszolgáló mellett használom. Az egyetlen eset, amikor lényeges várakozási idővel kell számolnom, akkor jelentkezik, amikor olyan hálózatra csatlakozom, ahol nincsen működő DHCP-kiszolgáló.
802.11b típusú kártyámmal még nem próbáltam ki a kódot, pedig kíváncsi vagyok, érzékeli-e a kapcsolatot a DHCP-kiszolgálóra fellépés előtt – én ugyanis általában csak a PCMCIA kártyát dugom be, ha olyan helyen járok, ahol tudom, hogy van kiszolgáló. Az érdekelt társaságnak azonban érdekes kísérlet és hasznos kiegészítés lenne.

Linux Journal 2004. január, 117. szám

Lisa Corsetti
Jelenleg programmérnök, egyben pedig az Anteil, Inc., egy különféle ipari és kormányzati területekre szánt, testreszabott, webalapú alkalmazásokra összpontosító cég elnöke. Lisa a Drexel Egyetemen szerezte elektromérnöki és informatikai diplomáját.

Az Xwindow távoli elérése (1. rész)

A Unix világában szokványos feladat egy távoli gépet karakteres felületen keresztül elérni. Annak járunk utána, hogyan lehet ezt grafikusan is megtenni.



Kezdettől fogva kiszolgáló alapúra tervezték az X Window grafikus rendszert, hogy hálózaton keresztül is működjön. Az X-kiszolgáló kezeli a képernyőt (display) és a programokat, amelyek a helyi gépen vagy a hálózat másik gépén futnak és az ügyfelek (kliens) ehhez kapcsolódnak. A képernyő elnevezés itt egy kicsit megtévesztő, mert a képernyőn kívül a billentyűzetet és az egeret, illetve akár több kijelzőt is takarhat.
Induláskor a grafikus programok a `DISPLAY` környezeti változóból tudják, hogy melyik X-kiszolgálóhoz kell kapcsolódniuk, ezt a `-display` vagy a `--display` kapcsoló megadásával felül lehet bírálni. A `DISPLAY` környezeti változó ilyen módon épül fel:

<kiszolgálónév>:<képernyőszám (display)>.<kijelzőszám (screen)>

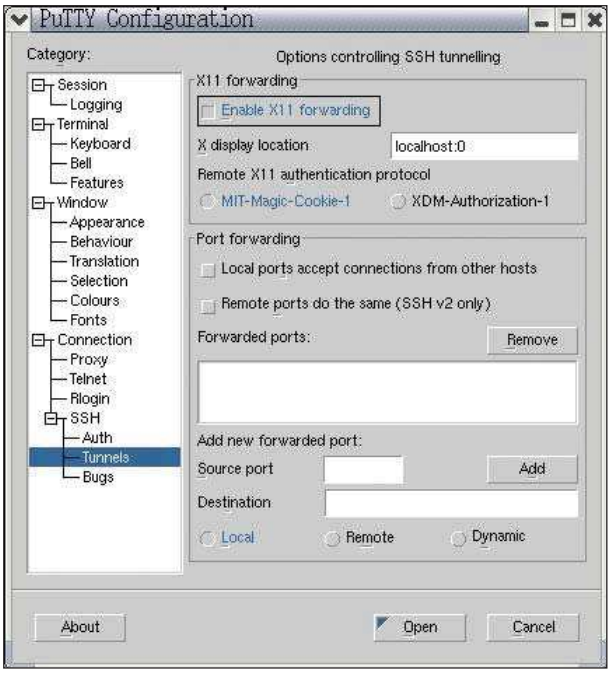
Ebből a kiszolgálónév elhagyható, ha az X-kiszolgálóval egy gépen belül vagyunk, továbbá a kijelzőszám is mellőzhető, amennyiben csak egyetlen monitorunk van. Ha például egy géphez egyetlen monitort használunk és egy X-kiszolgálót futtatunk, akkor a `DISPLAY` változó értéke `:0`. Ezt az `echo $DISPLAY` paranccsal tudjuk megnézni, és bash alatt a következő paranccsal állíthatjuk be

```
export
DISPLAY=<kiszolgálónév>:<képernyőszám>.<kijelzőszám>
```

A `:0` és a `localhost:0` között különbség van, mert az első esetben Unix protokollon, a második esetben pedig TCP-n keresztül tartja a kapcsolatot az ügyfél és a kiszolgáló bár mindkét hivatkozás a helyi gép 0-s képernyőjére vonatkozik. Megjegyzendő, hogy a virtuális munkaasztalok, illetve munkaterületek (Virtual desktops/Worksplaces) ablakkezelő-szintű fogalmak, ezért itt nem lehet hivatkozni rájuk.
Írásom példáiban abból indulunk ki, hogy az otthoni gépünkön akarjuk megjeleníteni a távoli munkagépen futó alkalmazásokat. Az otthoni gép legyen az ***otthonigep.hu*** és fusson egy X-kiszolgáló a `:0`-s képernyőn, a munkahelyi gép pedig legyen ***munkagep.hu***.
Ahhoz, hogy a munkagépen indított program grafikus felülete az otthoni gépen jelenjen meg, mindössze ennyi a teendőnk a munkagépen:

```
export DISPLAY=otthonigep.hu:0
mozilla&
```

Ahhoz, hogy akárki ne tudjon hozzáférni képernyőnkhöz, ablakokat pakolni rá és lehallgatni a billentyűzetünket, szabályoznunk kell a hozzáféréseket. Ezt kétféle módon tehetjük, az `xhost`-tal gépek (host) szintjén és az `xauth` paranccsal felhasználói engedélyeket készítve. Ezt a két módszert akár vegyesen is használhatjuk.
Az `xhost` az egyszerűbb, ezzel gépeket engedélyezhetünk, illetve tilthatunk.



1. kép A Putty beállítása

Az `xhost +localhost -crackers.hu` bárkinek engedélyezi a helyi gépről és tiltja a ***crackers.hu***-ról az X-kiszolgálóhoz való kapcsolódást. A már létrejött kapcsolatokat az új szabályok nem befolyásolják. Alapértelmezés szerint mindenhonnan tiltva van a csatlakozás. Ha kapcsolók nélkül indítjuk az `xhost` parancsot, akkor az érvényben lévő szabálylistát kiírja. Mivel az engedélyezett gépekről még bármelyik felhasználó csatlakozhat a képernyőnkhöz, az `xhost`-ot csak biztonságos helyi hálózatokon ajánlatos használni. Az `xhost +` mindenhonnan engedélyezi a kapcsolódást, ezért ezt csak a kipróbálás ideje alatt használjuk.
Az `xauth` utasítással ennél finomabb szabályozáshoz juthatunk. Ezzel a programmal úgynevezett `MIT-MAGIC-COOKIE`-t állíthatunk elő, ami tulajdonképpen hozzáférési jegyként működik. Először az `xauth`-tal létre kell hozni egy ilyen jegyet, majd a távoli géphez el kell juttatni, és az ottani X jogosultsági listájába (általában ***~/.Xauthority***) bele kell írni, innentől kezdve a kapcsolódáskor az X-es programok önműködően használják az azonosításhoz. Például a `$DISPLAY`-hez tartozó jegy kinyerése a következő utasítással történik:

```
xauth extract jegy.txt $DISPLAY
```

Ezt követi a jegy felírása a listára a távoli gépen:

```
xauth merge jegy.txt
```

Ha a távoli gépre `rsh`-val lehet csatlakozni, akkor ugyanez egy sorba írandó:

```
xauth extract - $DISPLAY | rsh dolgozo@munkagep.hu
xauth merge -
```

Amennyiben levélben vagy másolás–beillesztéssel szeretnénk átvinni a jegyet a távoli gépre, akkor `extract` és `merge` helyett használjuk a `nextract` és az `nmerge` utasításokat, ezek segítségével számformátumban kezeli az amúgy bináris jegyeket. Az `xauth` módszer jellegéből adódóan nincsen lehetőségünk lekérdezni, hogy kinek adtunk jegyeket, de azt igen, hogy a miénk hová érvényes: `xauth list`. Ha még nagyobb biztonságra törekszünk, akkor akár a jegy érvényességét is meghatározhatjuk.
Figyeljünk arra, hogy némely Linux-változatban (például Debian) biztonsági okokból alapbeállításban tiltott az `X tcp forwarding` az X protokoll TCP-n keresztül való továbbítása, azaz csak a helyi gépről lehet csatlakozni az X-kiszolgálóhoz! Ha `gdm`-et (GnomeDisplayManager) használunk, ennek engedélyezéséhez írjuk át a következő sorokat a */etc/gdm/gdm.conf*-ban:

```
[server-Standard]
name=Mezei X kiszolgáló
#command=/usr/X11R6/bin/X -deferglyphs 16
➥-nolisten tcp -audit 0
command=/usr/X11R6/bin/X -deferglyphs 16 -audit 0
flexible=true
```

Ezentúl feltételezzük, hogy az otthoni gépen engedélyezett az `X tcp forwarding` és az `xauth`-tal (vagy `xhost`-tal) a jogosultságokat is beállítottuk.

Az LBX használata

Az X protokoll továbbítása nagy adatátvitellel jár, ez különösen érzékelhető, ha nem a helyi hálózaton keresztül kapcsolódunk, ezért hasznos lehet tömöríteni az adatforgalmat. A következő programok erre használhatóak.
Az LBX (Low Bandwidth X) egy X proxy, az X-es programok ezen keresztül veszik fel a kapcsolatot az X-kiszolgálóval, ekképpen már menet közben tömöríteni tudja az adatforgalmat, amit az X-kiszolgálóban lévő LBX kiterjesztés csomagol ki. (Ez alapértelmezetten része az X-kiszolgálóknak, és az `xdpyinfo`-val ellenőrizhető.) Használatához a munkagépen futtatni kell az `lbxproxy` programot:

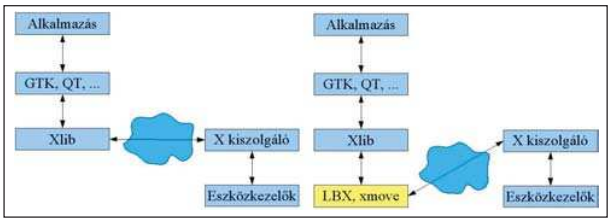
```
lbxproxy -display DISPLAY1 DISPLAY2
```

Ebben a `DISPLAY1` az a képernyő, amelyen meg akarjuk jeleníteni a programunkat, a `DISPLAY2` pedig az „álképernyő”, amihez majd az alkalmazások kapcsolódnak. Miután bejelentkeztünk a munkagépre, például a következő utasítást adjuk ki:

```
lbxproxy -display otthonigep.hu:0 :1&
export $DISPLAY=:1;
gnome-session&
```

A DXPC működése

A DXPC (The Differential X Protocol Compressor) hasonlóan működik, mint az LBX, de mivel ez nem az X-kiszolgáló szerves része, ezért mind a két oldalon futtatnunk kell egy proxyt. Ugyanakkor valamivel jobban tömörít, mint az LBX. Alapértelmezésben a kiszolgáló (amelyen az X fut) kapcsolódik az ügyfélhez (amin a programok futni fognak), de ez a `-w` kapcsolóval megcserélhető. Használata a munkagépen a következőképpen fest:



2. kép Az XWindow rendszer felépítése

```
export DISPLAY=otthonigep.hu:0
dxpc -f
export DISPLAY=:8
```

Alapértelmezésben a `:8`-as az álképernyő, de a `-d` kapcsolóval felülbírálható. A `-f` kapcsoló a háttérben való futtatáshoz szükséges. Az otthoni gépen a következő utasítást írjuk be:

```
dxpc -f munkagep.hu
```

Ezután már indíthatjuk is a programokat a munkagépen.

OpenSSH

Ha a `CX` kapcsolókat használjuk, az OpenSSH-val is megjeleníthetünk a távoli gépen futtatott alkalmazásokat a helyi képernyőn. Az `X` kapcsoló továbbítja az X protokollt, a `C` pedig tömöríti az adatforgalmat. Előfeltétele, hogy a távoli gépen (példánkban a munkagépen) az SSH démonnak engedélyezve legyen az `X11Forwarding`. Ennek megvalósításához a */etc/ssh/sshd_config*-ba írjuk be:

```
X11Forwarding yes
```

Az `ssh -CX <felhasználónév>@<kiszolgálónév>` parancs után máris indíthatjuk a grafikus programokat. Például az otthoni gépen egy `xterm`-ben:

```
ssh -CX dolgozo@munkagep.hu
gnome-chess&
```

A Putty használata

A Putty SSH-ügyfél is használható az X-felület továbbítására. Ezt grafikusan a ***Connection/SSH/Tunnels/X11Forwarding*** pontban állíthatjuk be.
Ezeknek az SSH-programoknak az az előnye, hogy a `DISPLAY` környezeti változó, valamint a hozzáférési jogosultságok beállítását önműködően elintézik számunkra. Továbbá nagyon fontos, hogy egyúttal titkosítják is a kapcsolatot. Igaz, éppen ezért jóval nagyobb erőforrást is igényelnek.

Az xmove

Tegyük fel, hogy otthon két gépünk van, és a munkahelyi gépen OpenOffice.org fut, amit az egyik, otthoni gépen jelenítünk meg és használunk. De egy idő után át kell ülnünk a másik otthoni gépünkhöz. Ekkor sajnos mindent mentenünk kell, majd a másik gépen szükségszerűen újra meg kell nyitni mindent. Az `xmove` ezen a gondon segít, ugyanis befészkeli magát az X-kiszolgáló és az ügyfél közé, a kiszolgálónak ügyfélként látszik, az ügyfél számára pedig kiszolgálóként. Így lehetőséget nyújt az egyes alkalmazások, illetve akár egész munkamenetek megjelenítésének átküldésére az egyik X-kiszolgálóról a másikra.

```
xmove -server otthonigep.hu:0 -port 1
export DISPLAY=localhost:1
ooffice&
```

Eddig semmi különös nincs a kódban, de következzen egy ügyes trükk; az `xmovectrl` program segítségével küldjük át az ablakokat a másik otthoni gépre:

```
export DISPLAY=munkagep.hu:1
xmovectrl -moveall otthonigep2.hu:0
```

Ezután állítsuk be, hogy minden új X-es program itt jelenjen meg:

```
xmovectrl -setdefaultserver otthonigep2.hu:0
```

Ha egy időre sehol sem akarjuk megjeleníteni az ablakokat, a következőket írjuk be:

```
xmovectrl -moveall -suspend
```

Fontos, hogy ha `xmove`-ot használunk, a `DISPLAY` változóban a gépnév is mindig szerepeljen. Az `xmove` használatának előfeltétele, hogy az X-kiszolgálók színmélysége megegyezzen.

A tűzfal

Ha tűzfalon keresztül szeretnénk csatlakozni, akkor a következő kapukat kell megnyitni (engedélyezni).

- X-továbbítás: TCP/6000 + képernyőszám (az LBX, `xmove` is ezt használja).
- DXPC: TCP/4000 a kiszolgáló alapértelmezetten kapcsolódik az ügyfélhez, kivéve, ha a `-w` kapcsolót használjuk.
- SSH: TCP/22
- XDMCP: UDP/177
- VNC: TCP/5900 + képernyőszám, vagy ha böngészőből szeretnénk elérni, akkor TCP/5800 + képernyőszám.

Sorozatunk következő részében innen folytatjuk.

Radics László
Az ELTE-n tanul programtervező matematika szakon. Reméli, hogy egyszer végleg levizsgázik analízisből. Szabadidejében szívesen kirándul, sakkozik és kártyázik a barátaival. A viharokat különösen szereti, ezért is lett a művészneve „Garabonciás".

KAPCSOLÓDÓ CÍMEK

Az LBX oldala
➲ http://ftp.xfree86.org/pub/X.Org/pub/R6.6/lbxproxy
A DXPC ➲ http://www.vigor.nu/dxpc
A Putty oldala
➲ http://www.chiark.greenend.org.uk/~sgtatham/putty/

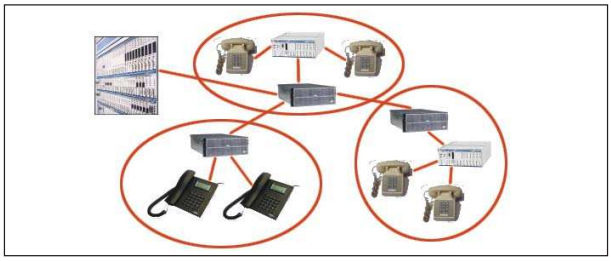
Egy nyílt forráskódú PBX-rendszer, az Asterisk

Használj egyetlen rendszert az IP alapú és a hagyományos telefonhang továbbítására, a hangposta vezérlésére, valamint írj CGI-szerű alkalmazásokat a telefonhasználók számára.

Házi telefonalközpontot (Private Branch eXchange, PBX) kellene felszerelni irodánkban? Esetleg egy hangposta rendszert szeretnénk üzembe helyezni otthoni linuxos gépünkön? Mit szólnánk egy önműködő hangválaszrendszerhez (Interactive Voice Response, IVR) otthonunk automatizálásához? Esetleg az interneten keresztüli telefonálás lehetősége (VoIP) sem jönne rosszul? Hogyan valósíthatnánk meg mindezt? Az Asterisk egy rendkívül érdekes és hatékony, a GPL felhasználási szerződés alatt terjesztett linuxos PBX-rendszer, melynek segítségével áthidalható az otthon is megtalálható hagyományos telefon és a világhálón keresztül történő hangátvitel közötti szakadék. Az Asterisk ezen felül számos más szolgáltatással is bír, amelyek valóban vonzó választássá teszik. Írásunkban megismerhetünk néhányat e tulajdonságok közül, és elég adatot szerezhetünk be a programról ahhoz, hogy mindenféle különleges eszköz beszerzése nélkül belevághassunk a dologba.

A háttér

Az Asterisk, a Digium által támogatott nyílt forrású projekt, melynek elsődleges gazdája ***Mark Spencer***, de számos foltot a közösség készített el hozzá. Írásom időpontjában még kizárólag Intel processzorokon futó Linuxon működik, bár nemrégiben már sikerült bizonyos eredményeket elérni a PowerPC-n futó Linux terén is, és erőfeszítéseket tettek az Asterisk BSD-re történő átültetésére is. A Digium emellett különböző eszközöket is kínál, amelyek együttműködnek az Asteriskkel. Ezek az eszközök olyan PCI-kártyák, amelyek a szabványos analóg telefonvonalat kötik össze a számítógépünkkel. A program más eszközöket is támogat, ilyenek például a Dialogic vagy a Quicknet termékei. Az Asterisk saját, IAX elnevezésű VoIP protokollal rendelkezik, de támogatja a SIP és H.323 protokollokat is. Ezzel el is jutottunk az Asterisk egyik legnagyobb előnyéhez: a különböző technológiákat egy környezeten belül össze tudja kapcsolni. Az Asterisken keresztül egy rendszerré köthetjük össze az IAX, SIP, H323 és a hagyományos telefonunkat (lásd az *1. képen*).
A fejlesztő ki is terjesztheti az Asterisket a C API használatával, vagy az AGI-k segítségével, ezek a CGI-parancsfájlok megfelelői. AGI-kat bármilyen nyelven írhatunk, ezek



1. kép Az Asterisk képes a hagyományos telefonvonal és a különféle VoIP szabványok szerinti vonalak összekapcsolására

futtatása pedig külső folyamatként zajlik. Ez az Asterisk képességbővítésének legegyszerűbb és legrugalmasabb módja (lásd az *1. listát*).

Vágjunk bele!

Egy ideje már nem készült új hivatalos kiadás, de szó van róla, hogy nemsokára megjelenik. Pillanatnyilag az Asterisk megszerzésének legjobb forrása a CVS:

```
export CVSROOT=\
:pserver:anoncvs@cvs.digium.com:/usr/cvsroot
cvs login (a jelszó: "anoncvs")
cvs co asterisk
```

Ha Digium gyártmányú PCI-kártya használatát tervezzük, a `zaptel`-re is szükségünk lesz. Ha az összekapcsolhatóság is szerepel célkitűzéseink között, akkor a `libpri`-t szintén le kell töltenünk.
Beállító parancsfájl nincs, egyszerűen a `make` parancsot kell használnunk. Az Asterisk sikeres lefordításához szükségünk van a `readline`, OpenSSL csomagokra és a 2.4-es rendszermagra a magforráskódokkal együtt:

```
cd asterisk
make clean install samples
```

Ez lefordítja az Asterisket, telepíti és emellett a minta-beállításfájlokat is felpakolja. Az utolsó rész felülírja a már létező beállításfájlokat, ezért ezt vagy ugorjuk át, vagy készítsünk biztonsági másolatot azokról a beállításfájlokról, amiket

meg szeretnénk tartani. A `zaptel` vagy ISDN használata esetén ezeket még az Asterisk fordítása előtt fordítsuk le. Az Asterisk telepítése alapértelmezetten a */usr/sbin/* könyvtárba történik, a beállításfájlok pedig a */etc/asterisk/* könyvtárba kerülnek. A hangposta üzenetei a */var/spool/asterisk/voicemail/* könyvtárban tárolódnak. A számlázás CDR-jeit (Call Detail Record, azaz hívásadat rekord) és a naplózás fájljait a */var/log/asterisk/* elérési útvonalon találjuk meg.
Az Asterisk indításához az `asterisk` parancsot kell kiadnunk parancssorból, ehhez a kipróbálás szakaszában nem árt hozzáadni a `-vvvc` kapcsolókat is. A `-vvv` kapcsoló a működésről a legtöbb üzenetet adó üzemmódot kapcsolja be, a `-c` kapcsoló pedig egy konzolképernyőt ad, amelyen keresztül kapcsolatot tarthatunk az Asterisk munkafolyamatával, például vezérlésre vagy állapotra vonatkozó parancsokat küldhetünk neki.
Az Asterisk tevékenysége és működőképessége több különböző beállításfájlon alapul. Ezek közül ebben a cikkben hármat vizsgálunk meg, de még továbbiak is léteznek. Az Asterisket beállítjuk, hogy a felhasználók az IAX protokollon keresztül hívhatják egymást, továbbá üzembe helyezzük a hangpostát, melynek segítségével a felhasználók lehetőséget kapnak hangüzeneteik kezelésére.

A tárcsázási eljárások

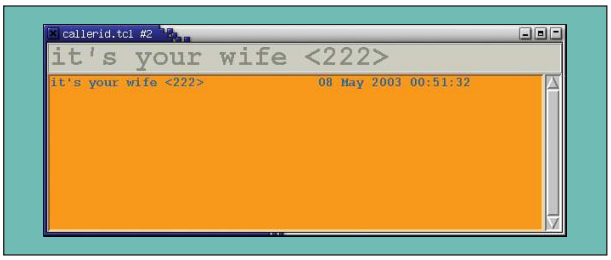
Mielőtt még nekilátnánk az Asterisk üzembe helyezésének, átfogó képpel kellene rendelkeznünk a tárcsázási eljárásokról (dialplan). Habár rugalmas és hatékony eszközről van szó, de könnyen bele is kavarodhatunk. A tárcsázási eljárás a számok értelmezésének meghatározására, valamint az útvonal kijelölésére használatos, s ebből kifolyólag az Asterisk lelkét képezi. A tárcsázási eljárás környezeteket (context) határoz meg, amik azoknak az úgynevezett kiterjesztéseknek (számjegyminták) a tárolói, amelyek meghatározott feladattal bírnak. Például előfordulhat, hogy az irodánk vagy a lakótársaink számára szeretnénk egy bizonyos környezetet meghatározni, amely számukra adott tárcsázási jogosultságokat teremt. Ugyanakkor meghatározhatunk egy külsősök vagy vendégek számára létrehozott környezetet is, ami csak bizonyos korlátozásokkal engedélyezi a tárcsázást, például tiltja a távolsági hívásokat. A környezetek neveit szögletes zárójelek ([]) között kell feltüntetni. Az adott környezetekhez rendelt kiterjesztések a név után írandók. Az egyes kiterjesztések elsőbbséggel (priority) rendelkezhetnek. A hívási folyamat végrehajtása sorrendben történik egészen addig, míg egy alkalmazás `-1` értéket nem ad vissza, ekkor a hívás félbeszakad, vagy az alkalmazás átirányítja a folyamatot. A kiterjesztés bejegyzésének formátuma a következő:

```
exten =>
<kiterjesztés>,<prioritás>,<alkalmazás(paraméterek)>
```

Íme néhány példa:

```
exten => 9911,1,Wait(1)
exten => 9911,2,Dial(Zap/1/${EXTEN:1})
```

A kiterjesztést az `exten=>` kifejezés jelöli. Ebben az esetben a 9911 a kiterjesztés, az 1 és a 2 az elsőbbség vagy lépés száma (ezeknek sorrendben kell egymás után következniük); a Wait és a Dial pedig az alkalmazások.



2. kép Grafikus program mutatja a beérkező hívások azonosító adatait

Az Asterisk külön programokat használ a kiterjesztések egyes lépéseinek végrehajtásához. Ezekről a programokról leírást az Asterisk konzolján keresztül kaphatunk. A `show applications` parancs listát ad a támogatott programokról, a `show application <program neve>` parancs pedig a megadott program használatához nyújt segítséget.
A kiterjesztések használhatók mind a kimenő, mind a bejövő hívószámra, ezzel a hívások feldolgozásakor nagyfokú rugalmasság érhető el. Sablonok használatára is lehetőség nyílik, egy aláhúzás jel (_) vezeti be őket:

- N: egyetlen, 2 és 9 közötti számjegy.
- X: egyetlen, 0 és 9 közötti számjegy.
- [12-4]: bármilyen, a zárójelek közt lévő szám.
- .: helyettesítő karakter.

Például a _NXX5551212 bármilyen adatszámra illeni fog, függetlenül a körzetszámtól.
A kiterjesztés bármilyen alfanumerikus karakterlánc lehet. Lássunk néhány beépített különleges karaktert:

- s: ha nem érkezik be tárcsázott szám, akkor itt kezdődik, mint egy analóg vonalról érkező bejövő hívás.
- t: időtúllépés esetén használatos.
- i: tiltott számok tárcsázása esetén használatos.
- o: operátori kiterjesztés.
- h: felfüggesztő kiterjesztés.

IAX-felhasználók létrehozása

Az első fájl, amit létrehozunk, az ***iax.conf*** (lásd a *2. listát*). Ez a fájl vezérli az IAX protokoll működését, és a protokoll felhasználóit határozza meg. A protokollnak két változata létezik, a régi az IAX, az új pedig az IAX2 nevet viseli.
A beállításfájl első szakasza az általános rész, amely az IAX protokoll jellemzőit határozza meg. A felsorolásban négy szerepel, de ennél több is megadható. A `port` érték annak a kapunak a száma, amelyen az IAX a párbeszédet folytatni fogja. Ez alapértelmezetten az 5036, így akár el is hagyható. A `binaddr` értéket arra használhatjuk, hogy az Asterisk egy bizonyos IP-címhez kapcsolódjon; például több hálózati kártyával felszerelt gép esetén. A 0.0.0.0 `binaddr` érték minden IP-címhez csatlakozni próbál. Az `amaflags` és `accountcode` értékeket a CDR-ek használják. Amennyiben az általános részben adjuk meg, akkor alapértelmezett értékként kerülnek felhasználásra, de lehetőség nyílik felhasználónkénti megadásra is. Az `amaflags` lehetséges értékei a `billing`, `documentation`, `omit` és a `default`. Az `accountcode` tetsző-

1. lista Példa a Caller ID AGI parancsfájlra

```
#!/bin/sh
# \
exec tclsh "$0" ${1+"$@"}

set port 10000
set hosts [list 192.168.123.166 192.168.123.168]

##
## Adat küldése a gazdagépnek
##
proc sendInfo {ip_ port_ callerid_} {

  if {[catch {socket $ip_ $port_} sock]} {
    return
  }

  fconfigure $sock -buffering line
  puts $sock $callerid_
  close $sock

  return
}

##
## Az összes változó megszerzése az stdin-ről;
## ezek "agi_" kezdetűek; egy tömb feltöltése
## az értékekkel
##

while {[gets stdin l] > 0} {
  if {[regexp {^agi_([\w]+):[\s]+(.*)} $l ->
➥k v]} {
    set AGI($k) $v
  }
}

##
##       A callerid adatainak elküldése
##       az összes felsorolt gépre
##
foreach H $hosts {
  sendInfo $H $port $AGI(callerid)
}
```

leges értéket vehet fel. A mostani beállításnál a `home` kerül a hálózaton belüli felhasználókhoz, és `externa`l a hálózatomon kívüli felhasználókhoz rendelt érték. Sok más értéket kihagytunk, de ezek nagy része a teljesítménnyel kapcsolatos.
A hátralévő szakaszok a felhasználók meghatározásai. Nekem három felhasználóm van: brett, maria és nico. A típusmegadáskor három lehetséges értéket adhatunk meg: a `peer` minden hívást fogadhat, a `user` hívásokat kezdeményezhet, a `friend`-nek pedig mindkettőre lehetősége nyílik. Én mindegyiket `friend` típusúként adtam meg. Minden gépet dinamikusnak (dynamic) adtam meg, de ha valamelyik gépünk állandó IP-címmel bír, állandó (static) tulajdonságúként határozhatjuk meg. A `secret` az a jelszó, amit a felhasználónak az Asterisk kiszolgálóhoz való csatlakozáskor kell megadnia. Ebben a fájlban két környezetet adok a felhasználók számára: a `[cg1]` és a `[cg2]` nevűeket. Ezeket részletesen az ***extension.conf*** fájl magyarázata során fogom ismertetni, de lényegében e környezetek adják a felhasználó számára a tárcsázási jogosultságokat.

A hangposta beállítása

A következő fájl a ***voicemail.conf*** (*3. lista*). Ennek is van egy általános része, amely a hangposta általános vagy átfogó jellemzőivel foglakozik. Az első érték, a `format`, az üzenetek hangformátumait sorolja fel. A következő két értéket az elektronikus levélben történő értesítés használja: a `serveremail` a forráscím (mezőből), az `attach` pedig arra utasítja az Asterisket, hogy az üzenetet csatolja a levélhez. Példánkban nem szeretnénk üzenet csatolását, ezért ebben az esetben elhagyunk néhány értéket.
Az `<mbox>` az a szám, amelyen a felhasználó számára az üzenetek menthetők és elérhetők. Ezt az ***extension.conf*** fájlban is használjuk a hívási folyamat megfelelő hangpostafiókba való irányítására. A `<passwd>` érték megadása az üzenetek lekérdezésekor szükséges, a `<name>` pedig a felhasználó neve. Az `<email>` és `<pager>` értékek olyan elektronikus címek, amelyeket a levél érkezéséről tájékoztató üzenetek küldésére használ a rendszer. A `<pager>` címre küldött üzenet rövid, mert kisebb eszközökön (személyhívó vagy mobiltelefon) is olvashatónak kell maradnia. Sok mobiltelefon- és személyhívó-szolgáltató rendelkezik olyan elektronikus-levélátjárókkal, amelyek képesek az üzeneteket az eszközre továbbítani.

A kiterjesztések meghatározása

A jelenleg vizsgált fájlok közül az utolsó az ***extension.conf*** fájl (lásd a 4. listán, mely az 59. CD Magazin/Asterisk könyvtárában található). Ez az egyik legbonyolultabb fájl, mivel a tárcsázási eljárásokat is tartalmazza. A példánkban látható tárcsázási eljárás képességeihez viszonyítva meglehetősen egyszerű. A fájl egy ***general*** és egy ***global*** nevű szakasszal rendelkezik. Az általános szakasz hasonlít a korábbi fájlok általános szakaszára, vagyis általános értékeket határoz meg. Ebben a példában nem adtam meg általános értékeket. Az átfogó szakasz a teljes körű változók meghatározására használatos. Ezek a változók a tárcsázási eljárásokban a `${VÁLTOZÓ}` utasításforma használatával érhetők el. Én egy változót határoztam meg `TIMEOUT` néven a válaszidő-túllépésére. Az olyan beépített változók, mint a `CONTEXT`, `EXTEN` és `CALLERID` szintén használhatók a tárcsázási eljáráson belül. A többi szakaszok mind környezetmeghatározások. A környezet egyszerűen számjegyminták egy csoportja. A példa esetében több környezetet is megadtam, ezek mindegyike valamilyen tárcsázási esetet – hangposta, `iax` és munkaidő utáni hívás – határoz meg. Ezeket úgy kell elképzelni, mint mini tárcsázási terveket. Ezek után két olyan környezetet adtam meg, amelyeket a felhasználókhoz rendeltem. E környezetek öröklik a már korábban meghatározott, az `include` kulcsszó használatával megadott környezetek adottságait.
Az első környezet a `voicemail`, azoknak a számjegymintáknak a listája, amelyekkel a felhasználók érhetik el hangpos-

2. lista Az iax.conf fájl tartalma

```
[general]
port=5036
bindaddr=0.0.0.0
amaflags=default
accountcode=home

[brett]
type=friend
host=dynamic
secret=brettsecret
context=cg1
callerid="brett <111>"

[maria]
type=friend
host=dynamic
secret=mariasecret
context=cg1
callerid="maria <222>"

[niko]
type=friend
host=dynamic
secret=nikosecret
context=cg2
accountcode=external
callerid="Niko <333>"
```

tájukat. A felhasználók tárcsázzák a 6245 számot, mire a VoicemailMain2 program bekéri tőlük a postafiókjuk számát és a jelszót. A felhasználók itt a postaládájukban lévő üzenetekkel kapcsolatos teendőiket intézhetik el (meghallgathatják, törölhetik).

Az `iax`-környezet az IAX-felhasználók közti PBX-tárcsázásra ad lehetőséget. Minden felhasználó számára különböző kiterjesztéseket határoztam meg. A felhasználó nevét (maria) tartalmazó kezdés átirányít a kiterjesztés számának bejegyzésére. Az 111 kiterjesztést szintén a hívóazonosítóval (callerid) egyeztettem. Amennyiben az azonosító megfelel, megváltoztatom a callerid nevét, így ennek viszonylagos jelentése van. Például, ha a tárcsázott kiterjesztés a 111, és a hívóazonosító 222, akkor a callerid nevét „it's your wife!" („a feleséged hív!" szövegre cserélem. Ez az üzenet jelenik meg mindannyiszor, ha a feleségem hív.

Az utolsó számminta környezet a késői órákban érkező hívások esetén használatos. Mivel nem szeretném, hogy külső hívások zavarjanak éjszaka, bármilyen tárcsázott számra fennáll az egyezés (_.). Vár egy másodpercig, fogadja a hívást, majd egy tárolt üzenetet játszik le, így a hívó kiválaszthatja, hogy kinek szeretne üzenetet hagyni („for brett, press 1 –, ha brettnek, akkor nyomja meg az 1-es gombot"). Ha a hívó megnyomja az 1-est, a hívás az `1,1,Voicemail2(111)` bejegyzésen folytatódik, amely a felhasználót az 111 postafiókba irányítja. Ez egy egyszerű példa arra, miként hozhatunk létre egy önműködő hangválaszrendszert.

3. lista A voicemail.conf fájl tartalma

```
[general]
format=gsm|wav49|wav
serveremail=asterisk
attach=no
maxmessage=180
maxgreet=60

;
;  Voicemail box definitions.
;  mbox# => password,name,email,pager/mobile
;
[cg1]
111 => 1111,Brett,brett_schwarz@yahoo.com
222 =>
2222,Maria,maria@foo.com,4255551212@mob.net
```

A `[cg1]` és `[cg2]` környezetek olyan szolgáltatásokat foglalnak magukban, amiket más környezetekben is megadtam. Ez lehetővé teszi, hogy könnyedén létrehozhassak különböző felhasználói csoportokat. Például a `[cg1]` minden általam korábban már meghatározott képességgel rendelkezik, viszont a `[cg2]` csak `iax` képességekkel, és a munkaidő utáni órákban hívás esetén a hangpostára kerül átirányításra. Az Asterisk tárcsázási eljárásai rugalmasságának kihasználásával hatékony tárcsázási lehetőségeket határozhatunk meg. A bemutatott példa csak felvillantja a benne rejlő lehetőségeket. A tárcsázási eljárásokat a makrók használatával szintén egyszerűbbé tehetjük, de ezt gyakorlásképpen meghagyom az olvasónak.

Az AGI használata

Az ***extensions.conf*** fájlban egy ***callerid.agi*** bejegyzés meghív egy AGI-parancsfájlt. Ez egy egyszerű példa az AGI-felület bemutatására. A parancsfájl a ***/var/lib/asterisk/agi-bin/*** könyvtárban helyezkedik el, és az Asterisk hívja meg külső eljárásként. Az AGI és az Asterisk az `stdin`, `stdout` és `stderr` csatornákon tartják egymással a kapcsolatot. Az AGI-nak történő értékátadás az `stdin` csatornán keresztül történik, az AGI pedig az `stdout` csatornán keresztül ad vissza adatokat az Asterisknek. Az Asterisk konzoljának szánt üzenetek az `stderr` csatornára kerülnek. Két érték mindenképpen átadásra kerül az AGI-nak: az AGI teljes elérési útvonala és azok a tulajdonságok (argumentum), amelyeket az AGI az `exten` bejegyzésen keresztül kap meg. Az AGI beolvassa a `callerid`-t, majd átküldi a másik gépen futó grafikus felhasználói programnak. Ez a grafikus felhasználói program a weboldalamról szerezhető be (lásd a 2. *képet*). Az adatok kikereséséhez AGI-parancsfájlok is használhatók. Amennyiben egy adatbázisból a hívóval vagy a felhasználóval kapcsolatos adatot kell kikeresnünk, használhatjuk az AGI felhasználói felületét is.

Hívjunk fel valakit

Mihez kezdhetünk az elvégzett beállításokkal? Miután létrehoztuk a fenti beállításfájlokat és elindítottuk az Asterisket (`asterisk -vvvc`), megpróbálkozhatunk a híváskezde-

ményezéssel. Jelen pillanatban az IAX ügyfélprogramok elérhetősége korlátozott. A SIP-ügyfelek, amilyen a `kphone` és az `xten` és a Cisco, SNOm és egyéb gyártók eszközei elérhetőek és együttműködnek az Asteriskkel, de ebben a cikkben az IAX használatára összpontosítottam. A legrégebbi ügyfélprogram a Digium által kifejlesztett Gnophone. Jelenleg több felületen futó, windowsos ügyfélprogramon is dolgoznak. Hozzáférhető még a SourceForge `tel` projektjéhez tartozó ügyfélprogram is. Ennek az ügyfélprogramnak módosítottam a felhasználói felületét. A program még próbaállapotú, de már működik. Ezt az ügyfélprogramot használtam, amikor Németország (***Reinhard Max***), Ausztrália (***Steve Landers***) és az Egyesült Államok (jómagam) között hoztam létre összeköttetést. Bármelyik ügyfélprogram mellett döntünk is, egy felhasználói nevet, jelszót és egy környezetet meg kell határoznunk minden olyan Asterisk-kiszolgáló esetében, amelyhez csatlakozni szeretnénk. Ezután bárkit hívhatunk, aki az ***iax.conf*** fájlunkban megadásra került (feltéve, hogy a tárcsázási eljárás beállítása megfelelő). Amennyiben a feleségemet szeretném hívni, egyszerűen a 222 számot tárcsázom, vagy beírom a maria szót (mivel a tárcsázási eljárásban ezt adtam meg). Ha a hangpostámban lévő üzeneteket szeretném ellenőrizni, a 6245 számot kell tárcsáznom.

Összegzés

Csak néhányat mutattam be az Asterisk képességei közül, de úgy gondolom, a cikk alapján az olvasó képet alkothat a programban rejlő lehetőségekről. Az Asterisk egyaránt jól használható a kisebb és a nagy, összetettebb rendszerek létrehozására. Például az egymástól távol lévő Asterisk kiszolgálókat az IAX protokollon keresztül összekapcsolva virtuális telefonalközpont létrehozására is lehetőségünk nyílik. Mivel az Asterisk Linuxon fut, meglévő eszközeinket a csatlakozófelületek létrehozására és az Asterisk irányítására használhatjuk. Például lehetőségünk van a CDR-ek, a beállításfájlok és a hangposta weben keresztül történő elérésére. Tulajdonképpen az Asterisk mellé egy CGI parancsfájlt is kapunk, amely lehetővé teszi hangüzeneteink elérését egy böngészőprogram segítségével. Mindenkinek javaslom, hogy fedezze fel az Asteriskben rejlő további lehetőségeket, és minél jobban hasznosítsa ezt a hatékony eszközt.

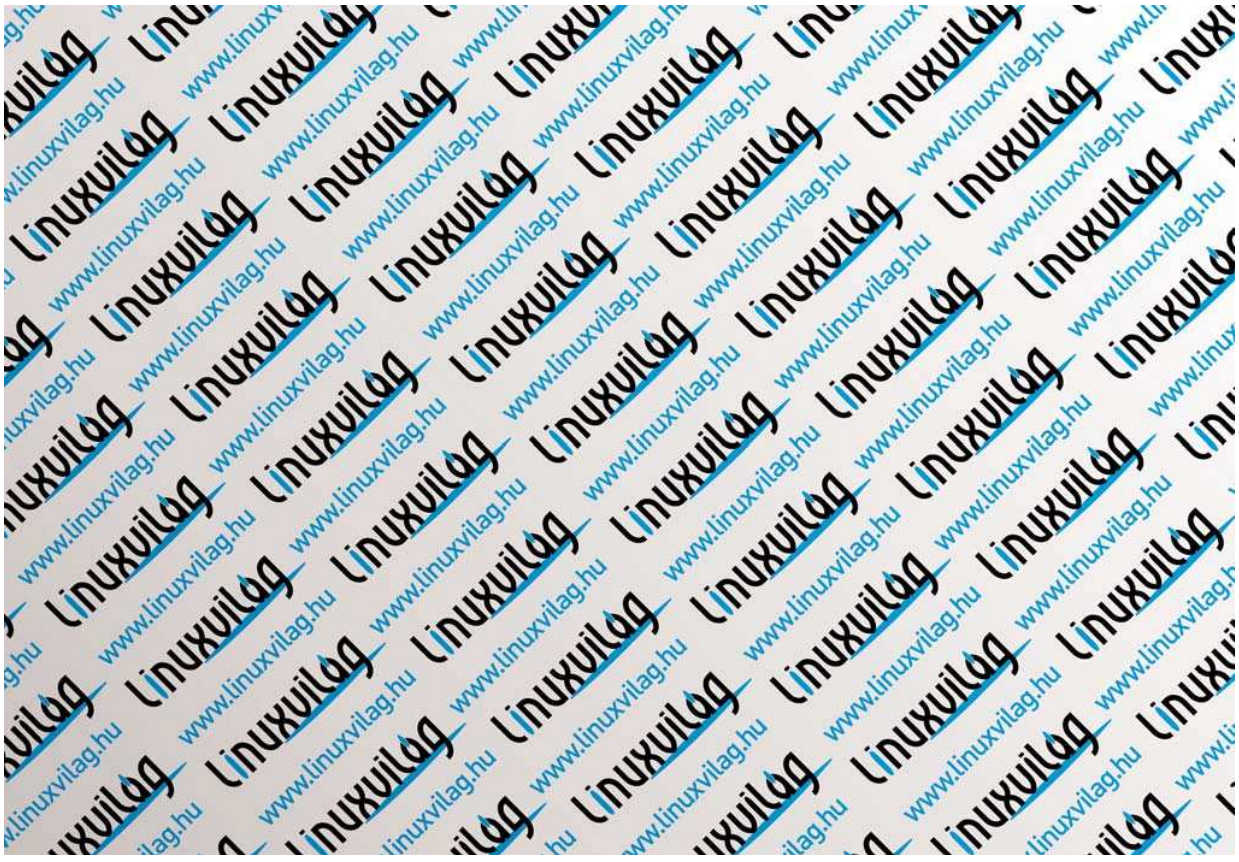
Köszönetnyilvánítás

Köszönetet szeretnék mondani a Digiumnak, Reinhard Maxnak és Steve Landersnek a cikk megírásában való közreműködésért.

A cikkhez tartozó Kapcsolódó címek az 59. CD Magazin/Asterisk könyvtárában találhatók.

Linux Journal 2004. február 118. szám

Brett Schwartz (➲ http://www.bschwartz.com) A Washington állambeli Seattle közelében él feleségével, fiával és kutyájával. Bár otthonosan mozog a különböző operációs rendszerek között, ha teheti, a Linuxot használja.



Magnatune – nyílt zenei kísérlet

Egy lemezkiadó, de nem az ördögi fajtából.

Hogyan építsünk fel egy üzletet zenészek átverése és vásárlók bosszantása nélkül? Miért van szükség háromféle webkiszolgálóra?

A Magnatune egy internetes lemezkiadó vállalat. Megszületéséhez a feleségem lemezét megjelentető angol lemezkiadóval szerzett tapasztalatok, illetve a zeneipar egészével kapcsolatos saját megfigyeléseim járultak hozzá. A Magnatune az alábbiakban különbözik a hagyományos kiadóktól:

- Minden eladás fele a művészt, fele pedig benünket illet.
- Csak letöltéseket árulunk, és nem foglalkozunk a digitális jogok kezelésével (Digital Rights Management, DRM). A vásárlók az albumokat háromféle formátumban tölthetik le: a tökéletes minőséget biztosító WAV vagy FLAC, a magas minőségű, változó bitsűrűségű (high quality VBR) Ogg Vorbis, vagy MP3, illetve 128 kbites MP3 fájlok közül választhatnak. A vásárlók döntik el, hogy az egészért mennyit szeretnének fizetni: 5 és 18 dollár közötti összegekből választhatnak.
- Minden zenénk meghallgatható áramló 128 kbites MP3 formátumban (egész albumokról van szó, nem részletekről), illetve Shoutcast MP3 rádióállomásokon keresztül is. Két kattintás a Magnatune oldalán, és máris a végtelenségig hallgathatjuk kedvenc stílusunk számait. Arra számítunk, hogy előbb vagy utóbb mindenki hall valami olyasmit, ami tetszik neki, és a vásárlás mellett dönt.
- Minden ingyenes zenénkre a Creative Commons-féle „Attribution-NonCommercial-ShareAlike” nevű jogdíj vonatkozik. Ez teszi lehetővé a zenék, illetve a belőlük készített változatok, remixek nem kereskedelmi célú, szabad felhasználását, amíg a Creative Commons jogdíj érvényben marad. Ha valaki kereskedelmi forgalomba hozás szándékával használja fel zenéinket, szerény jogdíjat számítunk fel. A licenc megvásárlása a weboldalon történik és nem teszünk különbséget a felhasználás módja tekintetében. Például nem akadályozhatjuk, és mi nem is akadályozzuk meg zenéink felhasználását akkor sem, ha például nem értünk egyet a felhasználó nézeteivel vagy nem tetszik a zenei ízlése.
- Sikeresek és nyereségesek vagyunk. Csúcsművészeink nagyjából évi 6000 dollárt keresnek, egy átlagos zenész pedig körülbelül 1500-at. Közvetlenül a zenészekkel és a zenészek tulajdonában lévő kiadókkal dolgozunk, és sosem olyanokkal, akik a minél nagyobb pénzösszegek lenyúlásában érdekeltek. Amikor a feleségem leszerződött egy brit lemezkiadóval, nagyon izgatottak lettünk. A vége az lett, hogy ezer korong eladása után tíz évre elvesztette saját dalai jogát (bár a CD-ket évek óta nem gyártják), és ez idő alatt összesen 45 dollárt keresett.

1. kép A Magnatune honlapja (➲ http://www.magnatune.com) a látogatók nyolc alapkérdésére válaszol

Az a kiadó sem a gonoszabbik fajtából való volt, sőt az üzletágban „rendes srácoknak” számítottak. Úgy állapodtak meg, hogy a nyereséget 70/30 arányban osztják el, nyereség azonban egyáltalán nem volt rajta. A kiadó mindenhol falba ütközött: a terjesztők csak akkor voltak hajlandók foglalkozni a kiadó lemezeivel, ha dollárezreket kölött volna felesleges hirdetésekre, a lemezboltok pedig csak csúszópénzért voltak hajlandók rendelni a lemezekből. Egy szó mint száz, minden összeesküdött a kicsi, de haladó szellemű kiadó ellen, ami végül törvényszerűen csődbe is ment.

Zeneipari megfigyeléseim

A rádiók nagyon unalmasak. Minden ismerősöm érdekes zenéket hallgat, viszont ilyesmit ritkán hallani a rádiókban.

A legtöbb stílusirányzatot szinte lehetetlen megtalálni az üzletekben, az éterből pedig teljesen hiányoznak. Manapság főleg countryt, popot és rockot játszanak, itt-ott némi unalmas, biztonságos klasszikus zenével megfűszerezve.
A CD-k túl sokba kerülnek és a művészek, ha szerencséjük van, minden egyes CD eladásából 20 cent és egy dollár közötti összeget kapnak. Emellett a legtöbb korong hamar elfogy, jómagam több használtat vásárolok az eBayen, mint újat az Amazontól. Az előadó az internetes eladások (például az Amazon.com) esetében, egy átlagos szerződést figyelembe véve, az amúgy is nevetséges jutalékának további ötven százalékát veszíti el. Ehhez jönnek még a nemzetközi eladások és a leárazások hatásai, s eljutunk oda, hogy a művész semmit nem kap művéért.
A lemezkiadók előadóikat tíz vagy több évre szóló szerződésekkel igyekeznek magukhoz láncolni. Ha az együttműködés nem válik be, akkor egyszerűen nem gyártanak többet a lemezeikből, a zenészek kezét azonban továbbra is köti a szerződés, melynek értelmében minden évben új albummal kell jelentkezniük. Még a világhírű előadók esetében is előfordul, hogy pénzzel tartoznak kiadójuknak, hiszen a kezdetben kifizetett előleg tulajdonképpen a művésznek adott kölcsönként fogható fel. Továbbá a kiadó általában oly módon intézi ügyeit, hogy a főkönyvben a lehető legkevesebb nyereség jelenjen meg, s így nincsen pénz az előleg visszafizetésére.
Az interneten zenét hallgatni általában fáradságos. Túl sok a reklám, s a rengeteg kattintgatás után rossz minőségben kell meghallgathatnunk a választott zenét. Egyszerűbben fogalmazva: túl sok energia-befektetéssel jár és szinte eredménytelen. Egy jól vezetett internetes rádióállomás, például a Shoutcast vagy a Spinner rendszerek adói, ezt a gondot orvosolják, de a lövészárkokba húzódott zeneipar őket is félre akarja söpörni. E küzdelemben során vetik be a szigorú szerződéseket és a bosszantó jogi huzavonákat.
Mindezeket átgondolva, azt a kérdést tettem fel, hogy miért nem csinálunk végre egy normális kiadót? Kell egy kiadó, mely segít a művészeknek a hírnév megszerzésében, legalább annyi jövedelmet ad számukra, mint egy hagyományos kiadó, és koncertek révén a rajongótáborhoz is közelebb viszi a zenészeket. A Magnatune az én szellemi termékem. A cél az, hogy egy kiadót működtessünk az internet valóságában: a fájlcsere, az internetes rádiók, a zenészek jogai és minden más fontos tényező figyelembevételével. Ezek fölött ugyanis nem lehet átsiklani.

A honlap

Hiszek abban, hogy egy sikeres vállalat, kezdeményezés vagy szervezet honlapjának a leendő látogatók nyolc kérdésére kell azonnal választ adnia. Az *1. képen* láthatjuk, hogy a Magnatune honlapja ezt meg is teszi:

1. Hol vagyok? – Egy grafikus jelkép a bal vagy jobb felső sarokban tökéletesen megfelel. Még jobb, ha jelmondatot is választunk, mely azonnal megragadja a figyelmet. A Magnatune jelmondata: „We are not evil”, azaz „Nem vagyunk gonoszak”.
2. Miért érdekelne ez engem? – Ide a tevékenység egysoros leírása, illetve ha lehetséges egy ráutaló mondat kerül, hogy a látogató mire is számíthat az oldalon. A Magnatune honlapján az „Internet music without the guilt” (internetes zene, bűntudat nélkül) és a „Magnatune, the Open Music Record Label” (Magnatune, a nyílt zene lemezkiadója) feliratok láthatók.
3. Mit csináljak itt? – Az első alkalommal idelátogatók számára is világosnak kell lennie a következő lépésnek. A Magnatune azt szeretné, hogy az emberek azonnal hallgathassák a zenét, tehát a különböző stílusok neveire kattintva rögtön az adott irányzat előadóihoz ugorhatunk.
4. Miért jó ez? – Az interneten rengeteg nagyszerű honlap található, az emberek ideje azonban véges. A látogató figyelmét csupán néhány másodpercig tudhatjuk magunkénak, tehát gyorsan tudatnunk kell vele, hogy miért is érdemes maradnia és tovább nézelődni. Ha e-kereskedelemmel foglalkozunk, akkor tudnunk kell, hogy ebbe az emberek már belefáradtak. Ha a második kérdést jól válaszoltuk meg, akkor talán kapunk újabb fél percet az idejükből. A Magnatune az alábbi szöveggel indít: „Mi egy lemezkiadó vagyunk. De nem vagyunk gonoszak. Nálunk az árut kipróbálhatod, mielőtt megvennéd. Nevezhetjük szabadon felhasználható zenének is.” A „lemezkiadó”, a „nem vagyunk gonoszak” és a szabadon felhasználható (shareware) furcsa társítás, tehát a látogató érdeklődését minden bizonnyal sikerül felkeltenünk.
5. Mik az újdonságok? – Látogatóinkat azzal ösztönözhetjük az oldalunkra való visszatérésre, ha azonnal láthatják, mi változott a legutóbbi látogatásuk óta. A Magnatune oldalán mindig újdonságokra lelhetünk (például újságcikkek), de a legtöbben az új művészekre és lemezekre kíváncsiak, ezért őket helyeztük el a főoldalon.
6. Létezik hírlevél szolgáltatás? – Minden weboldalnak szüksége van hírlevélre. Ha a feliratkozást lehetőségét a főoldalon kínáljuk, bizton számíthatunk rá, hogy a látgatók 2-5 százaléka ki is használja a lehetőséget.
7. Mit kell még tudnom? – A ***Rólunk*** (About) rész igen fontos. Az alapítóknak itt kell elmagyarázniuk, hogy miért hozták létre a weboldalt és a vállalkozást.
8. Akárhová eljuthatok? – A további lépésekre utaló megjegyzések ellenére a látogatók gyakran maguk szeretnék eldönteni, hova lépnének szívesen tovább. A Magnatune látogatóinak 15 százaléka azonnal a ***Művészek*** (Artists) feliratra kattint. Gondoskodjuk róla, hogy az oldal főbb kategóriáira mutató hivatkozások könnyen elérhetők legyenek.

A Genre (Műfajok) oldal

A zenei oldalakkal kapcsolatos legnagyobb gondom az, hogy ha csak zenét szeretnék hallgatni, és utána vissza akarok térni a munkámhoz, ez a lépés túl sok időt vesz igénybe. A második gond az, hogy legtöbbször nincs elegendő zene, s általában az is csak gyenge minőségben érhető el, s így az ember képtelen dönteni. A legtöbb ismerősöm vásárlás előtt meghallgatja a zenét, például a rádióban, koncerteken, egy barát lakásán, étteremben és így tovább. Mivel a látogatók csupán kis hányada ismeri előre a felkínált zenéket, ezért célszerű lehetőséget adni számukra, hogy meghallgathassák, és kényelmesen dönthessenek a vásárlásról. A Magnatune-nál bármit meghallgathatunk, 128 kbites MP3 fájlokként.

Ha a látogató a Magnatune honlapon kiválasztja a neki megfelelő zenét, például az ***Electronic*** stílust, a weboldal négy lehetőséget kínál fel:

1. Az Electronica kategória minden albumának meghallgatása egymás után.
2. A stílus képviselőinek alkotásaiból készített mix meghallgatása.
3. Valamely alkotó egyik albumának meghallgatása.
4. Egy adott alkotó ismertetőjének elolvasása.

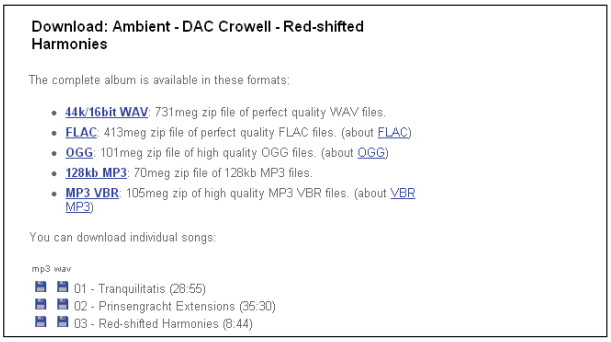
Az első három lehetőség választásával azonnal elkezdhetünk zenét hallgatni.
Az emberek az új zenéket inkább barátaik révén vagy saját érzékeiket követve szeretik megismerni, és nem szeretik, ha a rádió és az MTV mondja meg nekik, hogy mit kell hallgatni. A '80-as években a legtöbb programot nem próbálhattuk ki vásárlás előtt, manapság viszont ez szinte elképzelhetetlen. Egyszer majd a zeneipar minden résztvevője a szabad választás modelljét fogja követni, hiszen az ebből származó előnyök nagyon csábítóak.

Az alkalmazott programok

A Magnatune nyílt forrású programokat alkalmaz. Öt 1U magas kiszolgálóegységet használunk, melyek mindegyike 2,4 GHz-es processzort tartalmaz, és Linuxot futtatnak. Az adattárolás négy, állványba szerelt RAID-tömb segítségével történik, mindegyiket hét merevlemez alkotja, így összesen 28 merevlemezünk van. Mindegyik tömb RAID5 rendszerben működik, darabonként egy-egy működés közben is cserélhető meghajtóval.
A HTML-oldalak kiszolgálását PHP-t és OpenSSL-t futtató Apache 2 kiszolgálók végzik. Amikor a Slashdot a Magnatune-ről hírt adott, azt figyeltük meg, hogy az Apache képtelen volt megbirkózni a képek betöltésével. Jelenleg a képekre irányuló összes HTTP-kérelmet az AOLserver szolgálja ki, amely nagy terhelés esetén a legrövidebb válaszidővel képes e feladat elvégzésére.
Az MP3 fájlokat a Mathopd nevű, egyszálú, aszinkron HTTP-kiszolgáló kezeli, mivel nagy fájlok esetén ez képes a legnagyobb fokú rugalmasságra. A kiszolgálót úgy állítottuk be, hogy ugyanolyan az „Expires:” HTTP-választ továbbítson, mint amilyet a Yahoo is használ. A Mathopd válaszideje hosszabb, mint az AOLserveré, ezért kis méretű képfájlokhoznem használjuk.
A Magnatune minden weboldala PHP-t használ. A vásárlásokat egy MySQL adatbázishoz kapcsoljuk. Egy Perl-parancsfájl hozza létre a számlistákat és az m3u lejátszási listákat. A többi feladatot Tcl parancsfájlok kezelik, például a ZIP, Ogg Vorbis és FLAC fájlok létrehozását, illetve az albumonkénti jelszófájlok létrehozását az Apache számára. Apache HTTP-jelszavak védenek minden fizetős letöltést, a fájlokat pedig az `rsync` segítségével osztjuk el a kiszolgálók között.

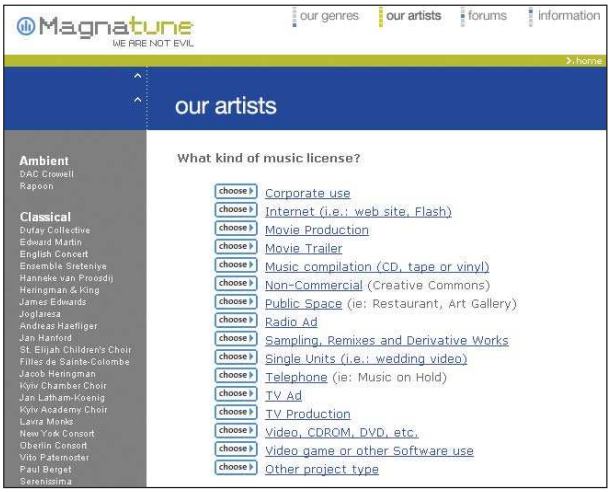
E-kereskedelem

A Magnatune zeneletöltési és -vásárlási folyamatában számos szokatlan elem található. A vásárlók döntik el, hogy mennyit szándékoznak fizetni, nem használunk bevásárlókosarat, támogatjuk a PayPalt és Visa kártyával való vásárlást, sőt név megadása nélkül is lehet fizetni.



2. kép Indulhat a letöltés, már csak formátumot kell választanunk

A ***Buy*** (Vásárlás) gombra kattintva egy oldal jelenik meg a „Mennyit szándékozik fizetni?” kérdéssel, erre 5 és 18 dollár közötti válaszokat fogad el, és alapértelmezés szerint a 8 dolláros árat javasolja. Egy másik figyelmeztető felirat így szól: „Az összeg ötven százaléka a zenészé, kérjük, legyen nagyvonalú.” A 2003. szeptemberi átlagár 9,82 dollár volt, ami azt mutatja, hogy ha a vásárló dönthet, akkor többet fog fizetni a javasolt árnál.
Úgy határoztam, hogy a Magnatune oldalán nem használunk bevásárlókosarat, mivel az őket alkalmazó rendszerekben gyakori gond, hogy a vásárló az utolsó pillanatban mégis lemond a gondosan összeválogatott termékek megvételéről. Ha az embereket érdekli egy album és a ***Vásárlás*** gombra kattintanak, akkor le kell csapni a lehetőségre. Ha kosarat használnánk, a felhasználó egy album kiválasztása után nyilván további lemezek után kutatna, s ha nem találna semmi neki tetszőt, a kezdeti lelkesedése alábbhagyna, és a vásárlás füstbe megy.
Egy átlagos napot figyelembe véve elmondható, hogy a vásárlók 30-50 százaléka választja a PayPal rendszerben történő fizetést. Véleményem szerint ennek az az oka, hogy a hitelkártyás fizetés esetén a vásárló adatai az ügylet lebonyolítása után is a kereskedőnél maradnak. Az emberek így nem érzik biztonságban magukat. A PayPal esetében a fizetés megtörténtével az ügy lezárul és a jövőbeli kockázat teljesen kizárt.
A PayPal kétféleképpen tudja értesíteni a kereskedőt a fizetésekről. Az IPN rendszerben a PayPal egyedi azonosítóval ellátott HTTP visszahívási üzenetet (callback) ad át webkiszolgálónknak, ami azután az azonosító birtokában a HTTP protokollon keresztül lekéri a PayPaltól az ügylet részleteit. Ez egy igen biztonságos módszer, de két gond is akad vele: programozása jelentős munkát igényel és nem teszi lehetővé a vásárlás utáni azonnali letöltést, mivel a webkiszolgálónknak az értesítést előbb meg kell várnia. Számos bevásárlókosarat használó rendszer támogatja az IPN-t, a letöltési útmutatót pedig elektronikus levélben küldi el a vásárlónak. Mi azért nem alkalmazzuk ezt a módszert, mert megfigyeléseink alapján az internetszolgáltatók levélszemétszűrő (spam) programjai gyakran megakadályozzák, hogy ezek a levelek eljussanak a vevőkhöz. A vásárlók erről nem tudhattak, s ez sokakat bosszantott.
Mi a PayPal másik, egyszerűbb szolgáltatását használjuk, amelyben a PayPal a fizetés lebonyolítása után a felhasználót visszaküldi az oldalunkra, HTTP POST adattal. Ez



3. kép Filmet készítenénk? Legális zenére van szükségünk a telefonközpontunk számára? Látogassunk el az albumok licencelésével foglalkozó oldalakra

jóval egyszerűbben programozható és eléggé megbízható. De nem minden PayPal befizetés érkezik meg azonnal hozzánk, előfordul, hogy napok telnek el a pénz megérkezéséig, és a kereskedőnek döntenie kell, hogy engedélyezi-e az azonnali letöltést. Mi úgy döntöttünk, hogy megbízunk vásárlóinkban, így a fizetés után rögtön letölthetők a zenék.

A hitelkártyaadatokat feldolgozó rendszerünk alapértelmezés szerint nevet, telefonszámot, postai címet és háromjegyű AVS kódot is kérne minden vásárlás esetén, de erre bennünket semmi nem kötelez. Hitelkártyapartnerünk egy százalékos díjat számít fel azért, mert vásárlóinktól nem kérjük el ezeket az adatokat, de ez egy 10 dolláros ügylet esetében csupán 10 cent. Ez a díj szinte elenyésző a 25 centes Visa díj és az internetes átjáró (gateway) használatáért fizetendő szintén 25 centes díj mellett, tehát úgy érezzük, megéri ennyit tenni vásárlóinkért.

A Magnatune oldalán a vevőnek csak a hitelkártyaszámát és a kártya lejárati dátumát kell megadnia – mindössze ennyi szükséges az ügylet lebonyolításához. Név és elektronikus levélcím is megadható, de ez nem kötelező – vevőink figyelmét pedig egy üzenetben hívjuk fel arra, hogy a megrendelés részleteit nem áll módunkban elektronikus levélben továbbítani. Azáltal, hogy ilyen kevés adatot kérünk, felgyorsítjuk az ügymenetet. Továbbá az elektronikus levélcím elhagyásával a látogató abban is biztos lehet, hogy a vásárlás következtében nem fog kéretlen levelekkel megtelni a postaládája.

A fizetés után

Az anyagiak rendezését követően egy köszönetnyilvánítást tartalmazó oldalra továbbítjuk a vevőt, mely a vásárolt fájlokra mutató URL-eket tartalmazza. A vevő egyszerű, négyjegyű HTTP-felhasználói nevet és jelszót kap. Ha az illetőtől elektronikus levélcímet is kaptunk, akkor a letöltési tudnivalókat erre a címre is postázzuk. A letöltési oldal a kért zenéket többféle formátumban is felkínálja (2. *kép*). A WAV és FLAC fájlok tulajdonképpen a korong tökéletes másolatai, de Ogg Vorbis, MP3-128 és MP3-VBR fájlok is letölthetők innen. Akár az összes változatot is leszedhetjük, ameddig csak akarjuk (a jelszó sosem jár le). Az egyszerűség kedvéért a teljes album egyetlen ZIP fájlként tölthető le. A ZIP körülbelül 10 százalékot tömörít a WAV fájlokon, ami azonban ennél fontosabb, így az egész albumot egyben, felesleges kattintgatások nélkül érhetjük el, ADSL- vagy kábelkapcsolat esetén nagyjából egy óra alatt.

Licencelés

Az albumok melletti ***License*** gombra kattintva egy oldal jelenik meg a „Milyen jellegű zenét szeretnél?” felirattal. Ezen tizenhat különféle típust láthatunk, kezdve a filmes, reklámjellegű, valamint internetes felhasználástól a CD-válogatásokon való megjelenésen át a telefonközpontok gépi menüjének háttérzenéjeként történő alkalmazásig. A csatlakozó oldalon a választott kategóriához kapcsolódó adatokat olvasgathatunk és adhatunk meg, illetve ehelyütt található az árajánlat is. A felhasználás részleteinek megadása után a fizetési oldalra kerül az érdeklődő, amelyen a díj kifizetését követően azonnal életbe lépő szerződést olvashatja el.

Merre tovább?

2003 májusában történt elindulása óta a Magnatune számos híroldalon feltűnt (Slashdot, Fark, BoingBoing, Wired, NPR). A sajtóban megjelenő híradások nyomán hatalmas forgalomnövekedés történt a weboldalon. Két különböző földrajzi helyen található öt kiszolgálónk mindegyike 100 Mbites vonallal kapcsolódik az internetre, ezek között a terhelést a fő weboldal dinamikus PHP kódja segít elosztani. Úgy találtam, hogy ez a felállás megfelel a jelenlegi igényeknek. Figyelembe véve azonban a Magnatune havi harminc százalékos forgalom- és bevétel-növekedését, egyre komolyabban hiszem azt, hogy nemsokára az egyenrangú (peer-to-peer) fájlcserélő hálózatokhoz hasonló rendszert kell alkalmaznunk a terhelés kiegyensúlyozása és az egyszerű bővíthetőség érdekében.

E cikk írásakor a Magnatune kiadó körülbelül hatvan zenész 130 albumát gondozza, s ez a szám havonta 15 zenésszel és harminc albummal bővül. Legsikeresebb zenészeink évi hatezer dollárt keresnek, az átlag pedig nagyjából 1500-at. Ha a harminc százalékos növekedés megmarad, akkor még ennél is többet tudunk fizetni művészeinknek. Mindennél fontosabb számomra, hogy közvetlen anyagi segítséget nyújthatok a zenészeknek. Izgalomba hozza őket a siker, az emberekhez eljut a zene, és az alkotók az anyagi biztonságnak köszönhetően további nagyszerű albumokat készíthetnek.

Linux Journal 2004. február, 118. szám

John Buckman
a Magnatune internetes lemezkiadó alapítója. Emellett a ➲ http://www.Piazzolla.org tangózenével foglalkozó oldal gazdája. Ő gondozza az Internetes Lant Társaság (Internet Lute Society) ➲ http://www.lutesociety.com címen elérhető oldalát és a Johann Sebastian Bach munkásságával foglalkozó ➲ http://www.JSBach.org honlapot is.

Műsorszórás: nem csak a helyi hálózaton (2. rész)

A VideoLAN projekt egyéb műsorszóró és különleges képességei.

Sorozatunk előző részében már bemutattuk a VideoLAN helyi hálózatokon alkalmazható műsorszóró képességeit, ám mint azt a cikk végén említettem, a dolognak ezzel messze nincs vége. Például nem használtuk ki a program nyújtotta apró, ám annál hasznosabb lehetőségeket, kényelmi megoldásokat, gondolok itt az adatfolyam valós idejű átalakítására, az egyidejűleg többféle protokollon történő sugárzásra, vagy épp a kényelmes webfelületről irányítható televízióállomás lehetőségére. Ezek mindegyikéről hamarosan szót ejtünk, de ez önmagában kevés lehet annak az olvasónak, aki számára szűknek bizonyulna a helyi hálózatok szabta mozgástér. Valóban akad a dolognak némi hátránya, ugyanis tömeges nézőtábort ezzel a módszerrel jelenleg még nem toborozhatunk. Azonban kár lenne azt hinnünk, hogy csak ilyen szűk keretek között mozoghatunk, ezért szeretnék néhány olyan módszert megismertetni, amellyel kitekinthetünk a helyi hálózatok világából. Részben ehhez kapcsolódóan fogunk megismerkedni azokkal a bizonyos, eddig nem használt képességekkel, és próbáljuk olyan módon felépíteni a képzeletbeli televízióállomásunkat, hogy az később megfeleljen az új követelményeknek. Ilyen új követelmény lehet például az, hogy HTTP protokollon keresztül távoli ügyfelek számára is elérhetővé váljon a sugárzott műsorfolyam. Látni fogjuk, hogy bizony létezik néhány körülmény, amely eltérő a helyi hálózatok esetében tapasztaltaktól. Vágjunk bele!

Átkódolás

Mint tudjuk, a videofolyamok erőteljes tömörítésen és átalakításon mennek keresztül, mielőtt elnyerik tárolható, különleges formátumukat. A tárolhatóság mellett némely formátum kifejezett célja, hogy könnyen sugározható legyen az internet kusza vezetékein is. A helyi hálózatokon ilyen gonddal megfelelő lejátszók esetében nem kell küszködnünk, azonban a fájlok tömörségét, mindenképp érdemes kihasználnunk, ezzel ugyanis sávszélességet takaríthatunk meg. Ezenkívül jó nekünk az, hogy létezik egy előre kialakított befoglalókeret, így az adatcsomagok kompakt, jól kezelhető egységeket alkotnak. Ezen okok, s részben a módszer egyszerűsége miatt a VLC az adatfolyamot a lemezen tárolt formában, az eredeti kódolással sugározza. Ez természetesen nem azt jelenti, hogy az adatfolyamot kivágja a hálózatra, csupán annyit, hogy nem nyúl a kép- és hangsávokhoz, „változatlan” állapotban továbbítja őket.

Felmerülhet a kérdés, hogy mi történik, ha valaki előre meghatározott formában szeretné a különböző módokon tárolt anyagait ilyen módon közzétenni? A megoldás az adatfolyam menet közben (on-the-fly) történő programbeli átkódolása. A módszer lényege, hogy a kép, illetve hangsávok nem a lemezen tárolt formátumban kerülnek sugárzásra, hanem dekódoljuk a kiszolgálón, s azonnal újrakódoljuk, majd az így kódolt adatfolyamot küldjük el a hálózaton. Ennek így elsőre nem sok értelme van, de ha jobban a dolgok mögé nézünk, látni fogjuk, hogy ez egy remek lehetőség.

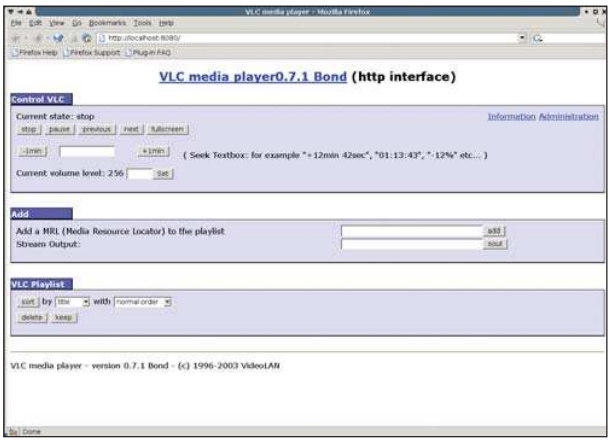
Egyik ilyen szükséglet a szabványosság. Ugyanis elképzelhető, hogy mi minden adatot ugyanolyan formában szeretnénk eljuttatni a nézőkhöz. Mint tudjuk, a digitális kép- és hanganyagok a lehető legváltozatosabb formában és kombinációban állhatnak rendelkezésünkre, tehát elég nagy a valószínűsége annak, hogy átalakításra szorulunk.

A másik olyan kényszer, ami miatt átalakításra szorulhatunk: a sávszélesség szűkössége. Előfordulhat, hogy a tárolt videofájl bitrátája túl nagy a hálózatnak, amelyen sugározni szeretnénk. Ez lehet azért, mert túlterhelt, vagy épp azért, mivel egy „átjárót” létrehozva szeretnénk egy kimenő kapcsolaton másik hálózatba juttatni az adatfolyamot, és a távoli hálózattal szűk vonalon vagyunk összekötve. Ilyenkor az a bevett szokás, hogy az adatfolyamot nagyobb veszteséggel áttömörítjük, így kisebb méretű (ugyanakkor gyengébb minőségű) lesz.

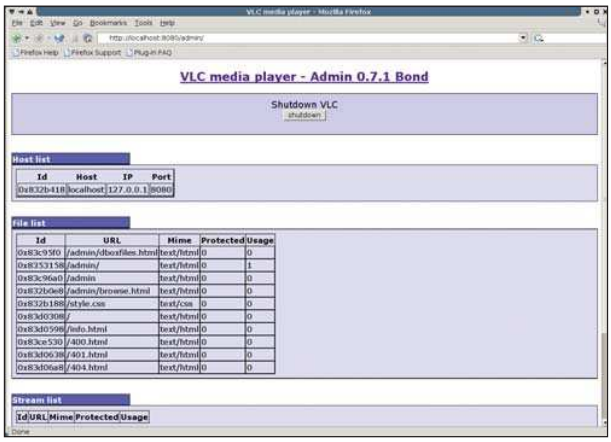
A VLC használata során remekül megoldhatjuk az ilyen jellegű nehézségeket, s ehhez mindössze néhány tíz karakterrel szükséges bővítenünk a parancssorban kiadott utasításunkat. Az előző cikkben említettem a VLC kimeneti moduljainak használatát, ezeket a modulokat nagyon egyszerű parancsfájl segítségével állíthatjuk be. A kapcsoló formátuma a következő:

```
--sout
#modul1{jellemző1=...,jellemző2=...}:#modul2
➥{jellemző1=...,jellemző2=...}:...
```

Adatfolyam átkódolása esetén nekünk épp egy ilyen kimeneti modul használatára lesz szükségünk, melynek neve: `transcode`. A `transcode` modul a bemeneti adatfolyamot magán átereszti és a kimeneten átkódolt adatfolyamként jeleníti meg, s mindezt valós időben teszi, azaz a bemenetet



1. kép A VLC HTTP-felülete



2. kép A VLC HTTP-felülete – felügyeleti oldal

menet közben (on-the-fly) alakítja át. A módszer egyetlen hátránya, hogy mivel a tömörítő algoritmus aszimmetrikus, a tömörítés meglehetősen gépigényes, így jó erős processzorra van szükségünk ahhoz, hogy az átkódolás folyamatos legyen.
Nézzünk most egy példát az átalakításra! Adjuk ki a következő utasítást a parancsértelmezőben:

```
vlc -vvv <bemeneti adatfolyam> --sout
➥#transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128,
➥deinterlace}:standard{access=udp,mux=ts,
➥url=239.255.12.42,sap=Probafolyam}
```

A parancs hatására a bemeneti adatfolyamból MPEG-4 formátumú képsávból és MPEG audio layer 2 formátumú hangsávból álló átviteli adatfolyam keletkezik (ez gyakorlatilag a szabványos DVB adatfolyam, de ilyennel találkozhatunk a DVD-ken tárolt videók esetében is), amely a 239.255.12.42-es csoportos üzenetküldési címre továbbítódik némi csatornainformációval fűszerezve. Számunkra most leginkább a `transcode` modul jellemzői az érdekesek, vizsgáljuk meg egy kicsit.

`vcodec` (video codec): segítségével adhatjuk meg az új képsáv tömörítéséhez használandó kodeket.
`acodec` (audio codec): ezzel a jellemzővel állíthatjuk be a használni kívánt tömörítő eljárást a hangsávra vonatkozóan.
`vb` (video bitrate): meghatározza, hogy mekkora lesz a kimeneti adatfolyam mérete (kbit/másodperc). Gyakorlatilag ennek segítségével szabályozható, hogy a tömörítés során az eredeti képanyaghoz képest mekkora legyen a veszteség mértéke. Minél nagyobb ez a szám, annál nagyobb az adatfolyam mérete, s annál jobb a kép minősége.
`ab` (audio bitrate): ugyanazt jelenti, mint a vb jellemző, csak épp ez a hangsávra vonatkozik.

Egyéb fontos kapcsolók:
`width`, `height`: a kép méretét (szélességét, magasságát) adhatjuk meg vele, így méretezhetjük át az eredeti képet.
`croptop`,`cropbottom`,`cropleft`,`cropright`: ezekkel a kép szélein jelentkező vékony fekete kereteket vághatjuk le a négy irányból.

Terjedelmi okok miatt az adatfolyamok szerkezetéről, az alkalmazható kodekekről nem tudok szót ejteni. Erről egyébként egy teljes cikk jelent meg a Linuxvilág 15. számában „Digitális videózás” címmel. Azok az olvasóink, akik mélyebben érdeklődnek a téma iránt a Linuxvilág honlapján (vagy előfizetőként) elolvashatják.

Többszörös adatfolyam-sugárzás

A módszer lényege az, hogy a bemeneti adatfolyamot nem egyetlen kimeneten tesszük közzé, hanem egyszerre több címre, több kimenetre is elküldjük. Ez akkor lehet hasznos, ha például szeretnénk kilőni az adatfolyamot a helyi hálózatból egy általunk meghatározott gépre, IP-címre úgy, hogy közben a helyi hálózaton a többiek is lássák az adást. Példaképp álljon itt egy helyzet: tegyük fel, hogy van egy cégünk két telephellyel, két helyi hálózattal, s a két telephely valamilyen szűkebb kapcsolattal össze van kötve. Azt szeretnénk, hogy az egyik telephelyen sugárzott adást mindkét helyi hálózaton megtekinthessük. Mivel az UDP protokollon küldött csoportos üzenetek a helyi hálózatok határán érvényüket vesztik. Ha csak nincs beállítva egy csoportos üzenetek továbbítására alkalmas átjáró (gateway), nekünk egy külső IP-re, a másik telephely átjárójára is el kell küldenünk az adatfolyamot, amely a két telephely közötti közvetlen kapcsolaton halad át, ily módon gyakorlatilag kézzel hajtjuk végre az útvonalválasztást (route). Ezután a másik oldalon a bejövő adatfolyamot ismét szét kell dobni az ottani helyi hálózatra, s készen is vagyunk.
Ha valami hasonlót szeretnénk csinálni, akkor megint csak a fentebb említett modulokat kell segítségül hívnunk, közülük is a `duplicate` nevűt, amellyel különböző kimeneti célokat adhatunk meg egyazon bemeneti adatfolyamhoz. A modul neve némiképp sántít, ugyanis azt sugallja, hogy megkétszerezi a kimeneteket, de valójában n-szerezi, ugyanis nemcsak két kimenetet határozhatunk meg, hanem többet is.
Egy ilyen parancs a következőképpen fest:

```
vlc -vvv <bemeneti adatfolyam> --sout
➥'#duplicate{dst=standard{access=udp,mux=ts,url=
➥239.255.12.42,sap=Probafolyam},dst=standard
➥{access=udp,mux=ts,url=192.168.1.2}}'
```

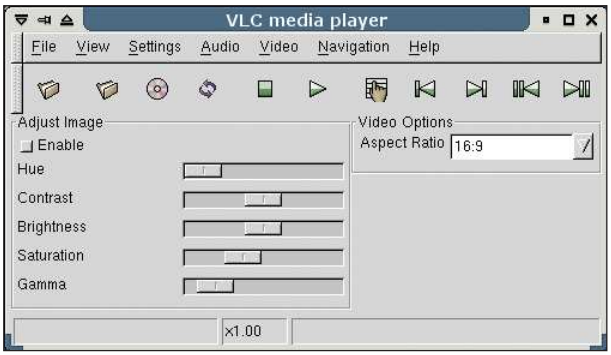
Ha közelebbről megvizsgáljuk a modult, láthatjuk, hogy a `dst` (destination = célállomás) jellemzővel adhatjuk meg egymás után a kimeneteket, vesszővel elválasztva. Láthatjuk, hogy a `dst` jellemző után a teljes adatfolyam-szerkezetet meghatározhatjuk, azaz minden adatfolyamot különböző kapcsolókkal láthatunk el, mintha valóban különböző kiszolgálókról származnának.
A fenti két lehetőséggel, tehát az átalakítással és a többszörös adatfolyam-sugárzással, azaz inkább ezek kombinálásával mindenféle érdekes dolgot és számtalan lehetőséget kaphatunk. Mivel egy adott kimenetet a teljes adatfolyam-szerkezettel jellemezhetünk, megoldható, hogy a helyi hálózaton az eredeti, kiváló minőségű ám nagy sávszélesség-igényű adatfolyam csordogáljon, ugyanakkor a kimenő, kis sávszélességű vonalon egy menet közben (on-the-fly) átalakított, rosszabb minőségű, kisebb méretű műsor továbbítódjon, s mindegyik adatfolyamot tetszés szerint testreszabhatjuk.

A VLC irányítása HTTP-n keresztül

A VLC-t a fejlesztők egy egyszerű kis HTTP-kiszolgálóval is felszerelték. Ez a kiszolgáló végzi egyrészt a VLC távoli irányítását, másrészt a HTTP alapú adatfolyam-sugárzást. (Érdekes tény, hogy a SHOUTcast esetében is épp egy ilyen felépítésű és szerepű HTTP-kiszolgálóval találkozhattunk, talán nem véletlenül. (Linuxvilág, 35. szám. 38–41. oldal, Bemutatkozik a SHOUTcast)) Nézzük előbb a webről történő vezérlést!
Ha igényünk támadna, hogy távolról szeretnénk irányítani a VLC-t kiszolgálóként, akkor ezt részben elintézhetjük valamilyen távoli terminálról. Ez azonban nem ad lehetőséget a pozícionálásra, és a sugárzott adatfolyam kicserélése is bajos lehet, hiszen a VLC grafikus felületéhez azon a bizonyos idegen gépen egy X-kiszolgálónak kell futnia, és egyébként is nehézkes egy ilyen rendszer kezelése. Ennek megoldására a fejlesztők ellátták a VLC-t egy-két apró CGI parancsfájllal, ennek segítségével a kiszolgálóhoz csatlakozhatunk, mint webkiszolgálóhoz, s immáron grafikus felületről, kényelmesen kezelhetjük kiszolgálónkat, szinte bárhonnan a világból, s nem kell hozzá semmi egyéb, csak egy böngésző.
Nekünk ehhez semmi mást nem kell tennünk, mint a kiszolgálón elindítani a VLC-t az alábbi utasítással:

```
vlc -I http --http-host 192.168.0.102:8081 --sout
➥udp:239.255.12.42
```

Ezután, ha valamelyik másik gépen (vagy akár magán a kiszolgálón) ellátogatunk a ***http://192.168.0.102:8081*** címre, akkor egy kezelőfelületen találjuk magunkat.
A felület vízszintesen három részre tagolódik. A legelső a pillanatnyi adatfolyamot kezeli. Lehet benne pozícionálni, ide-oda tekergetni és tud váltani a lejátszási lista elemei között. A következő részben adhatjuk hozzá a sugározni kívánt fájlokat a lejátszási listához. Ehhez meg kell adnunk a fájl teljes elérési útját. Lehetőség nyílik még a kimeneti forrás meghatározására, ám ez valamiért nem működik, talán nincs még belerakva a kódba, épp ezért már az indításkor célszerű megadni, mint ahogyan azt meg is tettük a fenti paranccsal. A legutolsó részben a lejátszási listát kezelhetjük, megváltoztathatjuk az elemek sorrendjét, törölhetünk közülük, s ha bármelyikre rákattintunk, a kiszolgáló onnantól kezdve azt kezdi el sugározni.



3. kép A VLC grafikus felülete kiterjesztett módban

HTTP alapú műsortovábbítás

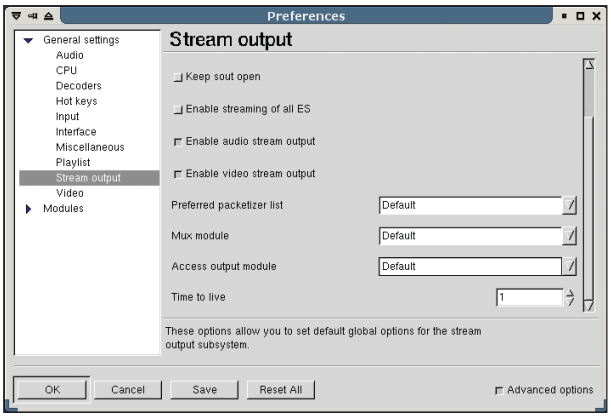
Gyakorlatilag ez az a módszer, amelyet a webrádióknál már megismertünk. Ilyenkor az adatfolyam úgy érkezik az ügyfél gépére, mintha csak egy HTML dokumentumot töltene le valamelyik kiszolgálóról. Itt természetesen nincs lehetőség a csoportos üzenetküldésre, így ehhez hatalmas sávszélességre van szükségünk, ugyanakkor megvan az az előnye, hogy könnyedén kilép a helyi hálózatok nyújtotta szűkös területről, feltéve, ha az ügyfélig elegendően nagy a sávszélesség.
Ha ilyet szeretnénk, akkor kimeneti forrásnak ezt a bizonyos beépített HTTP-kiszolgálót kell megadnunk. Figyelem, ez nem ugyanaz, mint az előbbi webről történő vezérlés. Ott csak egy felületet biztosítottunk a beavatkozásra, itt viszont azon fogjuk továbbítani az adatfolyamot, természetesen a gép egy másik kapuján (port).
Próbáljunk indítani most egy ilyen szolgáltatást. Adjuk ki az alábbi parancsot:

```
vlc -vvv <bemeneti adatfolyam> --sout
➥#standard{access=http,mux=ts,url=localhost:8080}
```

Az URL helyére természetesen azt a címet írjuk, amelyen a gépünket az adott adatfolyamhoz való hozzáférés érdekében elérni szeretnénk. Arra ügyeljünk, hogy ilyenkor a HTTP-kiszolgáló az URL-t virtuális kiszolgálóként kezeli, azaz csak az adott cím beírásával érhető el, ha mondjuk, a gép IP-jét címezzük, nem fogunk látni semmit.
Ha ez megvolna, csatlakozzunk az előbb létrehozott kiszolgálóhoz, először VLC-vel az alábbi módon:

```
vlc http://localhost:8080
```

A parancs hatására megkezdődik a videó lejátszása, mi pedig örülhetünk, hogy ezt bárki akár távolról is elérheti. Azonban itt is szükség lesz a VLC médialejátszó-képességeire, s az ügyfél gépén is jó, ha ez fut, ugyanis bármilyen formátumú HTTP-vel továbbított videó lejátszására képes. Egyéb esetekben, például MPEG-1 vagy MPEG-2 (VCD, SVCD) továbbításakor az MPlayer nevű lejátszó is minden gond nélkül boldogul a HTTP-ről történő lejátszással, de



4. kép A VLC grafikus beállítópanelje

gondolom ezen senki nem lepődik meg. Az viszont már meglepő, hogy egy szabványos, szakaszolható (streamelhető) MPEG-1 adatfolyamot ily módon sugározva a Windows Media Playerrel nem sikerült megnyitnom. Mint tudjuk, a weben elérhető különböző videotartalmakat teljesen hasonlatosan játssza le menet közben a Windows Media Player, ha a böngészőben rákattintunk akár az adatfolyam, akár egy fájl hivatkozására. Ennek ellenére a program állandóan hibaüzeneteket küldött. Folyamatosan tudta átmenetitárazni (cache), böngészőből a lemezre menteni, onnan is képes volt megnyitni, tehát az adatfolyam hibátlanul eljutott a célgéphez, ám menet közben nem volt képes az adatfolyam lejátszására. Gondolom közben mindenki kitalálta, hogy ezzel azt szerettem volna elérni, hogy az ügyfél-oldalon ne kelljen telepíteni „különleges” programot azért, hogy tévézhessünk a webről, de sajnos a legelterjedtebb lejátszó ezt nem tudja. Feltételezem, hogy egyéb programokkal megvalósítható a dolog, de ha már úgyis telepíteni kell, akkor használjuk inkább a VLC-t, Windows alatt az a legalkalmasabb – talán még filmnézésre is.

A grafikus VLC

Bár eddig mindvégig parancssorból vezéreltük a VLC-t, de lehetőségünk van a grafikus felület igénybevételére is, ezen jelentős mennyiségű kapcsoló kiváltása lehetséges, vagyis a beállítások palettája meglehetősen színes. Bár van, amit csak parancssorban tudunk megadni, azért rengeteg testreszabási lehetőséget ad a VLC grafikus felülete. A fentebb említett modulok szinte mindegyikét beállíthatjuk, ezenkívül a program egyéb részei is kiválóan vezérelhetők. Ahhoz, hogy mindezt elérjük, ne felejtsük el a beállítópanel jobb alsó sarkában engedélyezni a haladóknak szóló nézetet.

VOD

Ez a módszer (Video-On-Demand= igény szerinti videózás) egy kicsit el is tér az eddig tárgyalt műsorszórástól, hiszen az eddigiekkel ellentétben itt nem az adatfolyamnak ugyanahhoz a szakaszához kapcsolódik mindenki, hanem lehetőség nyílik igény szerint egy adott szakaszhoz kapcsolódni. Ez a gyakorlatban annyit jelent, hogy bármikor elkezdhetjük nézni a filmet, előre és visszatekerhetünk, megállíthatjuk stb. Majdnem olyan, mint a tévéadás és a VHS kazetta nézése közötti különbség, leszámítva azt az aprócska tényt, hogy itt nem áll rendelkezésünkre a kazetta.

A VideoLAN által javasolt megoldás igen egyszerű, így hangzik a receptjük: végy egy tetszőleges webkiszolgálót, például az Apache-ot, tégy elérhetővé egy videofájlt, ezzel a kiszolgálóoldalon meg is tettünk minden szükséges lépést. Amit hozzáad a VideoLAN, az annyi, hogy a VLC képes lejátszani az itt található fájlt anélkül, hogy az egészet letöltené.

Az egyszerűség annak köszönhető, hogy a HTTP protokollra bízzák a VOD módszer közben felmerülő gondok megoldását. Ez azt jelenti, hogy a HTTP 1.1 lehetőséget biztosít a `GET` művelet során a kiszolgálón lévő fájlokban történő pozícionálásra. (Ilyen pozícionálás történik akkor is, ha például befejezünk egy megszakadt letöltést.)

Ha tehát ezzel a megoldással videót nézünk, gyakorlatilag a webről töltünk le, csak épp nem írjuk lemezre a letöltött adatfolyamot, s eközben még ide-oda ugrálunk is benne.

Próbaképp tegyünk elérhetővé a gépünkön lévő webkiszolgálón keresztül egy videofájlt, lehetőleg valamilyen MPEG-et. Ez azért hasznos, mert szabványos és szakaszolható, s az sem árt, ha a többi módszerrel történő sugárzás esetén is mindig MPEG adatfolyamok továbbítására törekszünk.

Ha megvolnánk a fájlok közzétételével, az alábbi módon csatlakozhatunk az adatfolyamhoz VLC-vel:

```
vlc -vvv http://localhost/filmek/film1.mpg.
```

Mondanom sem kell, hogy a szolgáltatás sávszélesség-igénye mellett az eszközigénye is óriási. Itt ugyanis annyi szálon kell olvasni a merevlemezt, ahányan csatlakoznak hozzánk, annyiszor kell szétválasztani a kép- és hangsávokat, egyszóval minden művelet n-szereződik, arról nem is beszélve, hogy ebben a helyzetben a csoportos üzenetküldésnek még az elve is kizárt, hiszen nincs két egyforma csomag. A sok nehézségért cserébe viszont korlátlan szabadságot kapunk, mint tévénézők.

Tanulság

Ha jobban megnézzük, lehetőségeink szinte korlátlanok, s mindezt teljesen ingyen igénybe vehetjük. Kellő szakértelemmel egy valódi videóstúdió építhető néhány Linuxszal és a VideoLAN segítségével, amelyben talán az a legszebb, hogy elérhető közelségben van, bárki megvalósíthatja.

A dolog egyetlen szépséghibája, hogy közönség terén igencsak hadilábon állunk, erre valódi tévéműsort alapozni még nem lehet, ezért kezdetben maradni fog a helyi hálózatokon történő, ideiglenes, ám rendkívül hasznos próbálkozás, hogy aztán később szélesebb tömegekhez is eljuthasson a digitális adás, hiszen minden kétséget kizáróan ez a műsorszórás jövője.

Komáromi Zoltán
(komi@kiskapu.hu)
23 éves, a BME hallgatója, mellette PHP-programozóként dolgozik. Kedvenc területe a multimédia.

XMMS – ha többet akarunk tudni róla

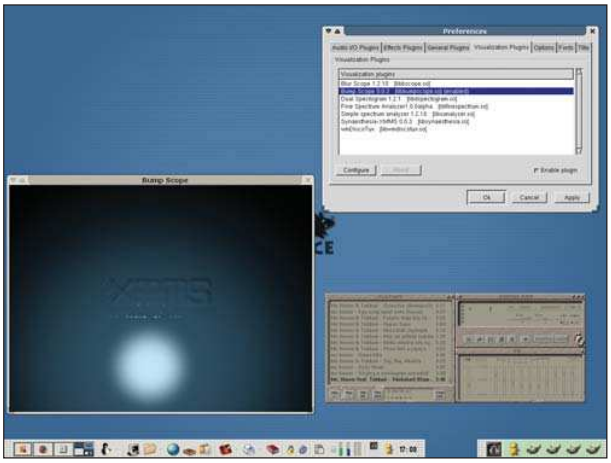
Bővítmények: fokozzuk a végtelenségig lejátszónk képességeit.

Bizonyára mindenki hallott már a Linux operációs rendszerek méltán népszerű audiolejátszó programjáról, az XMMS-ről (X-MultiMedia System). Ez a kicsi gyors, de annál sokoldalúbb program szinte minden linuxos rendszeren, minden terjesztésben megtalálható, s annyira jól sikerült, hogy nem is igazán akad vetélytársa. Senki sem törekedik arra, hogy újat írjon, hiszen ez is tökéletesen kielégít minden igényt. Ezenkívül egyszerű, kompakt, az alapvető működéshez nem igényel semmilyen különleges függvénykönyvtárat. Sikerességét tekintve az sem utolsó szempont, hogy a Windows alatt olyan nagy sikert aratott Winamp nevű program tökéletes mása mind a látvány, mind a működést illetően. Jogos a kérdés: ha csupán ennyi az igénye, mégis mitől olyan jó? Egyrészt azért, mert ez a leggyorsabb megoldás, de nemcsak Linuxon, hanem bármilyen más operációs rendszeren. Másrészt azért, mert a program működése, a külleme kivételével, szinte teljes egészében a különböző bővítményeken alapul, és sokkal jobban épít rájuk, mint a Winamp. Azt is mondhatnánk, hogy az egész program egy lejátszó kinézetű bővítménykezelő, amely mást sem tesz, csak az egyes bővítményeket használja. Talán ez az oka annak is, hogy senki sem akar másikat írni, ugyanis ez egy elég jó váz, és ha valakinek szüksége van valamilyen különleges lehetőségre, akkor elegendő csupán egy bővítményt írnia, amit azután könnyedén megetethet az XMMS-sel, és máris igénybe tudja venni az áhított szolgáltatást. Ezt mi sem bizonyítja jobban, mintsem az, hogy több száz elérhető bővítményt találunk kategorizálva az XMMS honlapján (➲ http://www.xmms.org/plugins.php).
Ezenkívül a legtöbb friss terjesztés részét képezik az XMMS mellett az egyes bővítmények is, ezeket legtöbb esetben a rendszer csomagkezelőjével egyszerűen hozzáadhatjuk. Az a tapasztalatom, hogy az emberek többsége nem is tudja, hogy milyen képességei vannak lejátszójának, nem ismeri a bővítmények rendszerét, ezért írásomban ebbe az irányba szeretnék egy kis kitérőt tenni az alapok bemutatása mellett. Mielőtt bővítmények után kutatnánk, nézzük meg, miként is illeszkednek az XMMS-hez.

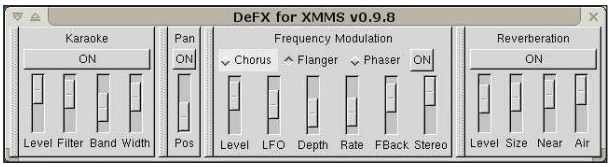
A bővítmények fajtái

Az XMMS alapvetően öt különböző csoportba sorolja a bővítményeket, feladatuknak megfelelően szétválogatva őket.

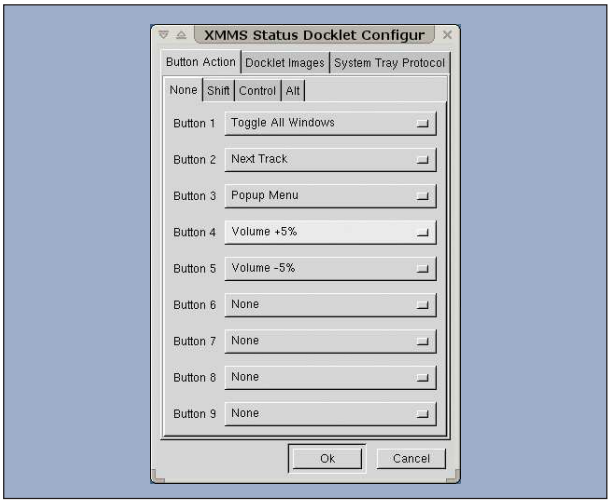
- Bemeneti bővítmények: feladatuk a lejátszani kívánt fájl (MP3, ogg, cdaudio) olvasása, dekódolása. Gyakorlatilag ezek határozzák meg, hogy milyen típusú fájlok lejátszására képes a programunk.
- Kimeneti bővítmények: ezeken keresztül hagyja el a dekódolt adatfolyam a lejátszót. A kimeneti bővítmények általában a hangkártyához csatlakoznak, s oda küldik az adatokat, de számos bővítmény létezik, amelyik valamelyik hangkiszolgálóhoz csatlakozik, így közvetve éri el a hangkártyát. Ezenkívül létezik olyan különleges kimeneti bővítmény is, amely hangkártya helyett fájlba írja a nyers WAV vagy egyéb formátumban lejátszott zenét.
- Általános bővítmények: mindenféle irányító és eseményvezérlő bővítmények tartoznak közéjük. Ilyen például a `lirc` bővítmény, amely lehetővé teszi az XMMS szabványos infravörös távirányítóval (amilyen a tévékártyákhoz is tartozik) történő vezérlését.
- Hanghatás- (audioeffect) bővítmények: ezek segítségével módosíthatjuk a kimenő hangfolyam akusztikai jellemzőit, visszhangot adhatunk hozzá, széthúzhatjuk a sztereócsatornákat, extra mélykiemelést tehetünk a zenére, vagy éppen az emberi beszédhangot eltávolítva karaoke-módban játszhatjuk le kedvenc számainkat.
- Látványelem-bővítmények: ide sorolhatók azok a bővítmények, amelyek a hang változásait, dinamikáját valamilyen grafikus módon ábrázolják. Ezek lehetnek mindenféle kivezérlésjelzők: látványos, futó „animációk” stb. Ezek többnyire az X-felületet használó 2D-s programok, de nem ritkák az OpenGL-t használó 3D-s megoldások sem.



1. kép A Bumpscope nevű látványelem-bővítmény bevetés közben



2. kép A DeFX nevű hanghatásbővítmény kezelőfelülete



3. kép A Docklet beállításai – a tálca ikonjának válasza az eseményekre

A bővítmények kezelése

Az XMMS meglehetősen egyszerűen kezeli a bővítményeket. Alapértelmezetten a */usr/lib/xmms/Plugins* könyvtárban található öt alkönyvtár a fenti kategóriáknak megfelelően. Ezekben találhatók az egyes bővítmények. Ha egy új bővítményre teszünk szert, és ez nem a rendszerünknek megfelelő, telepíthető formájú, elegendő, ha a *.so* fájlt, amely maga a bővítmény, bemásoljuk a megfelelő könyvtárba, majd újraindítjuk az XMMS-t. Ekkor beolvassa a benne található bővítményeket, és felkínálja őket használatra.

Azt is mondhatnánk, hogy ez önmagában kevés és elég szegényes, de szerencsénkre a fenti kezelési mód csupán a bővítmények bejegyzésére vonatkozik, azaz csak arra szolgál, hogy az XMMS indítás után lássa, mivel dolgozhat. Annak érdekében, hogy az egyes bővítményeket kényelmesen kezelhessük, testreszabhassuk, engedélyezzük vagy letiltsuk, az XMMS a beállításai között lehetőséget nyújt a bővítmények kezelésére, immáron grafikus formában. Ha előhívjuk a program beállítási paneljét, az ott található első négy fül az öt bővítménykategóriára vonatkozik. Az első fülön össze van fogva a ki- és bemeneti bővítmények kezelése. A felső listában láthatjuk, milyen fájlok lejátszására van lehetőségünk. Ezek közül a bővítmények közül mindig az az egy fog betöltődni, amelyik az általunk lejátszani kívánt fájlformátumot kezelni tudja, így mindig csak egy működik. Itt és minden későbbi bővítménylistára is igaz, hogy a lista alján található egy engedélyező, illetve tiltó jelzőbit, amellyel a bővítmény használatát szabályozhatjuk az XMMS számára. A mellette található *Configure* gombra kattintva hozhatjuk elő az adott bővítmény beállításait. Amit itt állítunk, az csakis a pillanatnyi bővítményre vonatkozik, s nem hat a többire. A beállítások azonnal érvényre jutnak, még csak az *OK* megnyomására sincs szükség, kivéve, ha ki- vagy bemeneti bővítménnyel akad dolgunk, ekkor ugyanis csak a következő szám lejátszásakor lépnek életbe a változtatások.

A kimeneti bővítmények esetén kissé más a helyzet. Mivel a ki- és bemeneti bővítményekből egyszerre egy-egy lehet aktív, egyértelművé kell tenni, hogy mikor melyik kimenetet választjuk, ezért itt nem lista látható, hanem egy lenyíló menü, amelyből mindig a kiválasztottat használja a program, egyébként a beállításokra ugyanaz érvényes, mint a többiek esetében.

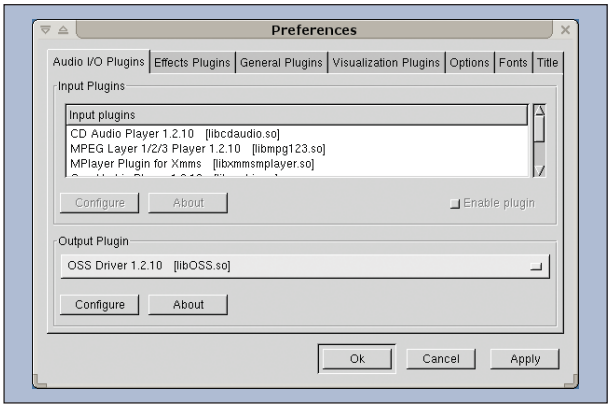
Most, hogy többé-kevésbé megismerkedtünk a bővítménykezeléssel, nézzünk szét egy kicsit, hogy milyenekkel találkozhatunk a mindennapokban.

Bemeneti bővítmények

Mivel az XMMS alapvetően MP3-lejátszónak indult, természetesen alapértelmezetten telepítve van az MPEG 1/2/3 bővítmény, s emellett a másik terjedőben lévő formátum, az ogg lejátszásához szükséges ***Ogg Vorbis Player*** könyvtár is. Ezzel nagyjából le is fedtünk mindenféle formátumot, amelyet az ember kilencven százalékban hallgat, na de mi történik a maradék tízzel? A válasz egyszerű – szinte mindegyik formátumra létezik `xmms` bővítmény (plugin), csak össze kell vadászni őket, aztán fordítani, telepíteni és készen is vagyunk. Ez alapvetően macerás munka, mivel a legtöbb bővítmény forrás formájában áll rendelkezésre, így előtte le kell fordítani, ám jelentős részük le sem fordul, mert írásmódbeli (vagy durvább) hiba rejtőzik a kódban. Szerencsére erre is akad megoldás. Bizonyára mindenki hallott már a magyar fejlesztésű MPlayerről, ami többek között arról híres, hogy szinte minden kép- és hangformátum lejátszására képes. Innen már sejthető a megoldás lényege: létezik egy XMMS-MPlayer nevű bővítmény, amely kapcsolatot teremt a két lejátszó között, így egyetlen bővítmény használatával elérjük, hogy mindenféle hangfájlt képesek legyünk meghallgatni az XMMS-ből, mégpedig áttetsző módon. Eközben mellékhatásként elérjük, hogy akár videókat is lejátszhatunk, ahogy azt a legújabb 3-as, illetve 5-ös változatú Winamp programok esetében láthatjuk, azaz egyúttal kaptunk az MPlayerhez egy grafikus előlapot (frontend) is. Nézzük először ezt a bővítményt!

Xmms-MPlayer

A telepítéshez mindenekelőtt szükségünk lesz egy működő MPlayerre a rendszerünkön, de ez manapság szinte mindenkinél fenn csücsül, szóval ezzel valószínűleg nem lesz gond. Továbbá telepítenünk kell még a `libglib` és a `libgtk` fejlesztői csomagokat. Ez a két csomag egyébként elengedhetetlenül szükséges minden XMMS bővítmény fordításához. Látogassunk el a ➲http://xmmsmplayer.sourceforge.net/ címre, s töltsük le a program forrását. Ha ez megvolna, csomagoljuk ki az állományt a `tar -xzvf <állomány neve>` paranccsal, és lépjünk be az XMMS-MPlayer forrásának könyvtárába. Adjuk ki a `./configure` parancsot, s ha mindent rendben talál, akkor a `make`, majd a `make install` parancsok kiadásával fordítsuk le a programot, illetve telepítsük a bővítményt. Ha mindegyik parancs végrehajtása sikeres volt, azaz nem kaptunk hibaüzenetet, akkor indítsuk el az XMMS-t, s a beállítások panelen ellenőrizzük, hogy szerepel-e a bemenetek között az XMMS-Mplayer nevű bővítmény.



4. kép Az XMMS beállítópanelje – minden, ami szem szájnak ingere

Kattintsunk a bővítmény beállításaira. Mint látjuk, ez meglehetősen szegényes, de emiatt egyáltalán ne fájjon a fejünk, ugyanis egyrészt az MPlayer használja a saját könyvtárban (***/home***) elhelyezett beállításfájl jellemzőit, másrészt itt a bővítmény beállításai között adhatunk meg parancssoros, tetszőleges kapcsolókat egy beviteli mezőben, így gyakorlatilag bármilyen beállítási lehetőséget elérhetünk ugyanúgy, mintha parancssorból futtatnánk a programot. Nézzük meg, mit is kaptunk. A lejátszási listához adjunk hozzá egy filmet vagy egy WMA (Windows Media Audio) formátumú zenét. Indítsuk el az XMMS-ben, mintha csak zenét hallgatnánk. A film esetében felpattan az MPlayer ablaka, s elkezdődik a lejátszás, közben pozícionálhatunk az XMMS csúszkájával, de az MPlayer esetében megszokott gyorsbillentyű-gyűjtemény is elérhető. Hasonló a helyzet a WMA típusú hanggal, azzal a különbséggel, hogy itt nem jelenik meg a képablak. Egyetlen aprócska különbség az MP3-lejátszáshoz képest, hogy nem működik a beépített hangszínszabályzó, és nem látszik a hangképelemző sem, tehát minden ugyanolyan, mintha audiokorongot játszanánk le. Ezen kívül minden szolgáltatás működik. Mostantól képesek vagyunk az AC3-tól kezdve egészen az ASF formátumú hangig bármit lejátszani, és ehhez jön még az összes videoformátum, amit az MPlayer támogat, s mint tudjuk, ebből jó sok létezik. Ezenkívül bátran a szívünkre tehetjük a kezünket, hogy tisztelegjünk az MPlayer fejlesztői előtt, akik javarészt hazánk fiai, s segítségükkel most lehetővé vált, hogy egyetlen bővítmény telepítésével megoldjuk a különböző fájlformátumok használatával járó nehézségeket.

Tone Generator

Mielőtt még áttérnénk a többi bővítményre, hadd mutassak be egy kevésbé hasznos, ám mulatságos darabot. A Tone Generator nevű bővítmény alapértelmezetten része az XMMS-nek, és segítségével különböző rezgésszámú (frekvenciájú) hangokat csalhatunk elő, párhuzamosan akár többet is. Használatához adjuk hozzá a következő URL-t: ***tone://1000***. Ez egy 1 KHz-es sípolást eredményez a hangszórónkon. Több ilyen sípolás összegét a ***tone://frekvencia1,frekvencia2,frekvencia3,...*** URL hozzáadásával és lejátszásával érhetjük el. Ez a gyakorlatban akkor lehet hasznos, ha például valaki az erősítőjét vagy egy magnófelvevőjét kalibrálja, s egy egyenletes rezgésszámú hangra van szüksége hozzá.

A kimeneti bővítmények

Az alapértelmezett kimeneti bővítmény az OSS-meghajtó, amely közvetlenül a hangkártyánknak küldi a hangfolyamot. Ha mi OSS helyett ALSA-n keresztül használjuk a hangkártyánkat (így fordítottuk a rendszermagot), akkor szükségünk lesz az ALSA bővítményre, amely a ➲ http://savannah.nongnu.org/projects/alsa-xmms/ címről tölthető le. Nem biztos azonban, hogy ezt le kell szednünk, ugyanis például SuSE alatt alapértelmezetten része az XMMS-nek.

eSound és az aRts

Előfordulhat, hogy valamelyik hangkiszolgálót szeretnénk használni. A két legelterjedtebb példány a gnome-os eSound és a kde-s aRts. Ezek használatához azonos nevű XMMS bővítményekre lesz szükségünk. Az eSound alapértelmezetten része az XMMS-nek, az aRts jelenléte viszont változatfüggő. Legtöbb Linux-változat tartalmazza, némelyikben az XMMS telepítésekor ez is felkerül, de ha valaki szerencsétlenebb helyzetben van, akkor a ➲ http://www.xmms.org/plugins.php?details=85 oldalról mind forrás, mind RPM csomag formájában letöltheti és telepítheti.

Crossfade

Ezeknél talán érdekesebb lehet az egyik kedvencem, a Crossfade nevű bővítmény, amely a ➲ http://www.xmms.org/plugins.php?details=167 címen érhető el, sajnos csak forrás formájában. Letöltés után pontosan ugyanúgy kell eljárnunk, mint az XMMS-MPlayer telepítésénél. A bővítmény egyébként a lejátszás időbeli folyamatosságát javítja, s teszi szebbé. A számok végén átkeveri az egyiket a másikba, két szám lejátszása között nem tart szünetet, indításkor és lejátszáskor pár tizedmásodperc alatt felhangosítja, illetve lehalkítja a zenét, pillanat állj (pause) esetén is hasonlóan jár el, egyszóval a fülnek kellemes módon kezeli a zeneszámokat. Megadhatjuk, hogy a kimeneti modulnak mi legyen a kimeneti modulja, azaz láncba fűzhető, így bármelyik kimenttel (OSS, ALSA, aRts, eSound) használható. Ha előhozzuk a beállításait, láthatjuk mi mindenre képes, javaslom kitapasztalásához, mindenki játszogasson el vele kedve szerint.

Általános bővítmények

Alapértelmezetten három bővítményünk van. Az egyik az IRman, amely lehetővé teszi, hogy a tévékártyánkhoz kapott távirányítóval az XMMS-t is vezérelhessük. Ha pedig már ilyen igényeink merültek fel, szinte természetes az is, ha a géphez csatlakoztatott botkormánnyal szintén szeretnénk ugyanezt megtenni. Erre ad lehetőséget a Joystick Control nevű bővítmény. Ezen felül még egy érdekes bővítményünk van, melynek neve: Song Change. Ez semmi mást nem csinál, mint azt, hogy minden szám váltásakor elindít egy héjparancsot. Ezt gyakorlatilag bármire lehet használni, ha valami magától működő dolgot szeretnénk elérni. Ezen túlmenően természetesen mindenféle vicces és hasznos bővítmény áll rendelkezésünkre, nézzünk néhány példát rájuk is.

Xmms-GoodNight

A nevéből gondolom már mindenki ki is találta, hogy miről van szó: egy olyan bővítményről, amellyel beállíthatjuk,

hogy a lejátszó adott időben, adott idő elteltével, vagy a lejátszási lista végére érve leállítsa a lejátszást, vagy zárja be a programot, esetleg kapcsolja ki a gépet. Eközben lehetőségünk van folyamatos halkításra, megadhatjuk, hogy másodpercenként hány százalékkal csökkentse a lejátszás hangerejét, így mielőtt kikapcsolná a lejátszást vagy épp a gépet, folyamatosan elkezdi csökkenteni a hangerőt, nehogy a hirtelen leállítással megzavarja nyugodt álmunkat. Mindezeket a bővítmény beállítópanelén módosíthatjuk, s itt adhatjuk meg többek között azt is, milyen paranccsal küldje a gépünket készenléti üzemmódba, vagy milyen utasítással kapcsolja ki. A program a ➲ http://fiktiv.szgtikol.kando.hu/~folti/src/ oldalról tölthető le forrás formájában.

Xmms-Status-Plugin

Ez a bővítmény a KDE és Gnome, vagy ezekkel megfelelő (compatible) munkakörnyezetek paneljén található tálcához (icon tray) ad hozzá egy ikont, melynek segítségével láthatjuk a futó állapotban lévő programunkat. A dolog szerencsére itt nem merül ki, ugyanis megadhatjuk, hogy az ikonra történő kattintáskor, gombtól függően mi történjen (hangerőszabályzás, bezárás, menü, léptetés stb.). Az ikon egyébként minden állapotban (pillanat állj, lejátszás, üresjárat) más és más. Ez szintén olyan bővítmény, amit én már jó ideje használok.
A bővítmény honlapja a ➲ http://www.hellion.org.uk/xmms-status-plugin/ címen érhető el.

Hanghatásbővítmények

Itt is találunk néhány alapértelmezett bővítményt, amellyel visszhangot vagy épp 3D-s hanghatást állíthatunk be, s van még egy karaoke lehetőséget biztosító énekhang-eltávolító szűrőnk is, amellyel jó eredménnyel szedhetjük le az emberi beszédet a zenéről. Érdekesebb dolgokra azonban csak akkor akadhatunk, ha néhány egyéb egyszerű bővítményt keresgélünk.

Xmms-Volnorm

Bizonyára sokunknak okoz gondot, hogy egyes zeneszámok bizonyos részeken nagyon halkan szólnak, máshol meg elviselhetetlenül hangosak. Nem ritka az sem, hogy egy lejátszási listában a különböző zeneszámok eltérően hangosak, így ha egyenletességet szeretnénk elérni a hangerőt illetően, kénytelenek vagyunk folyamatosan igazítani a hangerőszabályzón. Erre nyújt megoldást a fenti bővítmény, amely egyenletes hangerőre hozza a zenéket: a halk részeket erősíti, a hangosakat csillapítja. A bővítmény a ➲ http:///volnorm.sourceforge.net/ oldalról tölthető le forrás formájában.

Xmms-DeFX

Ennek a hangfeldolgozó bővítménynek a segítségével aprólékosan szabályozott hanghatásokat hozhatunk létre. Karaoke, moduláció és visszhangosítás található benne. Ezek természetesen egymással párhuzamosan is működnek, így kombinálva őket számtalan érdekes hatást hozhatunk létre, az egyes hatások jellemzőit függőleges csúszkákon állíthatjuk be, mintha csak egy keverőpulton dolgoznánk. A bővítmény a ➲ http://www.igloo.cl/~bruno/xmms-defx/ címről tölthető le.

Látványelembővítmények

Talán ezen a területen a leggazdagabb a bővítmények választéka. Számtalan, jobbnál-jobb kinézetű bővítményt találunk akár a rendszerünk csomagkezelőjében, akár az interneten. Az XMMS két látványelem-bővítményt tartalmaz, mindkettejük meglehetősen egyszerű, ám kis gépigényük miatt rendkívül gyorsak és látványosak. Elindításuk úgy lehetséges, mint például a hanghatások alkalmazása: engedélyezni szükséges az adott bővítményt. Ha ez megtörtént, azonnal felpattan egy ablak, és ha a zenelejátszás éppen zajlik, akkor láthatjuk is, mit „rajzol”. Közben vessünk egy pillantást a főablakra: láthatjuk, hogy a beépített hangképelemző továbbra is működik, nem hagyja abba – mint azt a Winamptól megszokhattuk –, ha elindítunk egy látványelemet. Ezen felbátorodva engedélyezzük gyorsan a másik bővítményt is olyan módon, hogy az előzőt nem tiltjuk. Most pedig tessék csodálkozni, ugyanis egyszerre megy mind a kettő. Most nyugodtan keressünk egy windowsos Winamp-felhasználót, s elégedett vigyorral az arcunkon nézzünk mélyen a szemébe. Itt ugyanis a nagy vetélytárssal ellentétben egyszerre n darab, úgyis mondhatnám tetszőleges számú látványelem-bővítmény futhat egyszerre. Tekintettel arra, hogy nagyon sok ilyen látványelem létezik, s mivel használatukat tekintve szinte nincs semmi különbség közöttük, ezért csak néhány egyéb bővítményről szeretnék szót ejteni.

Xmms-Synaesthesia

Egy rendkívül látványos, teljes képernyős módban is futó bővítmény, amely ráadásul személyes kedvencem. Sok beállítási lehetősége ugyan nincs, de nem is nagyon van rá szükség. A ➲ http://staff.xmms.org/zinx/xmms/ lehet letölteni.

Xmms-Bumpscope

Egy 3D hatású, de valójában 2D-s, teljes képernyős módú hanképelemző. Látványosnak látványos, de teljes képernyőn és nagy méretben meglehetősen lassú.
A ➲ http://staff.xmms.org/zinx/xmms/ címen találjuk a letöltési lehetőséget.

Végezetül

A bővítmények nagy száma miatt semmiképp sem lehet mindegyiket bemutatni, a cikket nem is részletes bemutatónak, hanem figyelemfelkeltőnek szánom, s abban reménykedem, hogy sokan nekifognak és hatalmas bővítményvadászatba kezdve, jól felokosítják a gépükön futó XMMS-t. Mielőtt azonban elkezdenénk a cserkészést a Google-on, javaslom, hogy nézzük meg Linux-változatunkban milyen bővítmények állnak rendelkezésre, így elkerülhetjük a felesleges fordítást és telepítési macerát. Csak akkor fogjunk bele a kézzel történő hozzáadáshoz, ha az elérhető csomagok között nem találtuk meg a keresett darabot. Sok szerencsét és még több kitartást kívánok mindenkinek!

Komáromi Zoltán
(komi@kiskapu.hu)
23 éves, a BME hallgatója, mellette PHP-programozóként dolgozik. Kedvenc területe a multimédia.

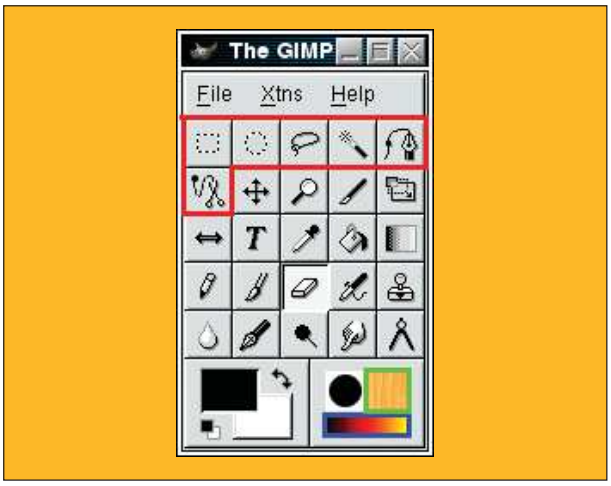
A Gimp rejtelmei (1. rész)

Sorozatunk segítségével a Gimp titkaiba nyerhetnek bepillantást.

A hogyan az már linuxos körökben általánosan ismert nézet, a Gimp egy pixelgrafikus rajzoló- és képfeldolgozó program, melynek fejlesztése több éve zajlik. Olyannyira kidolgozott program, hogy létének köszönhetően sok más linuxos, grafikus felületet használó alkalmazás is megvalósulhatott. Itt elsősorban a GTK+ eszközkészletre gondolok, hiszen a Gimp fejlesztése során vált szükségessé kialakítása.

Gyakran olvashatunk olyan leírásokat, amelyekben a Gimpet az Adobe Photoshoppal hasonlítják össze. Ezek a leírások sajnálatos módon még nemrégiben is azt hangsúlyozták, hogy a Gimp nem alkalmas nyomdai előkészítésre, mert nem ad lehetőséget a CYMK színrendszer szerinti színre bontás elvégzésére. Nos, bár nem vagyok nyomdász, de sorozatunk következő részeiben meg fogom mutatni, hogy a Gimpben is lehetőség nyílik e színrendszer használatára. A Gimpet alapvetően Linux alatt kezdték el fejleszteni, majd – ahogyan a Blender esetében láthattuk – később vált alkalmassá más operációs rendszereken való működésre. Az előbbiek alapján azt mondhatjuk, hogy a Gimp annyival többet nyújt a vetélytársánál, hogy többféle rendszer alatt működőképes és természetesen az alapelgondolást megőrizve, jelenleg is ingyen használható.

Amikor a Gimpet a telepítés után elindítjuk, első lépésben egy telepítővarázsló segítségével elvégezhetjük az alapvető beállításokat. A program felépíti a helyi felhasználó könyvtárában a saját könyvtárszerkezetét, majd beállíthatjuk, hogy mekkora memóriát használjon a visszavonási és egyéb képműveletek során. Amikor a program ezt a memóriamennyiséget felhasználja, a merevlemezen átmeneti tárat hoz létre, melynek helyét szintén ebben a varázslóban állíthatjuk be. A beállítások harmadik lépése, a képernyő felbontásának meghatározása. Itt a felbontás alatt azt kell érteni, hogy egy inchen (2,54 cm) mennyi képpontot szeretnénk ábrázolni. A Gimp le tudja kérdezni az X-kiszolgáló megfelelő jellemzőit, tehát amennyiben DDC-megfelelő monitorral rendelkezünk, a méretét az X-kiszolgáló meg tudja határozni, majd a képpont/inch értéket ennek megfelelően állítja be. Ha a beállítás mégsem felel meg, a Gimp lehetőséget ad a kézi beállításra is. Ekkor a ***Kalibrálás*** (Calibrate) gombot kell használnunk. A program kirajzol egy vízszintes és egy függőleges vonalzót, majd felkér bennünket, hogy a vonalzóval mérjük meg a kirajzolt ábrák méreteit. Ezeket az értékeket megadva a program elvégzi a beállítást és ezzel az utolsó lépéssel használatba is vehetjük.



1. kép Területek kijelölésének eszközei

Már első induláskor is feltűnhet, hogy a program rengeteg beállítási lehetőséggel bír. Most ezekből szeretnék ismertetni néhány lényegesebbet. A program eszközkészlete általában a képernyő bal felső részén jelenik meg, és amíg nincs megnyitva semmilyen fájl, itt találhatjuk az általános működésre vonatkozó beállításokat és eszközöket.

A ***Fájl/Beállítások*** (File menü/Preferences) menüpontot kiválasztva beállíthatjuk, hogy ha új képet hozunk létre, milyen tulajdonságokkal rendelkezzen. Megadhatjuk a méretét és a színmélységét, majd beállíthatjuk, hogy legfeljebb mekkora legyen a kép által felhasznált memória. Ahogy a bal oldali fanézetben lejjebb haladunk, a következő érdekesség a ***Súgó*** (Help System). Ebben a helyzetérzékeny súgó használatát állíthatjuk be az F1 billentyűre, és megadhatjuk, hogy az alapértelmezett belső böngészőt szeretnénk használni, vagy a Netscape jelenítse meg a súgóoldalakat.

A ***Környezet*** (Environment) pontban választhatjuk ki, hogy a program milyen eljárást használjon a képek átméretezésénél a minőségromlás csökkentésére. Itt a ***A legközelebbi szomszéd*** (Nearest Neighbor) adja a leggyorsabb eredményt, viszont ennél a beállításnál látszik leginkább a kép minőségének romlása. A ***Köbös*** (Cubic) lassú megoldást jelent, de a jelenleg használatos gépek teljesítménye mellett ez már nem jelent látható lassulást. Ebben az esetben a Gimp a képek nagyításakor a közbenső képpontokat

a szomszédos képpontok figyelembevételével, interpolációval számítja ki. Ugyanitt állíthatjuk be a visszavonási lehetőségek számát, és hogy a program milyen esetekben végezzen tényleges mentést. Feltételezhetjük, hogyha a kép nem módosult, akkor nem próbáljuk meg menteni, ennek figyelésével pedig a programot is megbízhatjuk, vagyis beállítható, hogy minden esetben történjen-e lemezművelet, vagy csupán akkor, ha a képen valóban történt változás. Fontos lehet a következő beállítási lehetőség – ezt a ***Munkafolyamat*** (Session) felirat kiválasztásával érhetjük el –, ugyanis itt tudjuk rávenni a Gimpet, hogy az ablakok helyzetét mentse, és mindig próbálja meg visszaállítani a tárolt állapotot. A továbbiakban meghatározhatjuk még a különféle felhasználó által telepített vagy létrehozott kiegészítők, ecsetek, minták, színkészletek és átmenetek tárolására szolgáló könyvtárak helyét, amelyeket a program az első indításkor a felhasználó saját könyvtárában létrehozott és a ***gimp-1.2*** elérési úton tárol.

Miután az alapismeretekkel felvértettük magunkat, folytassuk az ismerkedést az alapvető navigációs lehetőségekkel. A szerkesztett képet az egérgörgővel függőlegesen tudjuk görgetni, a CTRL billentyű lenyomása mellett pedig az egér görgője vízszintesen mozgatja a képet. A görgőt a SHIFT billentyűvel együtt használva a nagyítás arányát tudjuk meghatározni. A képen való mozgás másik eszköze a szerkesztett képet mutató ablak jobb alsó sarkában található kis négyirányú nyíl. Erre kattintva a kép kicsinyített másán tudjuk meghatározni az ablakban megjeleníteni kívánt tartalmat.

Most pedig térjünk át a kiválasztási lehetőségek megismerésére. Vélhetőleg az olvasó is tisztában van azzal, hogy egy pixelgrafikus rajzolóprogramban milyen nagy a jelentősége annak, hogyan tudjuk kiválasztani a feldolgozásra szánt képrészleteket. A Gimp eszközkészletében is öt ikon szerepel, amelyek csak a kiválasztás mikéntjének meghatározására használatosak. Ezeket láthatjuk az *1. képen* piros keretbe foglalva.

A bekeretezett gombokat szemlélve, balról jobbra haladva, a következőkkel találkozhatunk:

- ***Téglalap alakú terület kiválasztása***: az egérgomb lenyomása után a CTRL billentyűvel együtt használva a kijelölés az egérmutatóhoz képest középpontos lesz, míg a SHIFT billentyűvel négyzet alakú területet tudunk kijelölni.
- ***Ovális terület kijelölése***: a fenti kombinációk szintén érvényesek.
- ***Kijelölés szabadkézi*** rajzolással.
- ***Varázspálca***: hasonló színű képpontok kijelölésére használható. Ha a gombra duplán kattintunk, a megjelenő párbeszédablakban (a ***Küszöbszint*** (Threshold) értéknél) a hasonlóság küszöbszintjét adhatjuk meg.
- Területek kiválasztása ***Bezier-görbék*** segítségével: a görbék vezérlőpontjait helyezzük el egérkattintással, majd a görbületet meghatározhatjuk, ha lenyomott egérgomb mellett mozgatjuk az egeret. Mindig csak zárt görbével határolt területet tudunk kiválasztani, ennek meghatározását pedig a terület belsejébe való kattintással tudjuk véglegesíteni.
- Kiválasztás ***közelítéssel***: ez a megoldás a varázspálca és a szabadkézi kiválasztás kombinációjának tekinthető, ugyanis a kívánt területet nem szükséges pontosan körberajzolni, csupán vonalakat adunk meg, a Gimp ennek alapján próbálja meg az irányvonalakat a kép tartalmának megfelelően pontosítani.

Mindemellett a program lehetőséget nyújt arra is, hogy a már kijelölt területekhez hozzáadjunk vagy elvegyünk belőlük részeket. Ha valamelyik eszközt használjuk, és a CTRL billentyűt lenyomjuk, a Gimp a kijelölt területeket ponthalmaznak tekintve összegzi. Amennyiben a SHIFT billentyűt ütjük le, a program a területek különbségét képezi. A teljes képtartalom kiválasztására a CTRL-A billentyűk használhatók, és bármilyen kijelölésre vonatkozó adat a SHIFT-CTRL-A billentyűkkel törölhető. A meglévő kijelölt területek ellenkezőjét a CTRL-I billentyűkombinációval választhatjuk ki.

Természetesen ez még nem minden. A képen a jobb egérgomb segítségével hívható elő a menü, melynek ***Kijelölés...*** (Select...) menüpontja további lehetőségeket kínál számunkra. Egyelőre elegendő lesz megismerkedni közülük a ***Csökkentés...*** (Shrink...) lehetőséggel, ezzel adott képpontnyi mértékben csökkenthetjük a területet, továbbá a ***Növelés***-sel (Grow...), ami értelemszerűen a terület növelésére használatos. Ide tartozik még a ***Keret...*** (Border...) lehetőség is, mellyel adott szélességű keretet készíthetünk a kijelölt terület felhasználásával.

Ebben a menüben bukkanhatunk egy további érdekes lehetőségre, amit a ***Szín szerint*** (By color) menüponton keresztül érhetünk el. Ide kattintva megjelenik egy párbeszédablak, amiben a kiválasztott területeket láthatjuk. Ha a képre bökünk, a program az egész képen megkeresi a hasonló színű pontokat, és ezek fogják alkotni a kijelölt területet. A párbeszédablakban meghatározhatjuk az érzékenységet (a szokásos ***Küszöbszint*** (Threshold) mezőben), továbbá azt is, hogy az újabban kiszemelt színek hogyan kapcsolódjanak a már előzőleg kiválasztott területekhez. Itt is lehetőségünk nyílik összeadás, kivonás és csere végrehajtására. Ismerkedjünk meg a színek beállításának lehetőségeivel is. Képünkön az eszköztár bal alsó sarkában látható az éppen érvényes rajzoló- és háttérszín. Ezek valamelyikére duplán kattintva előbukkan egy színpaletta, amelyről kiválaszthatjuk a használni kívánt színt. Ha a szokásos Gimp-paletta nem felel meg, akkor vízfestékszerű palettáról, a HSV színtérhez közelebb álló háromszöggel meghatározottról választhatunk, vagy a GTK-ban megszokott megoldást is alkalmazhatjuk. Ezeket a színválasztó ablak fülecskéire kattintva érhetjük el. A színek meghatározása után a kiválasztottakat felcserélhetjük a mellettük található kétirányú nyilacska használatával.

A színek mellett meghatározhatjuk a kitöltési mintát is, amit a képünkön zöld kerettel jelölt területre történő dupla kattintással érhetünk el. Itt lehet – a kék kerettel jelölt területen dupla kattintással – a használni kívánt színátmenetet megadni. Az eszköztár alsó részén található vezérlők közül az ecset meghatározására szolgáló mezőt még nem vettük sorra, ezt szintén kettős kattintással kelthetjük életre. Természetesen a feliratot is módosíthatjuk, azonban ennek lehetőségeit a további részekben szeretném bemutatni. Addig is kellemes ismerkedést és jó szórakozást kívánok!

Fábián Zoltán

Debian otthonra (6. rész)

Amit látsz, azt kapod? – Nyomtassunk végre valahára néhány oldalt!

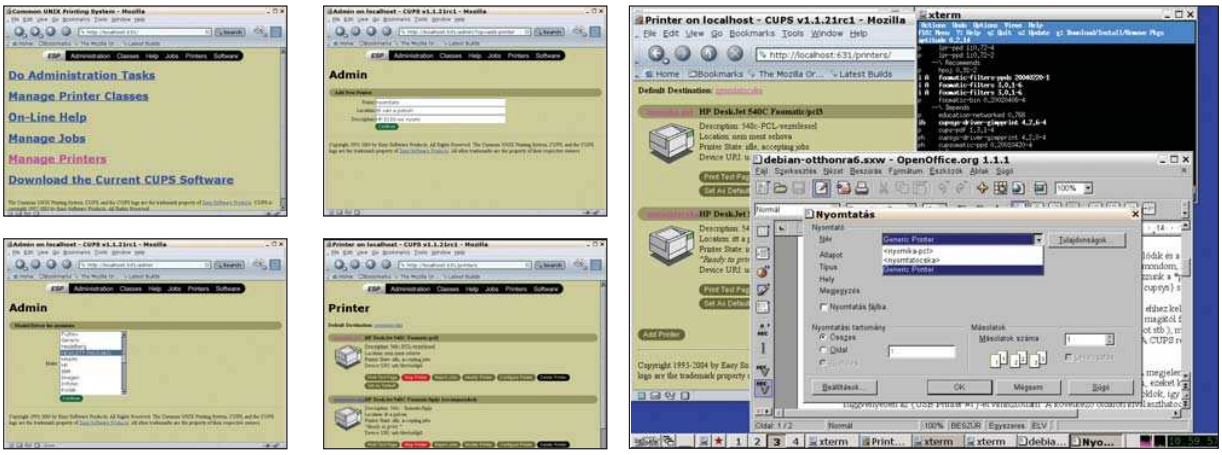
Sorozatunk előző részében felvérteztük magunkat egy kis elmélettel, lássuk hogyan boldogulunk a gyakorlatban. Ha nyomtatásvezérlést szeretnénk a gépünkre, lényegében két programcsomag közül választhatunk: az egyik az ***lpr***, a másik pedig a ***cupsys***. Az ***lpr*** „régi bútordarab" már a nagygépes időkben is jelen volt, így az lpr-parancsok szinte ipari szabványnak tekinthetők, bármelyik egyéb nyomtatásvezérlő rendszert is használjuk, nyugodtan számíthatunk rájuk (`lp`, `lpq`, `lprm`, esetleg `cancel` és `lpstat`. Ezekről később bővebben is szólunk).
A másik ilyen hozomány a */etc/printcap* fájl. Ennél a rendszernél a ***printcap*** állományban lehet megadni a nyomtatókat (nyomtató neve, eszköznév, be- és kimeneti szűrők, egyéb beállítások).
A másik testvér, a CUPS vagy ***cupsys***, mely aránylag új, de rendkívül gyorsan fejlődik és a használata is könnyebb (szerintem), ezért én a ***cupsys*** mellett teszem le a voksomat. Miért mondom, hogy könnyebb használni? Nos, a webes kezelőfelület folytán, továbbá a ***printcap*** állománnyal sem kell foglalkoznunk. Nézzük, mi is szükséges ahhoz, hogy életre keltsünk egy nyomtatót a ***cupsys*** segítségével!
Először is telepítsd a nyomtatásvezérlőt (továbbá jól jön majd egy böngésző is): ehhez kellenek a `cupsys`, `cupsys-bsd`, `cupsys-client` csomagok (illetve amiket még magától feltesz, például a gimpprinthez tartozó vezérlőt, a ***foomatic***-ot vagy helyette a ***cupsomatic***-ot stb.). Ezek után a böngészővel látogass el saját géped 631-es kapujára (***http://localhost:631***/). A CUPS rendszer fogad (lásd az *1. ábrát*).
Innen a menüből válaszd a ***Printers*** pontot, majd azon belül az ***Add new printer***-t. A megjelenő ablakban (lásd *2. ábra*) a leendő nyomtató nevét (két szöveges leírómező is található, ha van kedved, kitöltheted őket), majd az eszközt add meg. Én egy USB-csatolós HP 5150-et birtoklok, ezért az ***USB Printer #1***-et választottam. A következő oldalon kiválaszthatod a gyártót, egy hosszú listát kapsz az elérhető nyomtatóvezérlőkről (a gépen található meghajtók és leképezési módok stb. függvényében). Ha olyan szerencsés vagy, hogy a nyomtatódhoz nem találsz pont megfelelő vezérlőt (na, kell-e mondjam, hogy jelenleg a HP Deskjet 5150-es is ilyen nyomtató), akkor válassz egy „legjobb közelítést". Nálam például ez az 540C sorozat. Érdemes az ajánlott (***recommended***) vezérlőt választani.
Ha ez megvan, már telepítetted is a nyomtatót, és a ***Print test page*** gombbal egy próbanyomtatást készíthetsz. Ha pedig a nyomtató vad zakatolásba kezd (vagy zümmögésbe, duruzsolásba, nyikorgásba, gyártótól és modelltől függően), akkor kényelmesen hátra is dőlhetsz, amíg a próbanyomtatás eredménye meg nem jelenik. Ekkor ugyanis sok ember komolyan gondolkodóba esik. Vajon miért nem színes az a fránya nyomat? És miért nem olyan szépek a betűk, mint amilyennek lenniük kellene? Ha nem vagy megelégedve a nyomtatás minőségével, több dolgot is tehetsz:

a) a webes felületen keresztül próbálod meg finomhangolni,
b) ugyanezt egy részletesebb beállításokkal dolgozó programból teszed,
c) más vezérlőt használsz, valamint
d) hivatkozol az ősi Unix-időkre, ahol már az is csoda volt, ha színesben tudtak nyomtatni, majd büszkén kihúzod magad, hogy lám-lám, milyen gyorsan fejlődik a világ. Ezt a megoldást hálózati rendszergazdák szokták választani, amikor a titkárnő panaszkodik, hogy egy hónapja még szépen tudott nyomtatni a gép.

A próbálgatások jó ideig eltarthatnak, akármelyik megoldást is választhatod – ugyanazt a nyomtatót több néven is feltelepíted, különböző vezérlőkkel (és beállításokkal), így lényegesen gyorsabban tudod a kívánt nyomtatási módot használni (lásd az *5. ábrát*).
Egy gyors szösszenet erejéig azért visszatérek az USB-csatolós nyomtatók gyönyörűségéhez. Mint ahogy azt írtam, az egész rendszer folyamatos fejlődésben van, olyannyira, hogy gyakran saját magával is ellentmondásba kerül. Például így lehetséges, hogy az USB-eszközök elérése is többször változott, sőt a CUPS nem is mindig tudja megállapítani helyesen, hol is kell keresse az illetékes nyomtatót. Próbálkozik, próbálkozik, például figyeli milyen adatokat kap az USB-alrendszertől:

```
Apr 16 10:12:10 localhost kernel: hub.c: new USB
➥device 00:13.0-1, assigned address 4
Apr 16 10:12:10 localhost kernel: printer.c:
➥usblp0: USB Bidirectional printer dev 4 if 0
➥alt 0 proto 2 vid 0x03F0 pid 0x6204
```

Ebből úgy gondolja, hogy az eszköz neve valószínűleg `/dev/usblp0`. A webes felületen ezt ki is írja nekünk, mi meg csak várjuk, hogy nyikorogjon a kis drága, de csak nem szólal meg. Pedig várja az adatokat, hiszen élete értel-

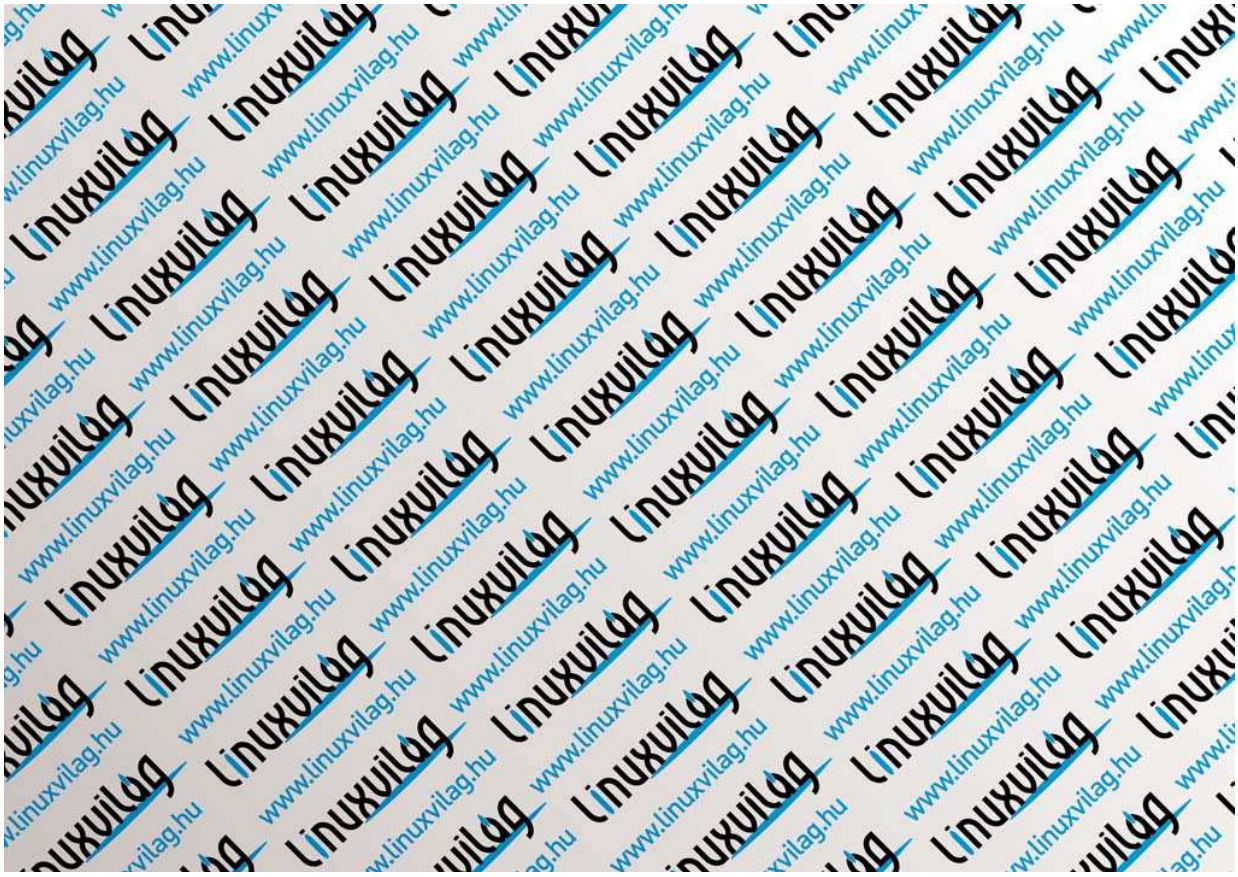


me a nyomtatás, csak éppen a `/dev/usb/lp0` alatt. Nos, a gond könnyen orvosolható, ha egy hivatkozást készítesz a `/dev` könyvtárba: `cd /dev; ln -s usb/lp0 usblp0`. Ugye mennyire magától értetődő? Merje nekem ezek után bárki azt mondani, hogy a Debian nem felhasználóbarát rendszer! Ha már itt tartunk, érdemes megemlíteni, hogy a HP az OfficeJet gépeihez külön vezérlőt fejlesztett, ez a `hpoj` néven elérhető csomag, ha valakinek ilyen „mindent bele” gépe van, érdemes ezzel a csomaggal szöszölnie, hogy egyszerre tudjon faxolni, nyomtatni és a Für Elise-t hallgatni. Amennyire tapasztaltam, a `hpoj` megtalálja a többi nyomtatótípust, de nem szereti, ha nem ő vezérli az összes HP-eszközt, tehát ne használjunk lehetőleg HP nyomtatót a ***cupsys***-en keresztül és OfficeJetet az (egyébként szintén ***cupsys***-re támaszkodó) `hpoj`-n keresztül.

Mindenkit biztatni tudok arra, hogy próbálja ki hányféleképpen tudja nyomtatóját munkára bírni, remek kihívás ez a leendő rendszergazdák számára!

Sok sikert!

Szy György (Szy.Gyorgy@linuxvilag.hu)
A Linuxvilág főszerkesztője,
a Kiskapu Kiadó vezetője.
Mindenki levelét örömmel várja.



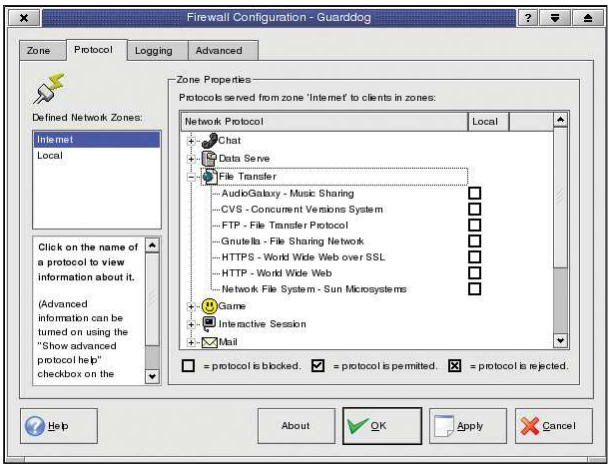
A magyar Linux: UHU (3. rész)

A Linux alapvetően hálózati operációs rendszer, s ez igaz az UHU-ra is. Habár azt mondják, a Linux elképzelhetetlen internethasználat nélkül, az igazság az, hogy ezt bármilyen rendszerről elmondhatjuk. A különbség mindössze annyi, a Linux ehhez is kiváló eszköz.

Az UHU-Linux már az alaprendszer telepítésekor rendkívül jól felszerelt hálózati programokkal. Gyakorlatilag vagy a telepítő-CD-n, vagy az interneten találunk olyan programokat, amik teljes mértékben képesek kiszolgálni a felhasználót. Mindezt üzembiztosan, gyorsan és elég biztonságosan teszi ahhoz, hogy ne kelljen homlokráncolva böngészni az interneten minden rezdülésre meredten figyelve, ami esetleg bármilyen támadásra utalhat. A linuxos felhasználó általában nyugodt ember....
Sorozatunk jelen részében ezekre a napi internethasználat során igénybe vehető alkalmazásokra összpontosítunk, néhány olyan program bemutatásával, amelyek alapértelmezetten települnek a rendszerrel.

Az „Őrkutya"

A beállításokat a GuardDoggal célszerű kezdeni. Amennyiben az ***UHU vezérlőpult*** és az ***UHU tárcsázó*** segítségével beállítottuk az internet-kapcsolatunkat, akkor mindenképpen ez a védelem a következő ajánlott lépése. Ebben segít bennünket a GuardDog, amely a Linux alapértelmezett tűzfalprogramjának grafikus felülete.
Az *1. képen* jól látható, hogy mind a külső, mind a belső hálózat beállításainak módosítására lehetőségünk nyílik. Ez igen hasznos, főleg akkor, ha az internetet megosztottuk, és nem szeretnénk, hogy a belső hálózati gép rendelkezésére álljon minden protokoll és kapu, mivel ez sebezhetővé teheti. A felület sokrétűsége viszont igen barátságos.
Gyakorlatilag otthoni felhasználásra vagy néhány gépes hálózatok védelmére tökéletesen alkalmas. Minden általánosságban használható protokoll letiltható, és ez igaz néhány jelenleg is divatos hálózati játékra (Quake, Diablo II, Half-Life, Direct play gaming). Ugyanakkor éppúgy kezelhető vele az SSH, HTTP, NFS, Gnutella, Audiogalaxy, FTP, CVS, IRC, ICQ, és a sort még hosszan lehetne folytatni.
A beállítások módosításával elérhető, hogy egy bizonyos kapu vagy protokoll a belső hálózaton igen, a külsőn viszont ne legyen használható, és fordítva. Természetesen ez minden hálózatra kötött gépen szintén beállítható, ilyen módon szigorú biztonsági rendszabályokat vezethetünk be. Ez azonban semmiképpen nem lesz képes kiszűrni korunk

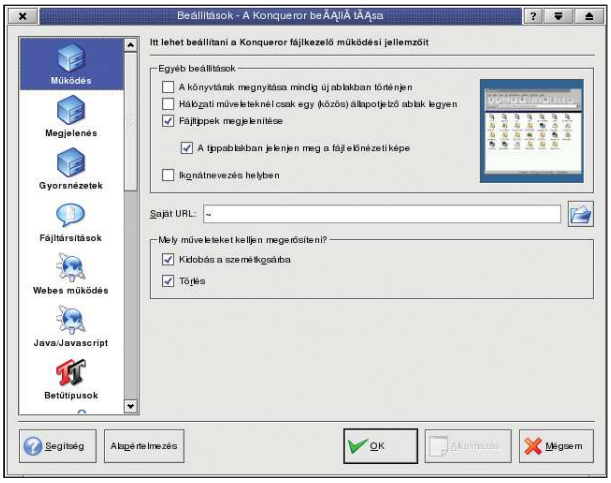


1. kép A GuarDog beállítópanele

férgekkel és vírusokkal fertőzött leveleit, ezek sajnos továbbra is eljuthatnak a felhasználókhoz. Amennyiben más operációs rendszert futtató gép is található hálózatukra kötve, célszerű egy vírusszűrőt is bevetni, mivel valószínűsíthető, hogy ez a más rendszerű gép lesz hálózatunk gyenge pontja. A GuardDog beállítása előtt érdemes több időt szánni az alapos tervezésre, hogy mely hálózati programokat és protokollokat fogjuk használni. Amennyiben valamit nem használunk, úgy érdemes fenntartani a tiltást (hiszen alapértelmezetten minden tiltott).

Böngészők: a bőség zavara

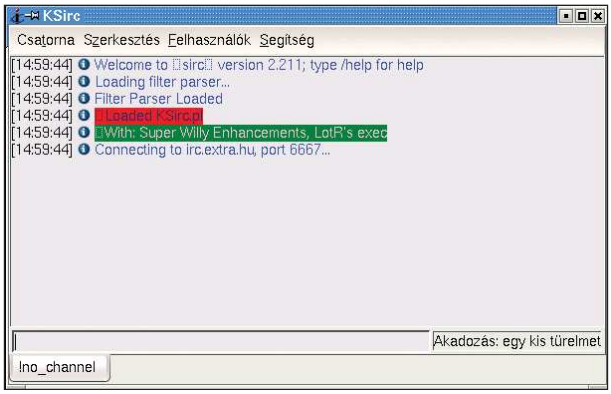
Ahogy azt már megszokhattuk, az UHU is több böngészővel rendelkezik. Az UHU alapértelmezett felületének a Gnome-nak éppúgy létezik alapböngészője (a Nautilus), mint a KDE-nek (a Konqueror). Mindkét böngésző igen jó, és remekül méretezhető. Letilthatók a sütik (cookies), a Java és a JavaScript, valamint a titkosítások is beállíthatók, továbbá teljesen testreszabhatók.
A két alapfelületen kívül is van azonban választék. A Mozilla egy remek nyílt forrású böngésző, amit gondolom, senkinek sem kell bemutatni, hiszen töretlen a fejlődése,



2. kép A Konqueror beállítása

3. kép Az Evolution

és felületfüggetlenségének köszönhetően sokan ismerik. Kevesebben tudják, hogy a Mozillának létezik egy beépített, jól használható levelezőügyfele is, illetve egy IRC-ügyfél szintén található benne, ami az internetes csevegést hivatott elősegíteni. Ezenkívül immár az *.uhu* csomagban is elérhető a méltán híres és népszerű Opera böngésző 7-es változata, ami külön figyelmet érdemel. Személyes kedvencem ez a böngésző, hiszen kicsi (Java nélkül megközelítőleg 3 MB helyet foglal), gyors és nagyon jól átgondolt, ráadásul kézre áll a menürendszere. Továbbá szintén tartalmaz levelező-ügyfelet, valamint egy kisméretű letöltésvezérlőt is beleépítettek. Igaz ugyan, hogy ez a letöltésvezérlő nem túl nagy, de kínál néhány hasznos szolgáltatást. Például a megszakadt letöltést képes folytatni, amivel rendkívül sok bosszúságot takaríthatunk meg, kiváltképp nagy fájlok esetén. Nem véletlen hát, hogy ezt a böngészőt nagyon sokan szeretik, és ezért nagyon elterjedt.
Aki kicsit nosztalgiázni szeretne (vagy csak a sávszélességet akarja kímélni, például GPRS esetén), annak az *.uhu* csomagban rendelkezésre áll a `lynx`, mely egy karakteres felületű böngésző. Méltán híres megbízhatóságáról, biztonságosságáról és gyorsaságáról.



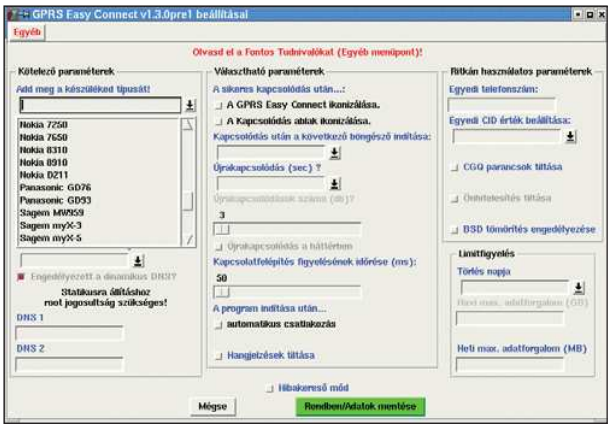
4. kép A KSirc csevegőprogram

Levelezés

Két fő levelezőügyfelet érdemes megemlíteni, a Kmailt és az Evolutiont. A Kmail egy egyszerű, ám igen jó ügyfél, az Evolutiont pedig a Linux Outlookjának is nevezik.
Az Evolution összkomfortos: határidőnaplót, személyi ütemezőt, naptárat, névjegykártya-jegyzéket tartalmaz, még az időjárási adatokat is ki tudja, mindemellett teljes értékű levelezőügyfél. Beállítása varázslóval történik, gyakorlatilag teljesen felhasználóbarát, és mindezt jó sebességgel valósítja meg. Mindössze egyetlen hibát találtam benne, ugyanis szerintem a ***Copy & Paste*** nem működik megfelelően, vagyis a másolás-beillesztés módszerrel körülményes átemelni egy-egy levél részleteit, például egy szövegszerkesztőbe. Habár a Kmail nem nyújt annyi szolgáltatást, mint az Evolution, és kizárólag levelezésre használható, viszont arra igen jó. Egyrészt lényegesen kevesebb erőforrással is beéri, mint az Evolution, és szinte mindent megtehetünk vele, aminek jó hasznát vesszük levelezéskor. A szűrők megfelelő beállításával a levélszemetet azonnal képesek leszünk törölni anélkül, hogy külön kellene bajlódnunk velük. Beállítható titkosítás és az üzenetszerkesztő is finomhangolható.

Pihenés

Sokan kikapcsolódásként szeretnének csevegni másokkal, és ilyen módon újabb barátokra szert tenni. Ebben többféle módon is segíthet a Linux. A webes csevegőprogramok használatához mindössze egy böngésző szükséges, ami remélhetőleg támogatja a Javát is. Sok webes csevegő ugyanis a Javára támaszkodik, habár PHP alapú programot is találhatunk közöttük. Ebben az esetben elegendő csak a böngésző is. Akadnak azonban kifejezetten csevegésre készített alkalmazások is, például ilyen az Internet Relay Chat (IRC), amely az internet hőskorában készült, de a mai napig töretlen a népszerűsége. Ennek oka valószínűleg az, hogy egy apró ügyfélprogrammal lehet csatlakozni és nincsen semmiféle más követelménye. A sávszélességet kíméli, hiszen teljesen szövegalapú, gyors és néhány alapvető paranccsal változatossá tehető a csevegés. A Linux alatt két nagyon népszerű program nyerte el a felhasználók többségének elismerését: az Xchat és a KSirc.
Mindkét alkalmazás teljes értékű IRC-program, és hogy melyiket használjuk, kizárólag „vallási” kérdés. Én személy szerint a Ksircet részesítem előnyben, de hozzáteszem,



5. kép GPRS-easy-connect, a hű segéd

kizárólag megszokásból. Arra figyeljünk, hogy az IRC is biztonsági kockázatot jelenthet, hiszen használata során saját kaput nyit a hálózaton (6667), és az IRC-hez sokan szeretnek parancsfájlokat írogatni, amelyek azután nyitott kapukat döngetnek (azaz igyekeznek kihasználni őket). Linux alatt sokkal kisebb a biztonsági kockázat, de létezik. Természetesen UHU alatt is elérhető több üzenetküldő szolgáltatás, például az ICQ vagy az AIM. Mindegyik rendszernek létezik linuxos ügyfele, és az ***internet*** könyvtárban is megtalálhatjuk őket. Érdemes kiemelni még a hírolvasó programokat is. Az UHU tartalmaz hozzá ügyfelet is, tehát semmi akadálya nincsen, hogy állandó hálózati kapcsolat esetén, hírkiszolgálókhoz csatlakozva folyamatosan nyomon kövessük a világ eseményeit. Ehhez remek kis programocska KnewsTicker, amelynél szinte csak és kizárólag a hírkiszolgáló címét kell megadni, és máris friss értesülésekhez juthatunk. A másik megoldás, hogy a KDE beépített kisalkalmazását (applet) használjuk, mindössze annyi a különbség, hogy ez esetben „futófény”-ként érkeznek a hírek a KDE tálcára, mivel ennek felületére épül be. Ugyanilyen hasznos, bár több szolgáltatással felruházott programocska a Knode is.

Hasznos holmik

A folyamatos hálózati, illetve különleges kapcsolat létrehozása során számos, pici program segítheti a felhasználót. Ezek az apró „utility”-k, igen kényelmessé tehetik a napi munkát, vagy látszólag bonyolult feladatokat segítenek könnyedén megoldani. Ilyen apró program például Korn. A Korn egy grafikus levélfigyelő, ami a tálcára települ, és előre megadott címről, illetve beállított időközönként lekérdezi, hogy van-e levelünk, ha igen, akkor mennyi. Segítségével követhetjük, hogy kaptunk-e valamilyen küldeményt, azaz nem szükséges feleslegesen megnyitni az ügyfelet, valamint szükségtelen nyitva hagyni memóriát és processzoridőt pazarolva. A másik figyelemre méltó alkalmazás, a GPRS-easy-connect. Ennek a magyar fejlesztésű programnak egyetlen feladata, hogy a GPRS-es telefonokat összekötve a számítógéppel, internetes kapcsolatot létesítsen. Ezt a munkát tökéletesen el is látja. Számtalan, jelenleg a kereskedelemben kapható készülékkel képes „szót érteni” és felépíteni vele a hálózatot.
Értékes program a kppp is, mellyel a modemesek járnak jól.

6. kép Csatlakozás KPPP-vel

Képes a modemes internetkapcsolat létrehozására, a modemet felismerve lekérdezi és beállítja. Gyakorlatilag a legjobban használható program a modemes internetelérés esetén, habár mindezt az UHU tárcsázója is tudja. Itt is megadható a felhasználónév, a jelszó, és az eszköz. Valóban nem tudja lekérdezni a modemet, viszont erre általában nincs is szükség. Amennyiben a Linux felismerte a modemet, gyakorlatilag azonnal használatba is vehető.
Végül ejtsünk szót egy letöltésvezérlőről, ami megkönnyítheti életünket. A Kget alapértelmezetten feltelepül, és ha KDE-felületet használunk, folyamatos segítéségét élvezhetjük. A Kget használhatóságáról ugyan megoszlanak a vélemények, de hasznossága kétségtelen. Első indulásakor megkérdezi, hogy szeretnénk-e, ha beépülne tálcára, és folyamatosan együtt működjön-e a Konquerorral. Amennyiben elfogadjuk az ajánlatát, folyamatosan felügyeli minden olyan letöltésünket, amelyet a Konquerorral végzünk. Tehát bármikor szeretnénk letölteni valamit ezzel a böngészővel, a Kget fogja felügyelni, és ezért csak a legritkább esetben fordulhat elő adatvesztés. Természetesen a Kget képes folytatni a megszakadt letöltést, sőt időben ütemezni is tudja.
Az UHU-Linux hálózati lehetőségeit most a teljesség igénye nélkül vettük sorra. Számtalan apró segédprogram található a rendszerben, amiről nem még szóltunk, hiszen olyan sok kiváló programról lenne szó, ami nemcsak a cikkünk, hanem a teljes lapunk terjedelmét meghaladná. Mindegyik feladatra legalább két, de sokszor ennél több választási lehetőségünk van, legyen szó akár web, helyi hálózat, FTP, IRC vagy bármely hálózati szolgáltatásról. Sőt, külön cikket is megérdemelne mindegyik, vagy olyan érdekesebb kérdéssel is lehetne foglalkozni, mint a hálózatos játékmód Linux alatt. Mindemellett az interneten is megszámlálhatatlan programot találhatunk, és számos egyéb kapcsolati lehetőséget is érinthetnénk még (Samba, videokonferencia). Cikksorozatunk következő részében az irodai alkalmazásokat vesszük sorra, hiszen a hálózati kapcsolat létrehozását, általában a dokumentumok szerkesztése követi. Természetesen a következő hónapban is UHU-Linux alatt tesszük mindezt.

Dancsok „strogg” Zoltán (strogg@mail.tvnet.hu)
Jelenleg technikai szerkesztőként dolgozik a BME-OMIKK-nál, ahol oktat is. Emellett egyetemi képzésben vesz részt, programozó matematikus szakon. Négy éve foglalkozik Linuxszal. Szabadidejében operációs rendszereket gyűjt és weblapot vezet.

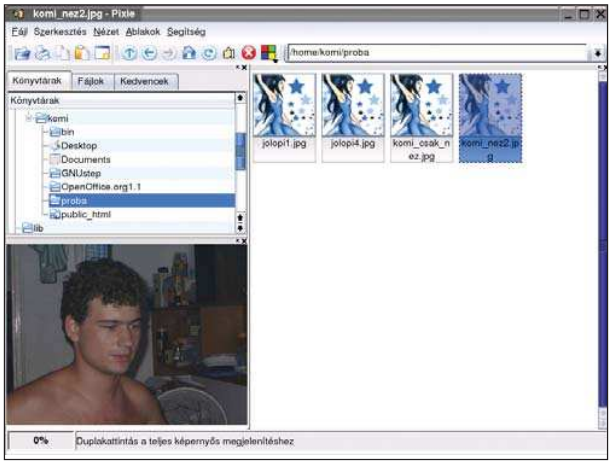
Hogyan térjünk át Linuxra lépésről lépésre

A képtelen Linux, mint képtelenség, avagy rendszerünk képmegjelenítő és képszerkesztő képességei.

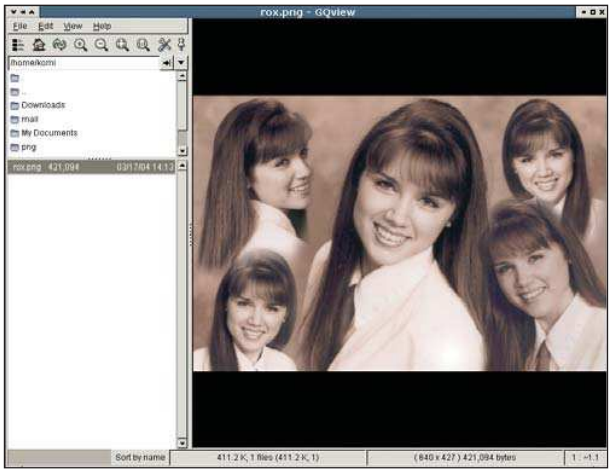
Számítógépes mindennapjaink része a digitálisan tárolt képek nézegetése, szerkesztése. A grafikus felhasználói felületek megjelenése szinte óta egyfolytában képeket bámulunk, amikor a számítógépünk elé ülünk: ha rákattintunk egy ikonra (amely egy a lemezen tárolt sok ezer kicsi kép közül), vagy csak unatkozva bámuljuk a monitort és észre sem vesszük, hogy a beállított háttérkép szintén egy kép. Nem csoda, ha a létező összes operációs rendszer igencsak jól el van látva képnézegető alkalmazásokkal. Mindig ki szoktam emelni, hogy a Linux milyen előkelő helyen áll az alkalmazások tekintetében, de ezt most nehéz lenne megtenni a rendkívül kiegyenlített mezőnyben. Most inkább más szemszögből szeretnék rávilágítani arra, hogy megint az járt a legjobban, aki Linuxot használ. A Linux ugyanis nem más, mint a rendszermag és a köré pakolt több ezer alkalmazás, segédprogram, rendszerprogram – beleértve az olyan profi eszközöket is, mint az OpenOffice.org irodai programcsomag vagy épp a Gimp képszerkesztő program. Ha tehát jobban belegondolunk, itt alapértelmezetten a rendszerrel kapjuk az összes olyan alkalmazást, amire csak szükségünk lehet. Ha csupán a képnézegetés, képszerkesztés területét nézzük: minden operációs rendszer tartalmaz beépített képszerkesztő, képnézegető eszközöket, ám ezek képességei a komolyabb használat során már nem elégségesek – ehhez más gyártók programjait kell telepíteni.

Ezek azonban ritkán ingyenesek, ha becsületesek vagyunk, és nem kalózkodunk, akkor pénzt kell adni értük. A Linux esetében csupán telepíteni kell néhány csomagot, s a gond megoldódott.

Szerencsénkre rengeteg program közül válogathatunk, bármit is szeretnénk csinálni. Ha szeretnénk meggyőződni erről, indítsuk el a YaST telepítőmodulját, s a ***csomagcsoportok*** szűrő alkalmazása után a kapott kategóriafában tekintsük meg a ***Grafika*** csomópont alatt található programkavalkádot. Írásomban ezek közül szeretnék szemelgetni és bemutatni a legérdekesebb, leghasznosabb darabokat; főképp azokat, amelyekre eddigi – immáron több éves – linuxos életem során támaszkodhattam, és amiket eredménnyel használtam. Ezenkívül néhány szót szeretnék ejteni a manapság oly elterjedt digitális fényképezőgépek Linux alatt történő használatáról.



1. kép A Pixie nevű képnézegető



2. kép A GQView képnézegető program

A legegyszerűbb képnézegetési mód

Gondolkodjunk egy kicsit, hol látjuk a mindennapok során a legtöbb képet? A válasz: a weben. Elég ellátogatnunk bármilyen hírportálra vagy épp kedvenc színészünk oldalára, mindenhol képek ezreivel találkozunk, s gondolom eddig még senki sem lepődött meg azon, hogy megjelennek, hiszen ez a dolguk. A webböngészők feladata pedig az, hogy megjelenítsék a képeket. Ennélfogva máris találtunk

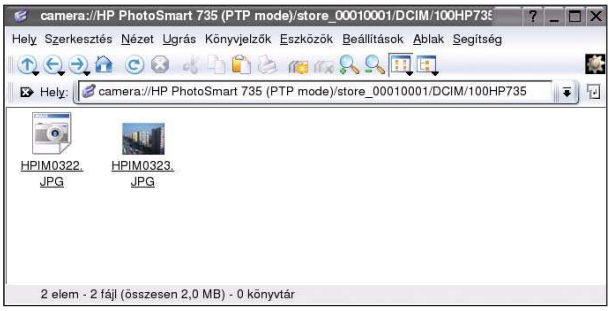
egy használható képnézegetőt, amellyel a helyi gépen tárolt képek is gond nélkül megtekinthetők. Az természetesen vicces volna, ha a digitális családi fotóalbumot webböngészővel akarnánk végignézni képenként, itt inkább csak arra szerettem volna felhívni a figyelmet, hogy egy eszközt nem feltétlenül szükséges mindig arra használnunk, amire rendeltetett. Ha egy program rendeltetéséből fakadóan képes valamilyen feladat ellátására, e képességet nyugodtan kihasználhatjuk más területeken is.

Pixie

Komolyabbra fordítva a szót, gondolom, mindenki egy kicsit okosabb módon szeretné megtekinteni a képgyűjteményét. Erre nyújt lehetőséget a KDE munkakörnyezethez fejlesztett Pixie nevű program (*1. kép*). A használatához a `pixieplus` nevű csomag telepítésére van szükségünk, amit a YaST csomagkezelő moduljával tehetünk meg. Ha ez megvolna, akkor adjuk ki a `pixie` parancsot, s lássunk csodát. A program első ránézésre megszólalásig hasonlít a Windows alatt megismert, méltán népszerű ACDSee nevű programra, s ez nem csak első ránézésre van így: működésük is megegyezik. Most keressünk egy olyan könyvtárat, amely tele van képekkel, és nyissuk meg. A bal alsó képernyőrészben megjelenik a kép kicsinyített változata. Teljes méretben úgy nézhetjük meg, ha ENTER billentyűt nyomunk ott, ahol éppen állunk. A teljes képernyős nézetben a bal felső sarokban megjelenik egy navigációs panel, amelyen a képek közötti váltogatásokon kívül a fényerő, illetve a kontraszt megváltoztatására is lehetőségünk nyílik, sőt, ha nagy kedvünk van hozzá, akkor el is forgathatjuk a képet. Fontos, hogy az itt végrehajtott változtatások nem maradandóak, csak a pillanatnyi megjelenítésre vonatkoznak. Az utolsó gombbal pedig kibekapcsolhatjuk a kép nagyítását, ugyanis alapesetben a kép kitölti a rendelkezésre álló helyet, s ez a kép kicsinyítő vagy nagyító irányú átméretezésével jár. Ha mi az eredeti méretben szeretnénk megnézni a képet, csak kattintsunk a nagyító ikonra. A képek között egyébként a kényelmes fel, le billentyűkkel is mozoghatunk. Az ENTER billentyű ismételt megnyomására visszaáll a képböngésző ablak.
A böngészőablakban érdekes szétnézni: jobb gombbal kattintva a képre előjön a kép gyorsmenüje, amelyből itt csak a ***Grafikus effektek*** menüpontot szeretném kiemelni, amelyben számtalan módosító lehetőség található. Az általam leggyakrabban használt eljárás ezek közül a képek elforgatása, ugyanis a digitális fényképezőgéppel készített állóképek a számítógépen fekvőként jelennek meg, vagyis mielőtt tároljuk, át kell forgatni őket. A program elvileg átalakítani (convert) is tud egyik formátumról a másikra, de ezt sajnos a gyakorlatban nem sikerült működésre bírnom.

GQview

A GQview egy másik hasonló program, amely szintén az ACDSee által teremtett képnézegető hagyományt folytatja. Telepítés gyanánt a `gqview` csomagot kell a YaST csomagkezelővel feltennünk (*2. kép*). Az alkalmazás a Gnome rendszerhez lett fejlesztve, de küllemében alig különbözik az előzőtől, bár itt egymás alatt van a könyvtárszerkezet és a benne lévő fájlok, s jobb oldalt található maga a kép. Ha teljes képernyős üzemmódban



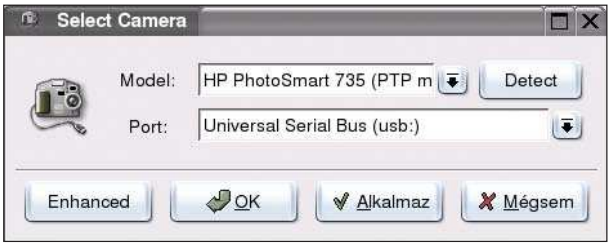
3. kép A Konqueror böngésző kamerakezelés közben

4. kép A Digikam fényképezőgép-kezelő

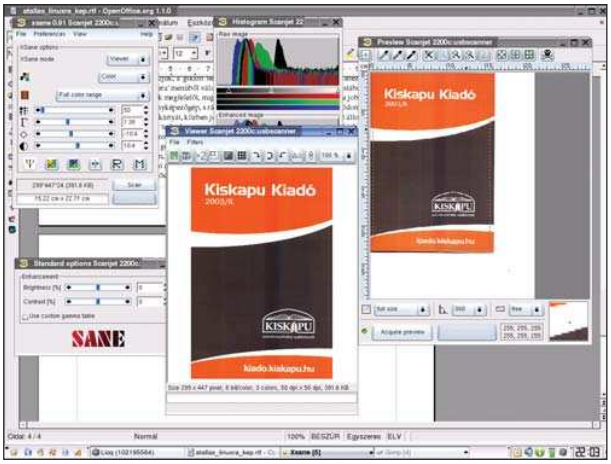
szeretnénk megtekinteni a képeket, akkor a megjelenő képre jobb gombbal kattintva a gyorsmenüben találunk erre vonatkozó menüpontot. Itt a teljes képernyős módban nem találunk sehol navigációs sávot, de bármikor kattinthatunk jobb gombbal a képen, hogy megjelenítsük a gyorsmenüt, amelyben már minden eddig megszokott lehetőséget használhatunk. Ennek az az előnye, hogy a navigációs sáv egyetlen képpontnyi helyet sem foglal el feleslegesen a képből. A képek között előre-hátra a PGUP, PGDOWN billentyűkkel mozoghatunk. A program egyik nagy előnye a Pixie-hez képest, hogy egy részletes beállítási panel segítségével kényelmesen testreszabható az alkalmazás. Az alapvető módosító hatások, mint például a kép elforgatása hasonló módon, a kép gyorsmenüjéből érhető el, sőt arra is lehetőségünk nyílik, hogy a képet egy külső képszerkesztő programmal szerkesszük. A képek átalakítása egyik formából a másikba is csak így lehetséges, segítségül kell hívnunk, például a Gimpet, erről a programról később ejtünk szót.
Természetesen lehetőségünk van indexképek megjelenítésére is, ehhez a fájllista gyorsmenüjében kell aktiválni a ***Nézet ikonokként*** menüpontot, ennek hatására egy könyvtárba lépve azonnal elkezdi előállítani egyenként az indexképeket. Ezenkívül a program diavetítő képességgel is bír, azaz a képeket valamilyen idő elteltével önmagától lépteti.

Kuickshow

Ez a képnézegető a KDE munkakörnyezet része, telepítésére nincs is szükségünk, alapértelmezetten benne van a rendszerünkben. A felépítése meglehetősen egyszerű, képességei messze elmaradnak az előző két vetélytársétól, ennek ellenére ez is egészen hatékonyan használható. Alaphelyzetben egy könyvtárlistát kapunk, amelyben mászkálva, ha képeket találunk, s rákattintunk, és egy új ablakban jelenik meg a kép. Ezután már itt is PGUP,



5. kép A GTKam fényképezőgép-kezelő program



6. kép Az Sane előlapja: az XSane lapolvasó program

PgDown billentyűkkel mozoghatunk a képek között, s itt is minden képnek van gyorsmenüje, ahol alapvető műveleteket – forgatást, kontrasztállítást stb. – végezhetünk.

Digitális fényképezőgépek kezelése Linux alatt

A SuSE kézikönyve szerint a digitális fényképezőgépek kezelése gyerekjáték. Én mindig is hitetlen ember hírében álltam, ezért elhatároztam, hogy megnézem, valóban ilyen egyszerű-e a kezelése. Ehhez kölcsönkértem lakótársam kameráját, egy HP PhotoSmart 735-öst, készítettem pár képet, aztán rákötöttem a gépem USB kapujára. A merevlemez némi kerregést hallatott, majd mindez abbamaradt. Ezek után elindítottam a Konqueror fájlböngészőt, és a hivatkozási sorba a `camera:/` elérési utat gépeltem (*3. kép*). Azonnal meg is jelent a fényképezőgép ikonja, amire rákattintva egy könyvtárlistát kaptam, köztük a tárolt képek könyvtárával. Ezt megnyitva, hagyományos állományokként láthatjuk a fényképezőgéppel készített képeket, akár egyenként meg is nézhetjük őket, ám a lassúcska USB-kapcsolat és a fényképezőgép akkumulátorainak igénybevétele miatt célszerű a képeket először merevlemezre menteni. Ezt a szokásos, windowsos másolás-beillesztéssel, vagy a fogd és húzd (drag and drop) módszerrel tehetjük meg. A dologban tehát semmilyen ördöngösség nincs, kameránk memóriakártyáját fájlrendszerként láthatjuk.

Digikam

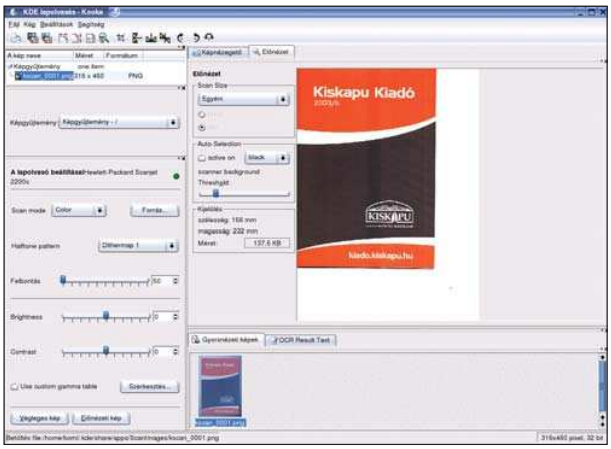
Ha egy kicsit komolyabban szeretnénk kezelni képeinket, s esetleg szeretnénk mondjuk indexképeket is látni, akkor próbáljuk ki a KDE munkakörnyezet részeként elérhető Digikam nevű programot (*4. kép*). Ez alapértelmezetten egy képnézegető, de lehetőség van fényképezőgépek kezelésére is. Ehhez a ***Beállítások*** menü ***Tulajdonságok*** menüpontjára kattintsunk, s a ***Kamerabeállítások*** fülön válasszuk az ***Automatikus felismerés*** gombot, amennyiben kameránkat már csatlakoztattuk. Ha nem ismerné fel, akkor adjuk hozzá kézzel, a megfelelő típus kiválasztásával. A főablakhoz visszatérve a ***Kamerák*** menüben találjuk a fényképezőgépünkre vonatkozó menüpontot. Rákattintva egy új ablakot kapunk, benne a csatlakoztatott gép memóriakártyáján található képekkel, amelyek indexképekként jelennek meg, s ugyanúgy megtekinthetjük őket az alkalmazás képnézegető moduljával, mint Konquerorban. Ezenkívül a képnézegető program számtalan lehetőségeket kínál, tudásában a Pixie-hez és a GQview-hez tudom hasonlítani – kényelmes.

gtKam

Ez az alkalmazás a Gnome munkakörnyezethez illeszkedik, s alapértelmezetten nincs telepítve. Ahhoz, hogy használni tudjuk, a `gtkam` nevű csomagot szükséges telepítenünk (*5. kép*), majd ugyanezzel a paranccsal indítva a ***Camera*** (Kamera) menüből válasszuk az ***Add camera*** (Kamera hozzáadása) menüpontot, s a listából keressük ki a fényképezőgépünknek megfelelőt, majd adjuk hozzá a gépek listájához. Ekkor a jobb oldali menüben megjelenik a fényképezőgép, s rákattintva a Konquerorban megszokott módon böngészhetjük a memóriakártyát, közben jobb oldalon megjelennek a gépen található állományok indexképei. Egy vagy több képet olyan módon tudunk menteni, hogy kijelöljük őket, majd jobb gombbal kattintva a ***Save*** (Mentés) menüpontot választjuk, megadjuk, hová szeretnénk menteni a képet, aztán várjuk, amíg a gépünkre átvándorol az anyag. Ez a program sokkal rugalmasabb megoldást nyújt a képek elhelyezését illetően, s mivel a csatlakoztatás módszerének körülményessége miatt célszerű a képeket gyorsan menteni a merevlemezre, gyakorlatilag mindegy, melyik megoldást választjuk, a mentést mindegyik program jól végzi.

Lapolvasók használata Linux alatt

Ha olvasóink a kezdetektől követik sorozatunk részeit, mostanra már biztosan be van állítva a lapolvasójuk, hiszen a bekapcsolt eszközt a YaST telepítés közben önműködően felismeri, s hozzáadja rendszerünkhöz. Ez azonban csak akkor van így, ha az eszköz a telepítés során rendelkezésre áll, azaz helyhiány miatt nem a szekrény tetején pihen, hanem csatlakoztatva van a géphez, áram alatt áll, és be is van kapcsolva. Aki nem így tett, s a lapolvasót utólag szeretné csatlakoztatni, különösebb meglepetés nem érheti, hiszen a jelenleg használatos USB-s eszközöket a SuSE a kábel csatlakoztatása után azonnal felismeri, s mint azt Windowsban már megszokhattuk, lehetőséget ad a beállítására. Ahhoz, hogy minderről személyesen is meggyőződhessek, kölcsönkértem az egyik ismerősöm HP ScanJet 2200c típusú lapolvasóját, majd a számítógépemhez kötöttem. Pár másodperc elteltével megjelent egy figyelmeztetés, miszerint új eszközt talált a SuSEPlugger, ez a modul felelős az új eszközök azonnali felismeréséért. Az ***eszköz beállítása*** lehetőséget választva elindult a YaST



7. kép A Kooka lapolvasó program

8. kép A sokoldalú Gimp képszerkesztő

lapolvasókat kezelő modulja, telepítette a lapolvasáshoz szükséges SANE nevű programot, ez felelős az eszköz és a felhasználói programok közötti kapcsolat megteremtéséért, majd feldobott egy listát több ezer lapolvasótípussal, előre kiválasztva az általam csatlakoztatott eszköz típusát. Nekem csak az ***OK*** gombra kellett kattintanom, s az eszköz azonnal rendelkezésemre is állt. Az eszköz beállítása egyébként utólag is módosítható a YaST ***Hardver*** moduljának ***Lapolvasó*** részében is.

Na igen, a lapolvasónk már működőképes, de ezzel egyelőre nem sokra megyünk, ugyanis szükségünk van egy olyan alkalmazásra is, amelyen keresztül a képeket beolvashatjuk az eszközről. Mint az egyéb területeken, itt is alapvetően kétféle megoldás használatos: az egyik a Gnome-felülethez, a másik a KDE munkakörnyezethez illeszkedik, lássuk melyik mit tud.

XSane

A nevéből könnyen kitalálhatjuk, hogy ez a SANE program egy előlapja (frontend), amely a GTK grafikus felületi elemkészletet (widget set) használja (*6. kép*). Alapértelmezetten nem része rendszerünknek, használatbavétel előtt telepítenünk kell az `xsane` nevű csomagot. Indítás után azonnal számtalan ablak ugrik elő a képernyőn, de ne ijedjünk meg a hirtelen jött áldástól: ami nekünk lényeges, az az Xsane főablaka, valamint az előnézeti (preview) ablak. Nézzük ez utóbbit, és a lapolvasóba helyezzünk be valami papírcetlit, majd az ***Acquire preview*** gombbal kérjünk egy új előnézeti képet. Itt jelöljük be az olvasni kívánt területet, majd a főablak ***Scan*** gombjára kattintva indíthatjuk el a nagyobb felbontású, jobb minőségű, tényleges beolvasást. Előtte természetesen lehetőségünk nyílik az olvasóprogram beállításainak módosítására, gondolok itt elsősorban a kontraszt és fényerő állítására a főablak közepén található csúszkasoron. Az egyik legfontosabb jellemzőt, a lapolvasó felbontását sajnos kissé eldugták, a ***jellemzők*** (preferences) menü ***beállítások*** (setup) pontja alatt tudjuk módosítani. Ennél a 600 dpi-s darabnál alapértelmezetten 300 dpi volt az eszköz felbontása. Minél nagyobb számot állítunk be, annál jobb minőségű képet kapunk és annál több lemezterületet kíván, de főképpen a beolvasásnál, sok-sok memóriát igényel.

Ha a ***Scan*** gomb megnyomásával elindítottuk a tényleges beolvasást, az eredményt hamarosan megnézhetjük egy új ablakban, egyébként itt lehet menteni a képet. A program tehát pontosan olyan, amilyennek Windows alatt megszokhattuk, de nem szabad elfeledkeznünk arról, hogy ahány gyártó, annyiféle beolvasóprogram létezik, így lehet, hogy csak számomra kísérteties a hasonlóság – mindenesetre mindent megtaláltam az alkalmazásban, amire szükségem volt.

Kooka

A Kooka a KDE grafikus környezet részét képezi, ám alapértelmezetten ez sem része rendszerünknek, s ha az ember nem figyel oda a telepítés során, s nem veszi észre a YaST lapolvasómoduljának súgóját, bizony nehezen talál rá az áhított alkalmazásra, ugyanis a program a `kdegraphics3-scan` csomag része, így ezt kell telepítenünk, ha használni szeretnénk a programot (*7. kép*).

Indítás után szinte ugyanazt a látványt kapjuk, leszámítva, hogy itt egyetlen szülőablak létezik, s ami eddig külön ablakban volt, az itt most különböző füleken, az ablak különböző részein található. Elrendezését tekintve sokkal összefogottabb, mint az XSane, s megvan az az előnye is, hogy magyar nyelvű. Az ablak jobb felének előnézet fülére kattintva láthatjuk a lapolvasó által készített előnézeti képet. Alapesetben ez üres, de mi ne lepődjünk meg, inkább kattintsunk a bal oldalt a lap alján található ***előnézeti kép*** gombra, majd a megjelenő képen jelöljük ki, hogy a lapolvasó ágyának mely részét szeretnénk beolvasni. Ezek után itt is lehetséges beállítani a kép fényerejét, kontrasztját, s itt végre kézre esik a felbontást szabályozó csúszka is. Éljünk ezekkel a lehetőségekkel (kezdetben a fényerőt és a kontrasztot nem érdemes macerálni, ilyen cselekedetre csak akkor vetemedjünk, ha a beolvasott kép túl sötét, túl világos vagy épp halvány), majd kattintsunk a ***végleges kép*** gombra, s várjunk türelemmel. A program azonnal a képgyűjteménybe menti a képet, majd a képernyő jobb alsó részén található ***Gyorsnézeti képek*** fülön biztosít számára egy ikonnyi helyet. Jobb gombbal kattintva a képet átmenthetjük oda, ahová valóban szeretnénk. A módszernek egyébként az az előnye, hogy egyszerre korlátlan számú átmeneti képünk lehet a programban, ezek nem vesznek el, bármikor menthetjük őket.

Egyéb lapolvasási lehetőség

Ha képet szeretnénk beolvasni, legtöbbször az a célunk, hogy tovább dolgozzunk vele valamilyen grafikai programmal, ha azonban tárolni szeretnénk, előtte nem árt néhány módosítást elvégezni rajta.
Ennek folyományaként leggyakrabban képszerkesztő programok használata közben lesz szükségünk a lapolvasónkra, ezért számos képszerkesztő, köztük személyes kedvencem, a Gimp pixelgrafikus képszerkesztő program is támogatja a külső források kezelését. Ennek segítségével egy adott programból közvetlenül megoldhatjuk, hogy a beolvasott kép azonnal a képszerkesztő megnyitott állományai közé jusson, elkerülve ezzel a felesleges mentést, megnyitogatást és a vele járó esetleges képromlást. (A Gimp például a *fájl* menü, *külső forrás* menüpontjával indítja el az XSane programot, melynek kimenetét azután felhasználja.)

Képszerkesztő programok

Ezen a területen igazából egyetlen program viszi el a pálmát, nem is akárhogyan, s ez a program a Gimp (GNU Image Manipulation Program). Emellett azonban számos kisebb szerkesztő és rajzolóprogram érhető el. Ilyen többek között az XPaint, amely a Windowsban megismert Paint nevű program linuxos megfelelője, ám a Painténél némiképp gazdagabbak a képességei. A program sok mindenre nem használható, ugyanúgy, ahogy a Paint sem, ezért komolyabb munkához szükségünk lesz valami épkézláb megoldásra.

A Gimp

Egy pixelgrafikus szerkesztőprogramról van szó, amely szinte fogalom a Linuxot ismerő emberek körében, hiszen ingyenesen még egy ilyen eszköz nem áll a felhasználók rendelkezésére. Minden túlzás nélkül elmondható, hogy felveszi a versenyt a nagyokkal: akár Corel PhotoPaintel vagy épp a méltán népszerű Adobe PhotoShoppal. Használatának módja ugyan sokban különbözik a versenytársakétól, de nem esélytelen megtanulni, sőt azt mondhatom, hogy inkább csak megszokás kérdése.
A program teljes használatának bemutatására egyébként egy újság is bőven kevés volna, s emellett ismét cikksorozatot jelentetünk meg a Gimp kezeléséről (*8. kép*). Az alapokkal – mint például egy kép átméretezése, a vászon méretének megváltoztatása, egy fájl más formátumban történő mentése – boldogulni fogunk mindenféle szakismeret nélkül is. Bajba inkább akkor kerülhetünk, ha valami többrétegű, átlátszósággal tűzdelt, töményebb képet szeretnénk készíteni, ehhez bizony külön el kell mélyednünk a program rejtelmeiben. Itt nincsenek szépséges eszközsorok (amiket egyébként a nagyok igen ritkán használnak), mindennek a kulcsa az adott képen elérhető gyorsmenüben (jobb gombbal történő kattintás után hívható elő) rejlik. A program egyébként messze túlmutat önmagán: rengeteg bővítményt kezel, s saját parancsfájlokkal akár a végtelenségig bővíthetők a képességei.
A használatbavétel előtt azonban álljon itt néhány jó tanács! Először is a SuSE 9.0-sban az alapértelmezetten telepített képszerkesztő a GKT 2.0-sra épülő Gimp 1.3, ezzel viszont számos nehézségbe ütköztem. Legfőbb baj, hogy rettenetesen megbízhatatlan, folyamatosan elszáll, ezért gyakran odavesznek az addig elkészített munkáim. Ezenkívül nagyobb baj, hogy néhány egyszerű művelet esetén is tudok olyan hibákat létrehozni, amelyeket ugyanazzal a műveletsorral lehet ismételni. Ezt valószínűleg a fejlesztők is érezhették, hiszen emellett a bizonyos Gimp2-nek nevezett csomag mellett az 1.2-es változatú Gimpet is benthagyták a rendszerben. Ez ugyan alapértelmezetten nincs telepítve, amennyiben használni szeretnénk, telepítsük a YaST segítségével a `gimp` nevű csomagot. Látni fogjuk, hogy ez még csak GTK 1.x-re épül, s nem is olyan szépek benne az ikonok, de mentségére legyen mondva, hogy nálam hosszú idő óta kifogástalanul működik. A kudarcok elkerülése végett mindenkinek azt javaslom, hogy az 1.2-es változatot használja, vagy keressen egy üzembiztos, újabb kiadást az 1.3-asból.
Egy kis érdekesség, hogy a GTK grafikus felületi elemkészlet (widget set), amelyre a teljes Gnome rendszer (s mellette még sok egyéb más is) épül, valójában a Gimp nevű programhoz készült jó pár évvel ezelőtt. Ám mint az bebizonyosodott, jól használható mindenféle ablakos rendszerben is. Aki nem hiszi, annak elárulhatom, hogy a GTK rövidítés nem más, mint a Gimp ToolKit, azaz Gimp eszközkészlet.
Aki komolyabban (például grafikai munkák elkészítésére) szeretné használni a Gimp képszerkesztőt, annak azt javaslom, hogy tanulmányozza a program hivatalos honlapján (➲http://www.gimp.org) található leírást és segédleteket.

Ami a folytatást illeti

Most, hogy a képek Linux alatt történő kezelését áttekintettük, nem ártana megismerkednünk a mozgóképek kezelésével, ám a téma bonyolultsága miatt kezdetnek az is elég lesz, ha csak a mozgóképek megtekintését tűzzük ki célul, a filmek szerkesztése amúgy sem elterjedt, ezt inkább csak nézni szokta mindenki.
Sorozatunk következő cikkében tehát megnézzük, milyen lehetőségeink vannak a digitális formában tárolt filmek megtekintésére kedvenc operációs rendszerünk használatával.
Szó lesz többek között a Xine nevű lejátszóról, a VideoLAN projekt keretein belül fejlesztett VLC-ről, és nem utolsó sorban a hatalmas népszerűségnek örvendő, magyar fejlesztésű MPlayer legújabb változatáról is.

Komáromi Zoltán
(komi@kiskapu.hu)
23 éves, a BME hallgatója, mellette PHP-programozóként dolgozik. Kedvenc területe a multimédia.

Linuxos kiszolgálót mindenkinek! (6. rész)

A SuSE Linux, mint kiszolgáló, kisvállalati és otthoni környezetben.

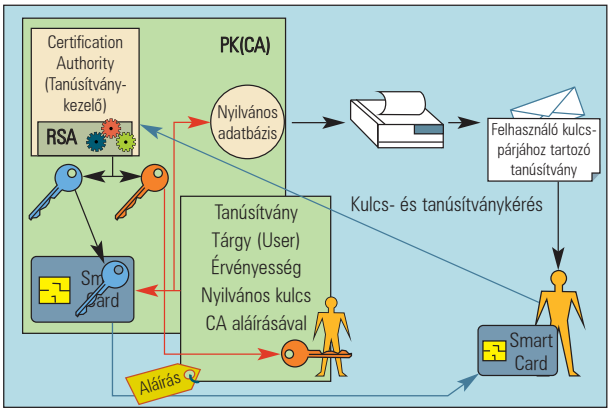
Cikksorozatunk előző részében egyszer már terítékre kerültek a tanúsítványok az SSL kiterjesztésű POP3 és IMAP protokollok kapcsán. Most kicsit általánosabban is foglalkozunk a tanúsítványokkal és a tanúsítványkezelő rendszerekkel. Saját tanúsítványkezelő rendszert telepítünk, elkészítjük az úgynevezett gyökértanúsítványt és saját tanúsítványokat is létrehozunk, amelyeket elhelyezünk a tanúsítványkezelő rendszerünkben. Ehhez szükségünk lesz az OpenSSL csomagra, amit a YaST-tal a már megszokott módon telepíthetünk. Az OpenSSL csomaghoz leírás is tartozik, amit felpakolhatunk. Ha mindezek végére értünk, tekintsük át a tanúsítványkezelő rendszerek felépítését és szerepét.

A tanúsítványkezelő-rendszerek szerepe

Az interneten tengernyi anyag található és rengeteg szolgáltatás vehető igénybe, ezt már mindannyian tapasztalhattuk. Az internet a szabad hozzáférésnek köszönhetően a jövő tudásbázisává és a kapcsolattartás csatornájává lehet. Ugyanakkor ez a szabad hozzáférhetőség kérdéseket is felvet, például a következőkön érdemes elgondolkodni. Mi szavatolhatja az interneten található adatok hitelességét, vagy az igénybe vett kapcsolati csatornák megbízhatóságát? A számítógépes hálózatok fejlődése lehetővé tette, többek között, az elektronikus kereskedelem és pénzforgalom kialakulását. Ezek a folyamatok azonban olyan adatbiztos környezetet igényelnek, amelyek könnyen kezelhetőek és törvényileg elfogadottak. A könnyű kezelhetőséget az elmúlt évek technikai fejlődése tette lehetővé, míg a törvényi szabályozás szerepét a Magyarországon 2001 szeptemberében életbe lépett 2001. évi XXXV. törvény hivatott betölteni.

A törvényi szabályozás létrejöttével elhárult minden akadály a PKI (Public Key Infrastucture, azaz a nyilvános kulcsú infrastruktúra) rendszerek hivatalos használata elől. A PKI rendszerek olyan környezetet teremtenek, amelyek lehetővé teszik a törvények által elfogadott kétkulcsú harmadik személyes hitelesítési és adatbiztosítási eljárások használatát számítástechnikai környezetben.

A PKI rendszerrel az elektronikus adatok kódolása és hitelesítése egységes módon történik. A rendszert használók mindegyike rendelkezik két kulccsal, egy nyilvános és egy titkos kulccsal. A nyilvános kulcs egy olyan kulcs, amit a felhasználó közzé tesz, ilyen módon az őt megszólító felek számára ismert. A titkos kulcsot a felhasználó a külvilágtól szigorúan elzárva tartja, amely egy állományban is lehet, amit CD-n vagy hajlékonylemezen őriz, de akár egy chipkártya is tartalmazhatja. (Például jelenleg ilyen chipkártyás megoldást használ hazánkban az APEH a kiemelt adózók által benyújtott bevallások digitális aláírásához.)



A két alapvető folyamat: a hitelesítés és a titkosítás (a módszer matematikai hátterét sorozatunk előző részében ismertettem röviden).

Titkosítás

A nyilvános kulcsú titkosítás nagy előnye több más titkosítással szemben, hogy a kódolás és a dekódolás nem ugyanazzal az eszközzel, kóddal történik, illetve az egyik kód a másikból jelenlegi matematikai tudásunk szerint nem állítható elő.

Így egy adott üzenet vagy adat kódolása úgy történik, hogy a küldő fél kódolja az adatot a címzett nyilvános kulcsával, így olyan adat jön létre, amely csak a címzett titkos kulcsával lesz visszafejthető. Mivel azonban a titkos kulcsot elvileg csak a címzett ismeri, ezért a titkosított adat más számára nem férhető hozzá.

Hitelesítés

Hitelesítés esetén arra van szükségünk, hogy megbizonyosodhassunk róla az adott üzenet küldője, vagy adott dokumentum szerzője valóban az a személy-e, akitől az adatot várjuk. Ennek módja, hogy az üzenet írója vagy a dokumentum szerzője az üzenetét a saját titkos kulcsával aláírja, majd a hitelesség ellenőrzéséhez a címzett oldalán ezt az aláírást kell dekódolni a küldő nyilvános kulcsával. Az alá-

írás gyakorlatilag az átküldött dokumentum leképzése egy olyan számmá – ellenőrző kóddá –, amely a dokumentum karaktereitől függ, illetve ez jellemzi az adott dokumentumot. Az adott dokumentumban bármilyen változtatás maga után vonja e számérték megváltozását, így a módszer arra is képes, hogy felfedezze azt, ha valaki illetéktelen módon az aláírás után módosította az üzenetet.
Amikor a címzett megkapja az üzenetet, a küldő nyilvános kulcsával dekódolja az aláírást, így meg tudja nézni, hogy valóban attól a feladótól kapta-e az üzenetet, akitől várta. Ha dekódolta az aláírást, a megérkezett üzenet ellenőrző kódját elkészítve megbizonyosodhat arról, hogy az átküldött dokumentumban történt-e változtatás a kézbesítés során.
Az eddig elmondottak logikusnak, fogalmazhatunk úgy is, egyszerűnek tűnnek. Felmerül azonban az a kérdés, honnan tudjuk, hogy a hozzánk eljutatott nyilvános kulcsok valóban ahhoz a személyhez tartoznak-e, aki hozzánk elküldte, illetve honnan tudjuk, hogy a kiosztásnál nekünk átadott titkos kulcs valóban csak a mi birtokunkban van, ahhoz más nem férhetett hozzá. E gond feloldására hozták létre a harmadik személyű hitelesítést, az úgynevezett tanúsítványkezelőket (CA – Certification Authority).
E tanúsítványkezelő rendszereknek feladata a kulcspárok kiosztása, a tanúsítványok kiállítása a kulcsok hitelességéről, valamint a megbízhatatlanná vált kulcsok visszavonása és nyilvántartásuk. A tanúsítványkezelőt akár nevezhetjük egy elektronikus közjegyzőnek is, akinek az a feladata, hogy a rajta keresztülmenő üzeneteket, esetünkben a kulcspárokat hitelesítse. Ebből természetesen következik az is, hogy egy tanúsítvány csak akkor tekinthető hitelesnek, ha megbízunk az adott hitelesítő szervezetben. E bizalom hiányában az általa kibocsátott tanúsítványok sem tekinthetőek megbízhatónak.

A rendszer felépítése

Kezdjük talán a legelején a tárgyalást, nézzük hol és miként lehet kulcspárokat előállítani. Alapvetően két kulcspár-előállítási mód létezik: az egyik a központosított, a másik a decentralizált kulcslétrehozás.
A központosított kulcs-előállításról akkor beszélünk, ha a kulcspárt a tanúsítványkezelő (CA) állítja elő és a titkos kulcsot kézbesíti a tulajdonosának, valamint a nyilvános kulcsot elhelyezi valamilyen hozzáférhető helyen, például valamilyen címtárban vagy névtárban.
Decentralizált kulcs-előállításról akkor beszélünk, ha a kulcspár nem a tanúsítványkezelő, hitelesítő szervezet oldalán készül, hanem ügyféloldalon. Ez történhet olyan módon, hogy a felhasználó saját maga előállít egy kulcspárt, azt hitelesítésre beadja valamelyik hitelesítő szervezethez, amely a bejegyzés (registration) után tanúsítványt bocsát ki a kulcspárhoz. A decentralizált kulcselőállítás másik módja, amikor egy chipkártyát használunk a kulcspár készítéséhez. Ilyenkor a chipkártya tartalmazza a titkos kulcsot és a chipkártya segítségével a nyilvános kulcs kerül előállításra és hitelesítésre a tanúsítványkezelő rendszer által. Ez utóbbi módszert nevezik fedélzeti (on-board) kulcslétrehozásnak. E módszer abból a szempontból célszerű, mert a titkos kulcs soha nem hagyja el a chipkártyát, így annak az esélye, hogy a titkos kulcs illetéktelen kezekbe kerül itt a legkisebb.
Természetesen a decentralizált kulcs-előállításnál a hitelesített kulcs beküldésére is vannak megbízható módszerek, ezek közül az egyik, ha a hitelesítésre szánt kulcsot a hitelesítő szervezet nyilvános kulcsával titkosítva küldjük el a hitelesítő szervezetnek, hiszen ez csak a hitelesítő szervezet titkos kulcsával lesz kibontható. A hitelesítés után a CA számunkra a titkos kulcsot a saját nyilvános kulcsunkkal aláírva küldi vissza, mivel ezt csak mi fogjuk tudni kinyitni a saját titkos kulcsunkkal. Ha még azt is szeretnénk, hogy más ne tudjon nekünk titkos kulcsot küldeni a hitelesítő szervezet nevében, akkor ez úgy oldható meg, ha az üzenetet a hitelesítő szervezet aláírja a saját titkos kulcsával, mivel így az aláírást kibontva a CA nyilvános kulcsával a küldő egyértelműen azonosításra került.
A fent leírtakból már látszik a rendszer működésének előnye, ez pedig az, hogy a két üzenetváltó félnek nem kell az üzenetváltást megelőzően kulcsot cserélni – például személyes találkozás, vagy megbízható telefonvonal útján – a biztonságos kapcsolattartáshoz, ez megoldható az első üzenetváltással.
Most nézzük a PKI rendszer vázlatos felépítését. A PKI rendszer áll egy tanúsítványkezelő rendszerből (CA), egy regisztrációs ügynökből (Registration Agent, RA) és kapcsolódik hozzá egy címtár, valamint egy ügyfél, aki a szolgáltatásokat igénybe veszi.

A CA

A tanúsítványkezelő rendszerek nemzedékeinek fejlődése jól nyomon követhető azon, hogy CA és az RA rész mennyire válik el egymástól. A kezdeti tanúsítványkezelők még nem különböztették meg a regisztrációs ügynököt, minden egy modulba volt összevonva. A fejlődés során az RA-k egyre több felügyeleti feladatot vettek át, míg a CA egyre inkább csak a tanúsítványok kiadásával foglalkozik.
A tanúsítványkezelő (CA) feladatai:

- RSA kulcspárok előállítása (központosított kulcselőállítás esetén).
- X.509 tanúsítványok kibocsátása.
- Nyilvános kulcsok személyhez kötése.
- Tanúsítvány-visszavonási listák karbantartása (Certification Revication List).
- Adatbázisok és névtárak karbantartása.
- Mester (Master) kulcsok kezelése.

A CRL (Certification Revication List – tanúsítvány-visszavonási lista) azon tanúsítványok gyűjteménye, amelyek valamilyen okból vissza lettek vonva. Ilyen ok lehet a titkos kulcs bizalmasságának sérülése. Itt jegyezném meg, hogy amikor tanúsítványkezelő rendszert készítünk, különös figyelmet kell fordítanunk az úgynevezett gyökértanúsítvány (Root Certification) védelmére, ugyanis ennek bizalmasságának sérülése az összes általunk hitelesített kulcs hitelességének elvesztésével jár. Ilyet komoly tanúsítványkezelő rendszer nem engedhet meg magának.
A CRL megfelelő időközökben való frissítése és közzé tétele az egyik legfontosabb kérdése a CA házirendjének. Szabályozni kell a CRL legnagyobb érvényességének idejét, a benne található tanúsítványok lejáratának idejét, valamint gondoskodni kell a CRL megfelelően sűrű időszakonkénti közzétételét egy mindenki által hozzáférhető helyen.

Az RA

A regisztrációs ügynök az újabb PKI rendszerekben gyakorlatilag teljesen átveszi a felügyeleti feladatokat a tanúsítványkezelőtől. Így az RA feladatai többek között az alábbiak:

- felhasználó-azonosító létrehozása;
- felhasználói kulcs kérése a CA-tól;
- felhasználói kulcs fogadása a CA-tól;
- felhasználói kulcs fájlban való tárolása;
- felhasználói kulcs kártyán való tárolása;
- tanúsítvány-visszavonás kérése a CA-tól;
- felhasználói házirend kezelése;
- felhasználói kulcs megújítás, illetve frissítés kérelmekre;
- kulcs-visszaállítás;
- felhasználó kulcs tárolására (smartkártyán vagy PKCS#12 kulcsfájlokban).

A PKI rendszer kezelése szempontjából rendkívül fontos a házirend megfelelő kialakítása, ami a kulcskezelésről és általában arról szól, hogy a rendszerben kinek, illetve melyik rendszerelemnek milyen szolgáltatási jogosultságai vannak.

Az ügyfelek

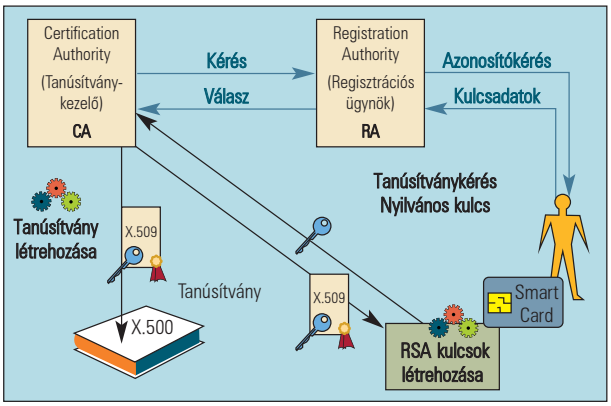
Az ügyfeleken szükségünk van olyan ügynökök telepítésére, amelyek képesek kezelni a tanúsítványokkal és chipkártyákkal kapcsolatos műveleteket.

A CA felépítéséről röviden

A tanúsítványkezelő rendszerben a különböző tanúsítványok egy faszerkezetben helyezkednek el. Ennek a faszerkezetnek a tetején az úgynevezett gyökértanúsítvány (Root Certification) áll. Ez egy különleges tanúsítvány a rendszerben, mivel ez az egyetlen olyan példány, ami önmagával van aláírva. Az összes többi kibocsátott tanúsítvány attól függően kerül elhelyezésre a fában, hogy melyik tanúsítvánnyal lett aláírva. Az aláírt tanúsítvány ugyanis az aláíró tanúsítvány egy gyerek csomópontjaként jelenik meg a fában. Ezzel a felépítéssel megoldható, hogy különböző különálló szervezeteket kezeljünk, illetve az általunk elismert szervezetek saját tanúsítványokat bocsássanak ki.

A gyakorlat

Az elmélet áttekintése után most nézzük miként is tudjuk létrehozni saját tanúsítványkezelő rendszerünket.
Miután telepítettük az OpenSSL csomagot, a */etc/ssl* könyvtárban, valamint a */usr/share/ssl* könyvtárban találjuk majd meg a szükséges, hivatkozott állományokat. Első lépésként a */usr/share/ssl/misc/CA.pl* állományról készítsünk egy hivatkozást a */usr/sbin* könyvtárban, mert erre a futtatható állományra a későbbiekben nagy szükségünk lesz. A *CA.pl* állomány egy olyan futtatható állomány, amely jelentősen leegyszerűsíti az OpenSSL használatát, így ennek a kiegészítésnek a használata célszerű lehet, ugyanakkor nem kötelező. Minden ismertetett parancs kivitelezhető a megfelelő `openssl` paranccsal is.
A */etc* könyvtáron belül találjuk az *openssl.conf* nevű állományt. Ebben az állományban tudjuk végrehajtani a legalapvetőbb beállításokat. Itt tudjuk megadni, hogy melyik könyvtárat használjuk alapértelmezett könyvtárként, azon



belül a különböző szerepű alkönyvtárak elhelyezkedése pontosan hogyan fest, hol találhatóak az alapvető tanúsítványok, továbbá a tanúsítványok érvényességére vonatkozó adatokat is itt tudjuk beállítani.
Érvényességi időnek ajánlatos olyan időtartamot megadni, ami nem túl rövid, de nem is lóg túl a tanúsítványok érvényességének előre becsülhető idején. Ha nincs tényleges feltétel, akkor használjunk 365 napos intervallumot az alábbi sor használatával:

```
default_days=365
```

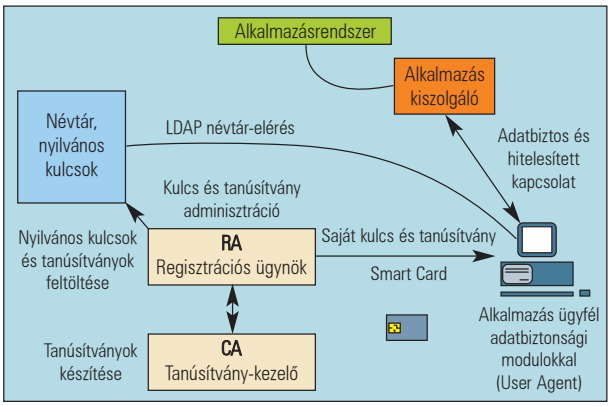
A `default_bits` változó értékének beállításával meg tudjuk adni, hogy a készített kulcsok hány bites titkosítást használjanak. Ajánlatos 1024 vagy 2048 bites titkosítást használni, mivel ez a jelenlegi számítási teljesítmény mellett gyakorlatilag feltörhetetlen.
Ha végeztünk az alapvető beállításokkal a `CA.pl -newca` paranccsal létrehozhatjuk az új tanúsítványkezelőnket.
A rendszer kérni fogja, hogy jelöljünk ki egy tanúsítványt, ami a jövőben a CA rendszer tanúsítványaként fog szolgálni. Ha nincs ilyen, akkor készíthetünk is egyet. Kövessük az utasításokat és készítsünk egyet, de sok időt nem érdemes ráfordítani, ugyanis mindjárt készítünk egy újat, mert ennek a tanúsítványnak 365 napnál hosszabb érvényességet szeretnénk adni.

Tanúsítvány készítése

Most hozzuk létre a CA első, egyedi tanúsítványát, a gyökértanúsítványt (Root Certification). Ez azért különleges tanúsítvány, mivel a rendszerben ez az egyetlen olyan tanúsítvány, ami önmagával van aláírva.
Adjuk ki a következő parancsot:

```
openssl req -config /etc/openssl.cnf -new -x509
➥-keyout private/cakey.pem \
-out cacert.pem -days 3650
```

Ezzel létrehozunk egy tíz évig érvényes tanúsítványt, amelynek a titkos kulcs részét elhelyezzük az *openssl.conf* állományban beállított gyökérkönyvtár *private* alkönyvtárban, míg a tanúsítvány részét a gyökérkönyvtárba tesszük.
A tanúsítvány készítésekor a Common Name (CN) értéknek adjuk meg, hogy ez melyik szervezet gyökértanúsít-



ványa lesz, például „AzÉnSzervezetem Root Certification". A tanúsítvány készítésekor meg kell adnunk egy jelszót, erre a jelszóra akkor lesz szükségünk, amikor a most készített gyökértanúsítvánnyal másik tanúsítványokat fogunk aláírni.
Ha végeztünk a gyökértanúsítvány készítésével, akkor ellenőrizzük, hogy az ***index.txt*** állomány üres legyen, valamint ellenőrizzük, hogy a ***serial*** állomány tartalma 01. Fontos, hogy a most készített tanúsítványt csak másik tanúsítványok aláírására használjuk, a hozzá tarozó titkos kulcsot és jelszót pedig olyan helyen őrizzük, ahol mások számára nem hozzáférhető.
Ha elkészültünk a gyökértanúsítvánnyal, akkor készítsünk belőle egy böngészők és egyéb alkalmazások által is feldolgozható tanúsítványt és telepítsük, mint megbízható gyökértanúsítványt. Innentől kezdve ugyanis az összes általunk aláírt tanúsítvány megbízható tanúsítványként lesz kezelve azokon a gépeken, ahová ezt a tanúsítványt telepítettük.

```
openssl x509 -in cacert.pem -out cacert.crt
```

Ha végeztünk a gyökértanúsítvány létrehozásával, akkor itt az idő, hogy elkészítsük az első olyan tanúsítványunkat, amit az imént készített gyökértanúsítvánnyal írunk alá. Ehhez először el kell készíteni egy kérelmet egy új tanúsítványhoz az alábbiak szerint:

```
CA.pl -newreq, vagy openssl req -config
➥/etc/openssl.cnf -new -keyout newreq.pem -out
newreq.pem -days 365
```

Ezzel a paranccsal létrehoztunk egy új kérelmet, amelyet a ***newreq.pem*** állományban helyeztünk el. A tanúsítvány létrehozásakor figyeljünk oda arra, hogy CN (Common Name) értéknek mit adunk meg. Amennyiben például a ***www.tartomany.hu*** weblap SSL tanúsítványát készítjük el, akkor a CN-nek a ***www.tartomany.hu*** értéket adjuk, ha azonban a ***cimzett@tartomany.hu*** elektronikus levélcímhez tartozó felhasználó számára készítünk digitális aláíráshoz, valamint titkosításhoz tanúsítványt, akkor a CN-nek adjuk a ***cimzett@tartomany.hu*** értéket.
Amennyiben az új kérelemmel elkészültünk, akkor nem marad más hátra, mint a kérelem aláírása a gyökértanúsítvánnyal (Root Certification):

```
CA.pl -sign, vagy openssl ca -config
➥/etc/openssl.cnf -policy policy_anything -out
newcert.pem -infiles newreq.pem
```

Az aláíráshoz szükségünk lesz a gyökértanúsítvány jelszavára, majd ha egyezik a jelszó, akkor a ***newcerts*** könyvtárban létrejön az ***XX.pem*** – ahol XX a ***serial*** állomány tartalma – tanúsítvány és ez megjelenik az ***index.txt*** állományban is.
Az elkészült tanúsítványból a már megismert módon készíthetünk ügyfélprogramok által feldolgozható tanúsítványt.

Tanúsítvány visszavonása

Tanúsítványt az `openssl -revoke newcert.pem` paranccsal vonhatunk vissza. A tanúsítvány visszavonása után ne feledjük, hogy frissíteni kell a visszavont tanúsítványokat tartalmazó adatbázist is. Ezt megtehetjük az `openssl ca -gencrl -config /etc/openssl.cnf -out crl/sopac-ca.crl` paranccsal. Ha frissítettük a listát, akkor gondoskodjuk a lista közzétételéről is, például a tanúsítványkezelő rendszerünkhöz tatozó weblapon.

Tanúsítvány megújítása

Ha egy tanúsítvány érvényessége lejár, akkor elképzelhető, hogy a tanúsítvány felhasználója meg szeretné újítani azt. Az ilyen kérelmek beérkezése esetén az eddig érvényben lévő tanúsítvány visszavonásáról is gondoskodnunk kell, mielőtt kiadjuk az új tanúsítványt. A régi tanúsítványt az ***index.txt*** állományban kikereshetjük, így nem fog nehézséget okozni a sorszám kiderítése, ami alapján a visszavonást el kell végezni.
Röviden talán ennyiben foglalható össze a tanúsítványkezelő rendszerek elméleti és gyakorlati működése. A gyakorlás kedvéért készítsünk tanúsítványokat, terjesszük ismerőseink, üzletfeleink körében, használjuk ki a digitális aláírás és a titkosítás nyújtotta biztonságot. Amennyiben olyan tanúsítványra volna szükségünk, amelyet valamelyik nagy tanúsítványkezelő hitelesít, úgy az interneten erre is találunk megoldást. A következőkben csak néhány példát említenék ezek közül.

Elérhető nyilvános CA szervezetek

Végül ejtsünk szót néhány nyilvános CA szervezetről, melynél mi is hitelesíttetni tudjuk az általunk használt kulcspárokat. Az EuroPKI egy olyan közös európai szervezet, amelynek feladata, hogy nyilvános kulcsú hitelesítést adjon magánszemélyek, szervezetek és különböző európai projektek számára. A szolgáltatás a ➲ http://www.europki.org címen érhető el.
Természetesen létezik magyar tanúsítványkezelő szolgáltatás is, példaként említeném a ➲ http://www.netlock.net oldal szolgáltatását.

Illés Viktor (viktor@ei.hu)
23 éves, a BME műszaki informatikus szakának hallgatója, mellette weblapokkal, linuxos és windowsos rendszerekkel foglalkozik. Szabadidejét legszívesebben a szabadban tölti, teniszezik és kerékpározik.

Írjunk saját grepet!

A GNU libc Posix-megfelelő, szabályos kifejezések kezelésére alkalmas függvényeinek bemutatása.

Előfordult már veled, hogy egy felhasználó által megadott kifejezésre történő keresést akartál megvalósítani C-programodban? Ha igen, és eddig még nem találtad meg a feladat szép megoldását, ez a cikk neked szól. A méltán népszerű `grep` nevű segédprogram által szintén alkalmazott szabályos mintaillesztés az egyik leghatékonyabb eszköz a haladóbb felhasználók kezében. Szabályos kifejezésekkel (regular expressions) pillanatok alatt hámozhatunk ki átláthatatlan méretű szöveges adathalmazból értékes adatokat.

Minden Linux-terjesztés része a GNU `libc6`, amely a Posix-szabványhoz erősen igazodik. A GNU `libc` a ***regex.h*** fejléccállománnyal szolgál a szabályos kifejezések adta korlátlan lehetőségek kiaknázásához. Az első és legfontosabb általunk is használt elem a `regex_t` típus. Ez egy szerkezettípus, mely a szabályos kifejezésünket fogja tárolni, illetve számos, belső használatra fenntartott tárolót fog össze. Közvetlenül tilos belenyúlni, csupán a megfelelő függvényhívásokon keresztül szabad hozzáférni a tartalmához. Ennek megfelelően, miután létrehoztunk egy `regex_t` típusú változót, a szabályos kifejezést bele kell töltenünk. Ezt valósítja meg a `regcomp()` függvény. Három jellemzője közül az első egy mutató egy `regex_t` típusra. Ez nyilvánvalóan a fent említett változónk címe lesz. A második maga a szabályos kifejezés egy karakterlánc formájában. Az utolsó pedig egy egész szám, mellyel a mintaillesztés módját befolyásolhatjuk.

A ***regex.h*** számos makróval áll rendelkezésünkre. Ezekkel bitenkénti vagy művelettel összekapcsoltan több beállítást hajthatunk végre egyszerre. Két igen fontosat emelnék ki a sorból: az egyik a `REG_EXTENDED`, amely továbbfejlesztett mintaillesztést ír elő. Ez hasonló működést jelent, mint amit az `egrep`-pel tapasztalhatunk. A másik a `REG_ICASE`, amellyel a kis- és nagybetűk közötti különbséget hagyjuk figyelmen kívül.

A feltöltött szerkezetet dolgunk végeztével fel kell szabadítanunk. Az ezt megvalósító `regfree()` egyetlen értéket vár. Egy mutatót kér a `regex_t` típusra, tehát itt is a szabályos kifejezésünket tároló szerkezet címét fogjuk átadni.

A tényleges munkát végző függvény neve `regexec()`. Ijesztően hangzó öt kapcsolója közül az utolsó három többnyire alapértelmezés szerinti értéket kaphat. Először szokás szerint szerkezetünk címét adjuk át. Másodikként azt a karakterláncot, amire el akarjuk végeztetni a mintaillesztést. A harmadik és negyedik kapcsoló a részekre bontáshoz szükséges. Amennyiben nem kívánunk élni ezzel a lehetőséggel, a `0`, illetve `NULL` értékek lehetnek. Utolsóként további makrók segítségével egyéb kapcsolókat állíthatunk be az illesztéshez.

Sokszor háttérbe szoruló, ám annál fontosabb a család negyedik és egyben utolsó tagja, a `regerror()`. Ez a `regcomp()` vagy a `regexec()` visszatérési értékéhez szolgáltat hibaüzenetet. Első értéke a hibakód. A második annak a szerkezetnek a mutatója, amellyel a hiba előfordult. A harmadik a hibaüzenet tárolására szánt átmeneti tár, míg a negyedik ennek a mérete.

Hogyan lesz mindebből program? Példaképpen készítsük el saját `egrep`-ünket! A feladatot annyiban egyszerűsítjük, hogy nem értelmezünk kapcsolókat, illetve az adatot kizárólag a szabványos bemenetről fogadjuk. A továbbfejlesztett mintaillesztés szabályait fogjuk követni, ahogy az `egrep` is teszi. Egyetlen korlátozással fogunk élni, miszerint feltesszük, hogy egy sor nem hosszabb 255 karakternél.

```
#include <stdio.h>
#include <regex.h>
#include <sys/types.h>

#define MAXHOSSZ 256

int main (int argc, char **argv)
{
    char sor[MAXHOSSZ];
    int hiba;
    regex_t regkif;

    if (2 > argc) {
       fprintf (stderr, "Hasznalat: %s
                ➥{regkif}\n", argv[0]);
       return 1;
    }

    hiba = regcomp (&regkif, argv[1],
           ➥REG_EXTENDED);

    if (hiba) {
```

```
      regerror (hiba, &regkif, sor, MAXHOSSZ);
      fprintf (stderr, "Hiba: %s\n", sor);
      return 2;
    }

    while (NULL != fgets (sor, MAXHOSSZ, stdin)) {
      if (0 == regexec (&regkif, sor, 0, NULL, 0))
          printf ("%s", sor);
    }

    regfree (&regkif);

    return 0;
}
```

Sorról-sorra tekintsük át a program működését! A # (kettős kereszttel) kezdődő sorokat az előfeldolgozó fogja értelmezni. Három fejlécállományt használunk, az ***stdio.h***-t a be- és kimenetet kezelő függvényekhez, a ***regex.h***-t a szabályos mintaillesztés függvényeihez, illetve a ***sys/types.h***-t. Ezen túlmenően meghatározunk egy makrót `MAXHOSSZ` néven, 256 értékkel. Ez adja meg annak az átmeneti tárnak a méretét, melyben a sorokat tárolni fogjuk.
A `main()` fejlécében az `argc` változó az `argv` elemszámát tartalmazza. Az `argv` nulladik eleme az a név, amellyel meghívták a programot. További elemei a program esetleges kapcsolói. A `sor` változónk a fő ciklusunk minden lefutása során egy sort fog tartalmazni a bejövő adatokból. Fontos, hogy legfeljebb `MAXHOSSZ` – egy karakter tárolására alkalmas, mivel a karakterláncot lezáró 0-t is tárolni kell.
A hiba a `regcomp()` visszatérési értékét fogja megkapni. Végül pedig a `regkif`, a szabályos kifejezésünk szerkezete. Az első `if` feltételben az átadott értékek számát ellenőrizzük. Az `argc` mindig legalább 1, mivel az `argv` nulladik eleme, a program neve mindig adott. Ezért az `argc` 2 lesz, ha a program egy értéket kapott, így a feltételünk akkor igaz, ha értékek nélkül indítottuk a programot. Ebben az esetben a szabványos hibacsatornára írjuk ki a program használatának helyes módját. Ezt követően 1 visszatérési értékkel véget ér a program.
Ha kaptunk értéket, a `regcomp()`-ot futtatjuk vele. A `regkif` címét az & (és) karakterrel képezzük, ez a függvény első értéke. Másodikként a programnak átadott szabályos kifejezést kapja. Harmadikként megadjuk a `REG_EXTENDED` makrót, mellyel továbbfejlesztett mintaillesztésre utasítjuk a függvényt. A `regcomp()` visszatérési értékét a `hiba` változóban tároljuk. A `hiba` nullától különböző értéket fog kapni, ha a szerkezetbetöltés sikertelen volt.
Amennyiben hiba történt, meghívjuk a `regerror()` függvényt. Első értékként a hibakódot kapja, majd a szerkezet címét. A harmadik érték az az átmeneti tár, amelyben a hibaüzenetet kívánjuk tárolni. E helyen felhasználjuk `sor` változónkat. Ennek hosszát ismerjük, a `MAXHOSSZ` makróval állítottuk be, ez lesz a negyedik érték. Miután a `sor` felvette a hibaüzenetet, a szabványos hibacsatornára írjuk ki az üzenetet, majd 2 visszatérési értékkel kiszállunk.
Ha minden rendben ment, jöhet a fő ciklus. Mindenekelőtt el kell döntenünk, hogy milyen függvényt használunk a szabványos bemenet soronkénti olvasására. Első kóbor ötletünk a `gets()` lehetne, de használata nem ajánlott. Semmiféle ellenőrzést nem tartalmaz az átmeneti tár túlírásra, így könnyen elszállhat a programunk, amellett, hogy nem is biztonságos. Első ránézésre kényelmetlennek tűnhet az `fgets()` alkalmazása, mert egy felső korlátot kell megadni a beolvasandó adatok méretére, de pont ez adja az előnyét is. Az `fgets()`-szel a `sor` tömbbe egy új sort olvasunk be a sortörés jellel együtt, a végén a lezáró nullával. A visszatérési érték a sor címe vagy `NULL`, ha a bejövő adat elfogyott. Ezzel kiváló ciklusfeltételt adtunk meg. A `regexec()`-et csak alapszinten használjuk, a szerkezet címét és a soron következő sort adjuk meg neki. Ha a visszatérési érték 0, akkor illeszkedik. Ebben az esetben kiíratjuk `sor` változót. Itt is megfigyelhetjük, hogy az `fgets()` a sortöréssel együtt olvassa be az adatot.
A program végén felszabadítjuk a szerkezetnek azt az elemeit, amelyeket a `regcomp()` dinamikusan foglalt, majd 0 visszatérési értékkel kilépünk.
Lássuk hogyan használható frissen írt programunk! A C-forrást előbb le kell fordítanunk. Mindig érdemes kitenni a `-Wall` (warning all) kapcsolót, mellyel arra kérjük a fordítót, hogy az összes figyelmeztetést jelenítse meg.
Alapértelmezésben ugyanis kizárólag a hibákat (errors) írja ki a képernyőre. Ennek megfelelően:

```
$ gcc -Wall -o regkif regkif.c
```

Így létrejött egy ***regkif*** nevű állomány, amit a fordító okosan már futtathatóvá is tett. Először nézzük meg, működnek-e a kivételkezelő részek. Először indítsuk el kapcsolók nélkül a programot:

```
$ ./regkif
Hasznalat: ./regkif {regkif}
```

Lássuk, hogyan működik a `regerror()`! Figyeljük a választ a különböző hibás szabályos kifejezések megadása esetén:

```
$ ./regkif "*"
Hiba: Invalid preceding regular expression
$ ./regkif "\1"
Hiba: Invalid back reference
$ ./regkif "[[:alma:]]"
Hiba: Invalid character class name
```

Most már jöhet egy éles próba. A */etc/services* fájlt fogjuk használni. Ez egy egyszerű szöveges fájl, ami hálózati szolgáltatásokhoz rendel kapuszámokat. Innen tudhatjuk meg, hogy az FTP szokásos kapuja a 21, a HTTP-é a 80. Remekül lehet benne keresni. Tegyük fel, hogy mi arra vagyunk kíváncsiak, hogy az 53-as mi szokott lenni.

```
$ cat /etc/services | ./regkif
"^[[:alnum:]]+[[:blank:]]+53/tcp"
domain  53/tcp  nameserver   #name-domain server
```

Számos lehetőséget nem ismertettem a helyszűke okán, ezért mindenképpen érdemes elolvasni a regex(3) útmutatásait.
Ami a legfontosabb: soha ne találd fel újra a kereket!

Fülöp Balázs

Számítógépes hálózatok (5. rész)

Sorozatunk előző részében kábeleken keresztül küldözgettünk adatokat. Most megnézzük, miként lehet ugyanezt tenni, kábel nélkül. Végül egy kis ízelítőt adunk a távközlési rendszerek világából.

Előző cikkünkben nekikezdtünk első nagy témakörünknek, az adatátviteli módszerek boncolgatásának. Eddig csak vezetéken történő adatküldéssel foglalkoztunk. Az ott felsorolt lehetőségek (például csavart érpár, koaxiális és optikai kábelek) csak akkor jöhetnek szóba, ha „helyhez kötött” kapcsolattartást végzünk. Ha valaki a villamosról szeretne adatokat továbbítani ismerősének, annak mindenképp valamilyen vezeték nélküli adatátviteli megoldáson kell gondolkoznia.
Nem csak ilyen esetekben jöhet jól a vezeték nélküli mód. A vezetékes adatátvitelnek akad egy jókora hátránya: előzetes kiépítést igényel. Magyarán szólva valakinek le kell fektetnie a kábeleket. Ha esetleg az összekötni kívánt két pont között egy folyó vagy netán egy őserdő is útba esik, akkor rendkívül drága és körülményes lehet a kiépítés. Ilyen esetekben is sokkal egyszerűbb (és nem utolsó sorban, olcsóbb) a vezeték nélküli adatátvitelt választani.
A továbbiakban körüljárjuk, milyen megoldások léteznek arra, hogy vezeték nélkül, elektromágneses hullámok segítségével adatot küldhessünk egy nem túl messze lévő szomszédunknak.

Az elektromágneses hullám

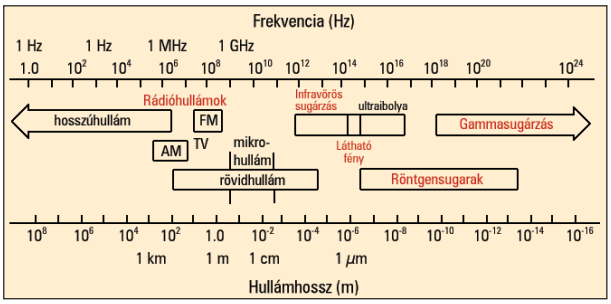
Az elektromágneses hullámokat az elektronok mozgása kelti. Fontos tulajdonságuk, hogy szabad térben és vákuumban egyaránt tovaterjednek, az utóbbiban a fénnyel megegyező sebességgel. (Az elektromágneses hullámok terjedéséhez – ellentétben a mechanikai hullámokéval – nincs szükség közegre).
Az elektromágneses hullámok létezését először ***James Clerk Maxwell*** angol fizikus jósolta meg, de – kísérleti úton – először csak ***Heinrich Hertz***-nek sikerült megfigyelnie. Ezért nevezték el róla a frekvencia (az elektromágneses hullámok másodpercenkénti rezgésszámának) mértékegységét, a Hertz-et (Hz). Az elektromágneses hullámoknak a frekvencián kívül van még egy fontos jellemzőjük, a hullámhossz, amely a két egymást követő hullámcsúcs (illetve hullámvölgy) közötti távolságot adja meg. A frekvenciát általában f-fel, a hullámhoszt pedig a görög λ (lambda) betűvel szokás jelölni.
A vákuumbeli fénysebesség (c), a frekvencia és a hullámhossz között az alábbi összefüggés figyelhető meg:

$$\lambda * f = c$$

Mivel a c állandó, ezért az f ismeretében könnyen kiszámolhatjuk a hullámhosszt, és fordítva. Fontos megjegyeznünk, hogy az elektromágneses hullám vákuumban lévő terjedési sebessége független a frekvenciától. Ha azonban valamilyen közegben (például üvegben, rézben stb.) terjed, akkor egyrészt kismértékben frekvenciafüggővé válik, másrészt csökken a terjedési sebessége, körülbelül a vákuumbeli fénysebesség kétharmadára.
Mivel elektromágneses hullámok gyorsuló töltések hatására keletkeznek, ezért könnyedén elő tudjuk állítani őket, ha egy vezetőben folyó áram erősségét folyamatosan változtatjuk. Az antennák is ezen az elven működnek. Ha az antenna megfelelő méretű, akkor a nem túl messze lévő vevőállomás ezeket a hullámokat észlelni tudja.
Az elektromos hullámok általában a tér minden irányában egyenletesen terjednek. Ez nagyon előnyös tulajdonsága, amennyiben műsort szeretnénk sugározni. Ha viszont két pont közötti kapcsolatot szeretnénk megvalósítani, akkor ez nem túl hatékony módszer, mivel a sugárzásra fordított energia igen nagy százaléka veszendőbe megy.
Nyilván létezik mód arra is, hogy az energiát csak egy irányba, pontosan a vevő felé sugározzuk. Erre találták ki az úgynevezett reflektorokat (pontosabban a parabolid alakú reflektorokat), amelyek az energiát mindig a vevő felé sugározzák. Gondoljunk csak a zseblámpa és a villanykörte esetére (a fény is elektromos hullám). Amikor otthon felkapcsoljuk a villanyt, a fény az egész szobát betölti. A zseblámpa azonban a fényt mindig egy irányba összpontosítja.

Az elektromágneses spektrum

Az elektromágneses spektrum nem más, mint a teljes mérhető frekvenciatartomány. Ez a nagyon alacsony rádiófrekvenciától a látható fényen keresztül a nagy energiájú röntgen-, illetve az azon is túl lévő gamma-sugárzásig tart (*lásd ábránkon*).
Adattovábbításra leginkább a rádióhullám, a mikrohullám, az infravörös hullám és a látható fény része alkalmas. Valójában sokkal jobb lenne, ha a nagyobb frekvenciájú röntgen és gamma-sugarakat használhatnánk, de egyrészt ezeket nagyon nehezen tudjuk előállítani és modulálni,



másrészt nem terjednek jól az épületekben. Nem mellékes az sem, hogy az egészségre sincsennek túl jó hatással.

Az, hogy egységnyi idő alatt mennyi adatot tudunk továbbítani, a sávszélességtől függ. A sávszélességet a hullámhossz- és a frekvenciatartomány szélessége adja meg. Általánosan azt mondhatjuk, hogy minél szélesebb a frekvenciatartomány, annál nagyobb az adatátviteli sebesség. Ez azt jelenti, hogy ha nagy adatsebességet szeretnénk elérni, akkor nagy frekvenciatartományra van szükségünk. Az azonban beláthatatlan következménnyel járna, ha mindenki olyan frekvenciasávot használna, amilyet csak szeretne. Ezért születtek országos és nemzetközi intézmények, amelyek kiosztják a frekvenciatartományokat.

Rádiófrekvenciás átvitel

A rádióhullámok hihetetlenül hasznosak, rendkívül sok előnyös tulajdonsággal bírnak, például könnyedén előállíthatók, nagy hatótávolságúak, az épületek sem szabhatnak határt a terjedésüknek és minden irányba egyenletesen terjednek.

A rádióhullámok terjedési tulajdonságai a frekvenciától függően változhatnak. Például az alacsonyabb frekvenciájú hullámok ugyan áthatolnak mindennemű akadályon, a forrástól távolodva azonban teljesítményük egyre jobban csökken. A magasabb frekvenciákon a hullámok inkább egyenes vonalban terjednek és visszaverődnek a házak faláról. Az eső azonban elnyeli őket.

Gondot okozhatnak az elektromos berendezések és a villamos motorok is. Ezek ugyanis a rádióhullámok összes típusát zavarják. A rádióhullámok adatkapcsolatra sem igazán alkalmasak, mivel rendkívül kicsi a sávszélességük. Az alacsony frekvenciájúak azonban akár 1000 km távolságig is használhatóak, és a Föld gömbölyűsége sem okozhat gondot, mivel követik a földfelszínt.

A magasabb frekvenciájú rádióhullámokat azonban a földfelszín elnyeli, ha túlságosan közel vannak hozzá. Szerencsére mégis alkalmazhatjuk őket nagyobb távolságokra is, mivel a légkör rendelkezik egy ionoszféra nevű réteggel. Ebben elektromos töltéssel rendelkező részecskék „laknak”, és a felszíntől számítva a 100 és 500 km közötti sávban foglalnak helyet. Ha bizonyos feltételek adottak, akkor ezt a réteget elérő rádióhullámok visszaverődhetnek. Ezt használják ki a rádióamatőrök, hogy a tőlük nagy távolságokra lévő partnerekkel is beszélgethessenek.

Mikrohullámú átvitel

A 100 MHz feletti elektromágneses hullámok azért jók, mert egyenes vonalban szeretnek terjedni, ezáltal jól fókuszálhatók. Ez azt jelenti, hogyha mondjuk egy parabolaantenna segítségével egyetlen nyalábba fogjuk össze ezeket a hullámokat, akkor az előző részből ismert jel-zaj-viszony nagyon jó lesz. Ehhez vennünk kell a fáradtságot, hogy az adót és a vevőt rendkívüli pontossággal egymáshoz igazítsuk.

A mikrohullámú rendszerek nem új vívmányok, az optikai kábelek megjelenése előtt a távközlésben gyakran használták nagy távolságok áthidalására. Ma már a mikrohullámú rendszereket a legtöbb helyen felváltották az optikai kábelek, mégis a mikrohullámú rendszerek egy vitathatatlan előnnyel rendelkeznek az utóbbival szemben: a kiépítéshez nem kell földet túrni több kilométeren keresztül. Gyakran két adótorony felállítása olcsóbb, mint 50 kilométernyi kábel lefektetése, főleg, ha azt egy zsúfolt városon vagy egy hegységen keresztül kell elvezetni.

A mikrohullámú adatátvitelt azonban nehezíti néhány dolog. Például a Föld nem lapos. A mikrohullámú sugárzás nem követi a földfelszínt, ezért ha nagyobb távolságra szeretnénk adatot eljuttatni, akkor ismétlőállomások szükségesek hozzá. Ezekről általánosságban annyit mondhatunk, hogy minél magasabbak, annál ritkábban kell ilyeneket elhelyeznünk. Ha például 100 méteres ismétlőadókat építünk, akkor körülbelül 80 km-enként egy szükséges belőlük.

Más nehézségek is felmerülnek: a mikrohullámnak az épületek akadályokat jelentenek, sőt a levegő is „beleszól” a dologba, ugyanis a hullámok egy kis része megtörhet (mivel a levegőben szóródnak valamennyire). A megtört hullámok később érhetnek az adóhoz, mint a közvetlenül érkezők, tehát más lesz a fázisuk. Ilyenkor könnyen bekövetkezhet az elhalkulásként ismert rettegett jelenség, amikor is a két hullám kioltja egymást. Az elhalkulás mértéke erősen függ az időjárási viszonyoktól és a használt frekvenciától.

Komoly gond az is, hogy a mikrohullámú sugár nemcsak a házakon, de a vizen sem képes áthatolni. Ez igazából esős időben okozhat gondot, amikor egy felhőszakadás teljesen elnyeli a hullámokat.

Ennek ellenére még ma is nagyon sok helyen alkalmazzák a mikrohullámokat. Megtalálhatóak a távbeszélő rendszereknél, az úgynevezett celluláris telefonhálózatoknál (hogy pontosan mi is ez, erről a következő részekben lesz szó) vagy akár a televíziós műsorszórásban. Sőt a mikrohullámú frekvenciák elég zsúfoltak, már-már a frekvenciahiány réme fenyeget.

A mikrohullámú frekvenciák használatát az állam szabályozza. Akad azonban egy nemzetközileg elfogadott frekvenciatartomány (a 2400 és a 2484 GHz közötti intervallum), amelynek használata nem engedélyhez kötött. Ezért ezt a tartományt alkalmazzák, például bizonyos háztartási eszközök is: vezeték nélküli telefonkészülékek vagy garázsajtónyitók. Meg kell jegyeznünk, hogy ez már egy kellően magas frekvenciatartomány ahhoz, hogy csupán drágább elektronikával lehessen előállítani őket, ráadásul még a mikrosütőnk is megzavarhatja működésüket.

Infravörös hullámú átvitel

Amikor az elektromágneses hullámok spektrumait vizsgáljuk, általánosságában elmondhatjuk, hogy minél jobban közelít az adott hullám frekvenciája a látható fényéhez,

annál inkább fényhullámként viselkedik (azaz nem rádióhullámként). Például annál kevésbé tud áthatolni szilárd tárgyakon. Így van ez az infravörös tartományba eső sugarakkal is, amiket előszeretettel használnak kis távolságokban történő adatátvitelhez.
Mivel az infravörös hullám könnyedén előállítható, ezért hűen szolgál minket nap mint nap olyan nélkülözhetetlen háztartási eszközökben, mint például a tévé távirányítója. Míg a mikrohullámok esetében hátrányként értékeltük, hogy nem haladnak át a házakon, az infránál ezt akár áldásnak is tekinthetjük. Senki sem örülne, ha a szomszéd a fotelből kapcsolgatva a saját tévénket is irányítaná. (Az infravörös frekvenciák használata ezért nem kötött engedélyhez).
A távirányítók mellett fontos felhasználási terület a biztonságtechnika és a mozgásérzékelés. Az utóbbi jó pár évben pedig egyre elterjedtebbé vált az adatátvitelre való felhasználása, főleg, hogy alkalmazása rendkívül kényelmes, még kábeleket sem kell dugdosnunk.
Hátránya viszont, hogy napfényben használhatatlan, mert a Nap ugyanolyan erősen süt az infravörös tartományban, mint a látható fényében.

Látható fényhullámú átvitel

Ez a legősibb elektromágneses hullámokon alapuló kapcsolat, főleg, ha az indiánok által használt füstjeleket is annak tekintjük. Ma már ennél kifinomultabb módszerek is rendelkezésünkre állnak, például a lézer. Nincs más dolgunk, mint két háztömb tetejére egy-egy lézert és egy-egy fényérzékelőt telepíteni. A lézert úgy kell beállítani, hogy az általa kibocsátott fénysugár pontosn az érzékelőre essen. Ez a megoldás egyrészt olcsó, másrészt kellő sávszélességgel rendelkezik. Még engedélyt sem kell kérnünk hozzá, mint a mikrohullám esetében.
Ez így elsőre nagyon jól hangzik, de számos buktatója akad. Egyrészt, ha sűrű a köd és az orrunkig sem látunk, egészen bizonyos, hogy a lézer sem jut sokkal messzebbre, mint a látóhatárunk. Nem sokkal lesz jobb a helyzet a verőfényes nyári nepsütésben sem. Ha kellően meleg a beton, akkor megindul a felfelé irányuló hőáramlás. Pontosan ilyesmit észlelhetünk, amikor az aszfaltot tükröződni látjuk, de a csillagok is ezért ragyognak az égen. (A csillagászok nagy bánatára, akik a hegyek tetejére költözve próbálják kikerülni a légkör zavaró tényezőit). Ez a jelenség eltérítheti a lézersugarakat, így könnyedén elhaladhatnak a fényérzékelő mellett.
Másik műszaki gond lehet, hogyha túl nagy távolságot akarunk áthidalni a lézer segítségével. Ilyenkor elég nagy célzómesternek kell lennünk, hogy a lézersugár pontosan a fényérzékelőre essen. Ezen segíthetünk azzal, hogy lencséket helyezünk a lézer elé, ezáltal kicsit jobban „megszórva" a sugarat.

Távbeszélő rendszerek

A távbeszélő rendszerek már igen hosszú múltra tekinthetnek vissza. Feladatuk eredetileg az volt, hogy az emberi beszédet érthető formában továbbítsák az egyik helyről a másikra. Sorozatunk azonban a számítógépek „párbeszédének" célbajuttatásával foglalkozik. Miért kell mégis a távbeszélő rendszerekkel foglalkoznunk?
Meg kell jegyeznünk, hogy eredetileg a távbeszélő rendszer a számítógépes kapcsolatra nem igazán volt alkalmas. Ma már kicsit más a helyzet, ugyanis jelenleg a telefontársaságok többsége is digitális technikát és fényvezető szálakat használ. Nem ritka, hogy a távbeszélő rendszer egybeolvad a számítógép-hálózatokkal. Ez az oka annak, hogy érdemes szót ejtenünk a távbeszélő rendszerekről.

Távbeszélő rendszer felépítése

Az otthoni telefonunk két rézvezeték (úgynevezett előfizetői hurok) segítségével kapcsolódik telefontársaságunk legközelebbi végpontjához, vagy ha úgy tetszik, helyi központjába. Ide „fut be" a környék összes telefonkészüléke, egy körülbelül 10 km sugarú környezetből. Ezeket a helyi központokat egyértelműen azonosítja az előhívószám, valamint a telefonszám első valahány számjegye.
A végközpontok úgynevezett távhívó állomásokkal állnak kapcsolatban, úgynevezett helyközi tönkök segítségével. Ha a hívó és a hívott fél helyi központja ugyanahhoz a távhívó állomáshoz csatlakozik, akkor a beszélgetés a távhívó állomáson belül jön létre. A „távközlési hiearchiában" ezen kívül létezik még egy magasabb szinten lévő összeköttetés is. Ezek a körzeti, illetve regionális kapcsolóközpontok, amelyeket központi tönkök kötnek össze.
A tönkök mibenlétéről és működéséről a következő részben lesz szó.

Digitális átvitel

A távbeszélő rendszerekben az átvitel eredetileg analóg módon történt. Ezt úgy kell elképzelnünk, hogy a hangjelek váltakozó feszültségjel formájában továbbítódtak. Miután fejlődésnek indult a digitális technika, és szélesebb körben terjedni kezdtek a számítógépek, a telefontársaságok fokozatosan kezdtek átállni a digitális úton történő átvitelre.
Mint ahogy arról már volt szó, a digitális úton történő átvitelnél csak két feszültség szint megengedett, ez rendszerint a +5 V és a -5 V. A digitális átvitel sokkal jobb, mint az analóg.
A digitális átvitel előnye, hogy egyszerre több mindent továbbíthatunk rajta, ezáltal nő a kihasználtság. Például a beszéd mellett küldhetünk képeket, adatot és faxot.
Végül, de nem utolsó sorban: a digitális rendszer üzemeltetése és karbantartása olcsóbb és egyszerűbb. Az olcsóság elsősorban a tengerentúli hívásoknál mutatkozik meg. Ha analóg jelek haladnának a víz alatt, akkor lépten-nyomon erősítőket kéne telepíteni, hogy az analóg hullámforma ne torzuljon el a felismerhetetlenségig. A digitális technikával viszont elég, ha a túloldalt meg tudjuk különböztetni a két jelszintet.
A következő részben innen folytatjuk: mélyebben megismerkedünk a távközlési-rendszerek felépítésével és működésével.

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.

COREBlog

Nincs webfejlesztő rendszer blogcsomag nélkül. Írásunkban egy webújságokhoz szánt, minden földi jóval ellátott, Zope alapú rendszert mutatunk be.

Erovatban az elmúlt év során számos tartalomkezelő (CMS) rendszerként felhasználható nyílt forrású terméket vizsgáltunk meg. Amennyiben újság-, magazin- vagy weblapkiadással foglalkozunk, amelyek tartalma időről időre változik, kétségtelenül sokat nyerhetünk egy ilyen CMS-rendszerrel. Végtére is miért módosítanánk kézzel a hivatkozásokat, fejléceket és egyéb elemeket, ha létezik egy program, amely mindezt megteszi helyettünk? A hagyományos CMS nagy, összetett alkalmazás mivel figyelembe kell vennie a web és szervezeti fajták számtalan típusát.

Bárki új cikkeket hozhat létre a csapatunkban vagy csak a riporterek? Melyik szerkesztő tehet fel cikkeket a webre? Milyen fejlécek és láblécek szükségesek? Milyen típusú keresőrendszerre van szükségünk? A kérdések sora szinte végtelen, így értethető miért olyan bonyolult egy CMS program telepítése és üzemeltetése.

Amennyiben rendszeresen szeretnénk megjelentetni cikkeket, de egy teljes értékű CMS karbantartási munkáit túlzásnak találjuk, érdemes egy pillantást vetnünk a weblogokra. A webloglok, röviden blogok, az 1990-es években jelentek meg személyes naplókként, online noteszekként, ahová az emberek jegyzetelhettek, gyorsan és egyszerűen jelentethettek meg cikkeket. Bár a blogok rendkívül változatos stílusban készülnek, formátumuk meglehetősen hasonló, ez jelentősen csökkenti a program összetettségét, így aztán sokkal könnyebb beállítanunk és kezelnünk őket.

Ebben a hónapban egy nyílt forrású weblogprogramot mutatunk be, valamint a bloggoló közösség berkeiben egyre inkább terjedő néhány szabványt ismertetünk. Utunk során egy pillantást vetünk a Zope alapú COREBlogra, amit a weblogok közt is különösen könnyű telepíteni és üzemeltetni. Igazság szerint személyes érdekek is vezettek egy jó bloggoló csomag felkutatásában. Az utóbbi pár évben számtalan weblogot olvastam, úgy gondoltam itt az ideje, hogy magam is bloggoljak. Keresésem eredménye mostanra valószínűleg már meg is található a ***blog.lerner.co.il*** címen (*1. kép*).

Mi is az a weblog?

A legkülönfélébb alakú és méretű weblogokkal találkozhatunk, ezek többnyire alkotójuk érdeklődését és stílusát tükrözik. Kijelenthetjük, hogy bizonyos alapelemek minden weblogban megegyeznek:

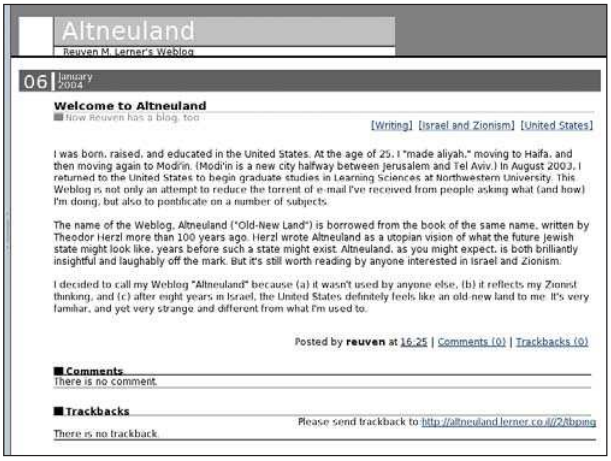
- Sorrend: a küldemények fordított időrendben követik egymást, azaz a legfrissebb cikk kerül az oldal tetejére. A weblog honlap általában csak az elmúlt néhány nap küldeményeit mutatja, a régebbiekhez az archív eszközön keresztül juthatunk el.
- Megjegyzések: a weblogolvasóknak lehetőségük nyílik megjegyzések felvitelére, ezeket gyakran közvetlenül a kérdéses cikk után küldenek be. Ebben a tekintetben a weblogok kicsit hasonlítanak a webalapú fórumokra, azzal a különbséggel, hogy itt csak a bloggoló nyithat új témát.
- Egyszerzős: általában csak egy személy használja a weblogot. Akadnak olyan weblogok is, amelyeket több szerző készít, ez azonban viszonylag ritkán fordul elő. A weblog megjegyzéseit, mint az imént említettük, más emberek viszik fel.
- Hírügynökségek: a weblog tartalmát általában az RSS (Really Simple Syndication, azaz „rendkívül egyszerű hírügynökség") nevű XML formátumon keresztül érhetjük el. Ezen keresztül könnyen vizsgálhatunk és gyűjthetünk különféle weblogokat, tulajdonképpen afféle személyes újságot hozva létre.
- Visszakövetés: a Movable Type kereskedelmi csomag által bevezetett módszer szerint, a visszakövetések segítségével a weblogok nyilvántarthatják az egymás közötti kapcsolatokat.
- Webalapú szerkesztés: minthogy a weblogok kizárólag a weben léteznek, kezelőfelületüket is olyan módon alakították ki, hogy küldeményeink feladását és szerkesztését böngészőnk segítségével tudjuk elvégezni. A weblog ilyenfajta karbantartása valóban értelmes ötlet, de böngészőben írni és szerkeszteni nem éppen a legnagyobb élmény. Sok weblogcsomag nyújt asztali alkalmazásokat e gond enyhítésére. A felhasználók szövegfeldolgozót (word processor) vagy egyszerű a szövegszerkesztőt is használhatnak, majd a kész eredményt utólag feltölthetik. Léteznek ilyen közvetett szerkesztési lehetőséget kínáló nyílt forrású rendszerek is, igaz némi állítgatást igényelnek. Emiatt vagy hozzá kell szoknunk a böngészők szövegmezőjének kezeléséhez, vagy meg kell tanulnunk, hogyan lehet Mozilla alatt kicsit barátságosabb szövegszerkesztőt beállítani. A nyílt forrású Epoz pro-

jektre is vethetünk egy pillantást a ➲http://epoz.sf.net címen, amely böngészőfüggetlen, JavaScript-alapú szerkesztővel rendelkezik.

Mindez azonban nem aranyszabály. Sok weblog egyáltalán nem rendelkezik megjegyzés-, hírügynökség vagy visszakeresés szolgáltatással. De miként az angol nyelvű újságok kifejlesztettek maguknak közös címsorszabályokat, fejléceket és cikkrangsorolást, ugyanúgy a weblogok is kifejlesztettek bizonyos elvárásokat. A képességekért folytatott harc igen kiélezett: amint az egyik weblogcsomag felvesz valamilyen hasznos tudást, azt általában a többiek is hamarosan megvalósítják.
A fenti képességek kialakítása nem lenne túl nagy falat egy tapasztalt web–adatbázis-programozó számára, amennyiben olyan magas szintű nyelvvel dolgozhat, mint a Perl vagy a Python. Ha a cikkek tárolására relációs adatbázis-kezelőt használunk, mondjuk PostgreSQL-t, többé nem kell foglalkoznunk a rendezéssel és tárolással, elegendő a kimenetre összpontosítanunk. Néhány híresebb blogger, többek közt ***Tim Bray***, a kiváló Ongoing weblog (➲ http://www.tbray.org) szerzője, el is készítette saját weblogváltozatát.
Nagyon szeretek új programokat írni, de nem szeretem újra feltalálni a kereket. Látván, hogy mennyi jó weblogkészítő megoldás létezik, köztük olyanok is, amelyek bővítménykezelőn keresztül teszik lehetővé saját képességek írását, úgy döntöttem inkább egy már létező programot használok, azután a hozzá adott API-n keresztül kibővítem és testreszabom. Ráadásul némelyik hasznos weblogképességet, például a megjegyzéseket és a nyomkövetést nem is olyan könnyű létrehozni. Nem azt mondom, hogy különösebb gondot okozna, de tekintve, hogy már 100 000 megjegyzéskészítő létezik a weblog küldeményekhez, nem ragaszkodom a 100 001. változat megírásához.
Nem árt tudni, hogyha esetleg weblog írásra adnánk a fejünket, létezik olyan választási lehetőség is, amikor semmiféle programot nem kell megírnunk, se beállítanunk. Nevezetesen, ha az internet számtalan weblogszolgáltató oldala közül választunk egyet. Ezek a legtöbb embernek tökéletesen megfelelnek, én azonban szeretnék egy kicsit több irányítást a program felett. A tetejébe a weblogomat szeretném beilleszteni a tartományom honlapjába, ami azt jelenti, hogy saját rendszert kell telepítenem.

COREBlog

Amennyiben a legegyszerűbben telepíthető weblogcsomagot keressük, és már ismerjük a Zope alkalmazáskiszolgálót, érdemes egy pillantást vetnünk a COREBlogra. ***Atsushi Shibata*** COREBlog programja folyamatosan fejlesztett Zope-termék, azaz bővítménymodul.
A Zope, mint azt a rovat korábbi részeiben említettük, a Zope Corporation által fejlesztett nyílt forrású alkalmazáskiszolgáló. A Zope nagyrészt Python nyelven készült, objektumközpontú, és az adatok nagy részét a ZODB objektum-adatbázisban tárolja. A Zope alatti fejlesztés erősen eltérő más nyelvek és alkalmazáskiszolgálók megoldásaitól, némi időbe és próbálkozásba beletelik, mire az ember hozzászokik a gondolatvilágához. Ugyanakkor nagyon rugalmas, és a fejlesztők számára lehetővé teszi új modulok (termékek) felvitelét a rendszerbe.



A szerző blogja a Zope alapú COREBlog csomagot használja

A COREBlogot szabványos tarlabdaként kapjuk meg, amelyet a ***lib/python/Products*** könyvtárban kell megnyitnunk a Zope gyökér könyvtárunk alatt. Jelenleg a legfrissebb COREBlog-változat a 0.53b, amit a ***COREBlog053b.tgz*** nevű fájlként tölthetünk le a ➲ http://www.coreblog.org címről.
A most következő lépések során feltételezni fogjuk, hogy a `ZOPE` változó a Zope gyökérkönyvtárára mutat (az én rendszeremen ***/usr/local/zope***) valamint, hogy a COREBlog ***tar*** állománya a ***/tmp*** könyvtárban található:

```
# cd $ZOPE/lib/python/Products
# tar zxvf /tmp/COREBlog053b.tgz
# chown -R zope.zope # avagy a megfelelő
➥tulajdonos/csoport
```

Indítsuk újra a Zope-ot kézzel vagy a webfelület vezérlőpultján, és a COREBlog máris önműködően megjelenik a felvett termékek között.
A COREBlog beüzemeléséhez létre kell hoznunk egy példányt a jobb felső sarokban található ***Add*** menüvel. Állítsuk böngészőnket a honlapunk ***/manage*** URL-ére (például ***www.example.com/manage***), majd válasszuk ki a COREBlogot. Itt egy ID-t (azt az egyedi nevet, ami az URL-ben fog megjelenni), a címet (ami a weblog neveként lesz látható), valamint a karakterkódolást (ami alapértelmezetten ASCII, de én általában UTF-8-ra állítom, hogy teljes Unicode-támogatást kapjak) kell megadnunk.
Ettől kezdve a weblog majdnem készen is áll a nagyközönség számára. A hozzá rendelt ID-re végződő URL-en megtekinthetjük weblogunkat (például ***/atf***), illetve ha a névhez hozzáfűzzük a ***/manage*** szöveget (esetünkben ***/atf/manage***), akkor módosíthatjuk a beállításokat.
Mivel a COREBlog alatt minden küldeményt legalább egy kategóriába bele kell sorolni, mielőtt valamit elküldhetnénk, legalább egy kategóriát létre kell hoznunk. Erre egyébként egy nagy vörös felirat figyelmeztet is bennünket. A blogkezelő képen találunk egy ***Categories*** panelt; kattintsunk rá, és vegyük fel az új kategóriát. Ezt át is nevezhetjük, illetve megnézhetjük, hogy az egyes kategóriákhoz éppen mennyi küldemény tartozik.
A kategória létrehozása után kattintsunk az ***Entries*** panelre, és lássunk hozzá az új megjegyzések megírásának. Minden

bejegyzésnek legalább egy címet, törzset és egy kategóriát szükséges tartalmaznia. A többi érték elhagyható vagy önműködően feltöltődik megfelelő alapértelmezett értékekkel, ilyenek például a pillanatnyi dátum és idő. Miután írásunkat elküldtük a blogra, érdemes megtekinteni a ***Preview*** gombbal, de azonnal ki is adhatjuk az ***Add*** gomb segítségével. Kiadás után történetünk már az egész világ számra olvasható. Bárki, aki olvassa a weblogunkat, az oldal tetején olvashatja a friss küldeményt.
Nem szükséges mindenkinek az oldal tetejét böngésznie, ha új küldeményeket keres. Az oldalléc friss küldeményeket felsoroló része hivatkozik valamennyi új küldeményre. A naptár segítségével tetszőleges napon készült valamennyi küldeményt megtekinthetjük. A témára kattintva pedig a hozzá tartozó küldeményeket nézhetjük meg. Ez azonban csak a témákra működik, az altémákra már nem.

Testreszabás

A COREBlog testreszabásának legegyszerűbb módja, ha a ***Settings*** elnevezésű webalapú tulajdonságszerkesztő eszközt használjuk. A ***Settings*** panelen a honlap különféle oldalainak tartalmát és viselkedését szerkeszthetjük. Például pár szót írhatunk magunkról vagy a weblogunkról, esetleg beállíthatjuk, hogy a megjegyzések alapértelmezetten tiltva legyenek. Amennyiben a blogunkat jellemző színeket és karakterkészletet szeretnénk lecserélni, kattintsunk a ***Skins*** panelre, ahol az ilyesfajta információkat találjuk. Ennél mélyebben is beállíthatjuk a COREBlogot, ha a kimenethez használt DTML-oldalakat írjuk át. Főleg az oldallécet érdemes megnézni (a lap jobb oldalán), amelyet a ***COREBlog/dtml/modules*** állományában tudunk megváltoztatni (a ***$ZOPE/lib/python/Products*** alatt). Ezek az oldalak DTML-ben, a Zope eredeti kiszolgálóoldali sablonozó nyelvén íródtak. Az ***index_html.dtml*** mindössze annyit tesz, hogy a könyvtárban található többi állományt meghívja:

```
<dtml-var calendar>
<dtml-var about>
<dtml-var recent_entries>
<dtml-var recent_comments>
<dtml-var recent_trackbacks>
<dtml-var categories>
<dtml-var archives>
```

Amennyiben a modulok sorrendjét akarjuk megváltoztatni, esetleg csak bizonyos modulokat szeretnénk használni, a legegyszerűbb, ha ezt az ***index_html.dtml*** állományt módosítjuk. Akár saját modulokat is készíthetünk: DTML állományokat készítünk, majd a COREBlog API segítségével lekérdezzük a blogban leírt adatokat és a kiadott utasításokat.
Természetesen, minthogy a COREBlog nyílt forrású projekt, bármilyen részt módosíthatunk tetszés szerint. Az igazat megvallva, nincs túl sok minden, amit módosítani lehetne, a COREBlogba épített képességek általában éppen azok, amiket a saját weblogunkba is szívesen beletennénk. Tekintve, hogy a projekt folyamatos fejlesztés alatt áll, hamarosan további képességek megjelenése is várható.

Hírügynökség

Miután beindítottuk bloggoló bölcseletünket, az emberek naponta többször eljönnek megtekinteni a legújabb bölcsességeinket és érdekes hivatkozásainkat. Akik egynél több weblogot szeretnének használni, vagy egyesíteni szeretnék blogjukat valamilyen hírújsággal, az Aggregator program segítségével több különböző oldalról is összegyűjthetik az adatokat. Az Aggregátorok nem közvetlenül a HTML formátumot nézik; hanem az RSS alapokon közreadott hírügynökségi adatot.
A COREBlog alatt nagyon egyszerű belépni a hírügynökségbe. Tulajdonképpen semmit sem kell tennünk, amennyiben weblogunk küldeményeit be szeretnénk tenni a hírügynökségbe, ugyanis a COREBlog önműködően kezeli a dolgoknak ezt a részét, a népszerű RDF és RSS formátumokat egyaránt elő tudja állítani. A COREBlog már működő hírügynökséggel kerül hozzánk.

Összegzés

A tartalomkezelő rendszerek rendkívül hasznosak azoknak a szervezeteknek, amelyek fel is tudják használni őket. Ha azonban csak a saját honlapunkat használjuk, egy teljes értékű CMS rendszerrel kicsit túllövünk a célon. Az elmúlt néhány évben rengetegen fogtak weblogok kiadásába, s bár a weblog programok még mindig viszonylag gyerekcipőben járnak, e termékek mindenkinek megkönnyítik az életét, akik rendszeresen megjelenő, könnyen érthető, általános formátumban szeretnék írásaikat megjelentetni.
A COREBlog weblogok készítésére és kezelésre tervezett program, amely úgy tűnik alapkészletként nagyon jól beválik. Bővítménykezelő rendszerrel is rendelkezik, így a teljes rendszer újraírás a nélkül is fel tudunk vinni új modulokat. Körutunkat egközelebb a nyílt forrású weblog programok világában folytatjuk, és néhány más csomagot hasonlítunk össze a COREBlog rendszerrel.

Linux Journal 2004. április 120. szám

Reuven M. Lerner (➲ http://www.lerner.co.il/atf) Nyílt forrású programokra, valamint web- és adatbázis-alkalmazásokra szakosodott tanácsadó. Könyve, a Core Perl, 2002 januárjában jelent meg a Prentice Hall gondozásában. Reuven feleségével és lányaival nemrég költözött Chicagóba.

KAPCSOLÓDÓ CÍMEK

A COREBlog oldala ➲ http://www.coreblog.org
A Zope a ➲ http://www.zope.org, a Python pedig a ➲ http://www.python.org honlapon érhető el.
Jó bemutatót olvashatunk a visszakeresésekről a Movable Type szerkezet kiötlőitől, ha megtekintjük a ➲ http://www.movabletype.org/trackback/beginners vagy a ➲ http://www.cruftbox.com/cruft/docs/trackback.html lapot.
Az RSS és hírügynökségekkel kapcsolatos adatokat, a ➲ http://www.xml.com/pub/a/2002/12/18/dive-into-xml.html címen érdemes keresni.

François, tudsz titkot tartani?

Ismerjük meg a nyilvános kulcsú titkosítás alapjait, és a GNU Privacy Guard csatlakoztatását levelezőprogramunkhoz.

Valóban szégyen, François, a következő alkalomra egyszerűen meg kell győződnünk arról, hogy elérhető-e. Minden más hibátlannak tűnik, mon ami, az összes munkaállomás a beléptetésre készen áll. Nagyszerű! Óh, látom, vendégeink már meg is érkeztek. François, irány a borospince keleti szárnya, az öreg zárt ajtó mellett találsz egy kis készletet az 1999-es Côte-Rôtie-ból. Igen, azt az ajtót soha nem tudtuk kinyitni. Siess François, megígérem, hogy semmi ijesztőt nem találsz arrafelé. Foglaljatok helyet, mes amis. A mai menünk majdnem tökéletes, de egy fogás sajnos hiányzik belőle. Talán a Côte-Rôtie Rhône, ez a pikáns és misztikus vörös enyhíteni fogja hiányérzetünket. Mára nagyon szerettem volna elkészíteni a híres Crème Linuxaise-t, de gondok adódtak. Tudjátok, ez egy régi családi recept, nem szeretném megkockáztatni, hogy rossz kezekbe kerüljön. Elektronikus levélben való küldését sem vállaltam attól tartva, hogy valaki lehallgatja a hálózatot, különben François már elő is készíthette volna. Ilyen nagy titokról van szó!

A következő alkalommal mindez már nem jelenthet gondot, ugyanis az étterem minden felhasználóját GnuPG-vel és nyilvános kulcsú titkosítással szerelem fel, így kényes adatokat is biztonságosan küldhetünk majd. A GnuPG teljes neve GNU Privacy Guard (GNU magánéletőr).

A GnuPG az üzenetek és adatok általános titkosítását teszi lehetővé. Ez a PGP (Pretty Good Privacy) jogdíj nélküli, nyílt forrású megfelelője. Számos linuxos levelezőprogram teszi lehetővé, hogy GnuPG-vel titkosított leveleket küldjünk és fogadjunk, ezt szeretném ma bemutatni nektek.

A legtöbb telepítőkészlet tartalmazza a GnuPG-t, amennyiben a feltelepített rendszerünkön nem találjuk, először a telepítőlemezeken érdemes keresgélnünk. A legfrissebb változatot a ➲ http://www.gnupg.org címről is letölthetjük, de előbb járjuk körül egy kicsit a témát.

Hol is tartottam? Ja igen. Az előzményeket megvizsgálva, láthatjuk, hogy minden titkosítási eljárás egy megosztott kulcsállományon alapul. A kulcsot, amellyel az üzenetet titkosítottuk, annak a személynek adjuk oda, akivel titkosított kapcsolatot fenn szeretnénk tartani. Gondoljunk a titkos dekódoló gyűrűre, és már nem is állunk messze az igazságtól. A baj csak az, hogy aki megszerzi a kulcsot, minden üzenetünket meg tudja fejteni. A GnuPG esetén az üzenetek titkosítására két kulcsot használunk, az egyik

1. kép A KMail a kulcs importálásának megerősítését kéri

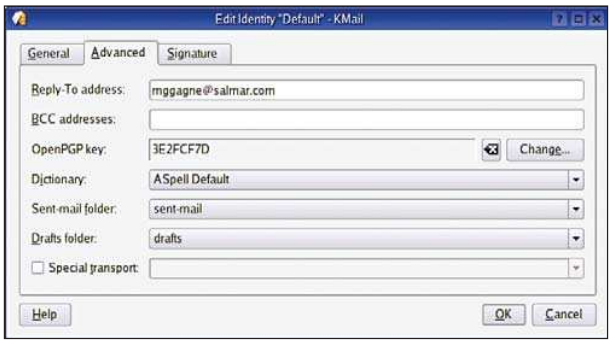
2. kép A KGpg-vel nem jelent gondot a kulcsok kezelése

a személyes kulcs (private key), amit féltve őrzünk és senkinek nem adunk ki. Amikor egy üzenetet titkosítunk, a személyes kulcsunk és egy nyilvános kulcs (public key) segítségével tesszük. A nyilvános kulcs nem a sajátunk, hanem azé a személyé, akinek az üzenetet küldeni szeretnénk, például François-é. A titkosító-visszafejtő folyamathoz mindkét kulcs szükséges, és ha valaki csak az egyikkel rendelkezik, nem megy vele semmire, ezért nem adjuk ki soha a kezünkből a személyes kulcsunkat. Ahhoz, hogy ilyen szupertitkos tevékenységet folytathassunk, előbb el kell készítenünk a kulcspárunkat, amely egyrészt a személyes (nagyon jól őrzött) kulcsunkat, másrészt pedig a nyilvános (a barátainknak szétküldendő) kulcsunkat tartalmazza. Ezt az alábbi paranccsal tehetjük meg:

```
gpg --gen-key
```

Ezt egy kis kérdés-felelet szakasz követi. Az első kérdés a titkosító algoritmussal, a rejtjelezéssel kapcsolatos. Az alapértelmezett a DSA és ElGamal, fogadjuk is el. Amikor a kulcs hosszát kell megadnunk, a 768, 1024 és 2048 közül

3. kép A KMail beállítása a GnuGP-vel való titkosítás használatára



4. kép A nyilvános kulcsunk megadása a KMailben

választhatunk. Mivel a DSA szabványos kulcshossza 1024, most válasszuk ezt. Ezt követően a kulcs érvényességének lejártát kell megadnunk, alapértelmezett esetben az érvényesség nem jár le, de valahány napot, hetet, hónapot, akár még évet is választhatunk. Egyelőre itt ismét fogadjuk el az alapértelmezett beállítást és erősítsük is meg választásunkat. Végül a program a kulcs használójának nevét, elektronikus levélcímét és egy megjegyzést kér tőlünk.
Már csak a jelszókifejezés megadása maradt hátra, ez a befejező lépés. Olyat válasszunk, ami biztonságos, de megjegyezhető. Amikor ezt is beírtuk, a GnuPG előállítja számunkra a kulcspárunkat. Az eredményt pedig a saját könyvtárunkban lévő *.gnupg* könyvtárban ellenőrizhetjük:

```
$ cd /home/marcel/.gnupg
$ ls
gpg.conf      pubring.gpg   random_seed
secring.gpg   trustdb.gpg
```

A *gpg.conf* fájl a gpg parancs alapértelmezett beállításait tartalmazza, ezért hasznos olvasmány. A *pubring.gpg* és *secring.gpg* fájlok rendkívül fontosak. Azonnal készítsünk róluk biztonsági másolatot, és helyezzük biztonságba a számítógépen kívül. Ezeket nem szabad elveszítenünk – a *secring.gpg* tartalmazza a személyes titkos kulcsunkat. Az utolsó fájl, a *trustdb.gpg* a bizalmi adatbázis. Ez azt a bizalmi szintet határozza meg, amit az összegyűjtött nyilvános kulcsokhoz rendelhetünk. Ahhoz, hogy a felhasználók egymás közt titkosított adatokat cserélhessenek, a kulcsaikat is egymás rendelkezésére kell bocsátaniuk. Kitalálhatjuk, hogy kulcs exportálására a `--export` kapcsolót használhatjuk, ami mellé valószínűleg a `-a` kapcsolót is be fogjuk írni, hogy a kimenet ASCII formátumú legyen:

```
gpg -a --export 3E2FCF7D > marcelkey.asc
```

A létrejövő fájl egy egyszerű ASCII szövegfájl; hogy miként juttatjuk el a címzettekhez, rajtunk múlik. Léteznek olyan kulcsok tárolására szolgáló kiszolgálók, ahova a nyilvános kulcsunkat feltölthetjük, bárki számára letölthetővé téve őket (ez jól jöhet, ha sokfelé szeretnénk szétosztani a kulcsunkat). Sőt kulcscserélő összejöveteleket (key-signing parties) is rendeznek, amelyeken a tagok összejönnek, és kicserélik nyilvános kulcsaikat vagy levélmellékleteiket.
A kulcs fogadójának a következő módon kell importálnia a kulcsot:

```
gpg --import marcelkey.asc
```

Az alábbi paranccsal bármikor megnézhetjük a kulcskarikánkon összegyűjtött kulcsainkat:

```
gpg --list-keys
```

A parancs eredménye attól függ, hogy hány kulccsal rendelkezünk, és a következő listából láthatjuk, hogy mire is számíthatunk. Különösen az 1024D után következő hexadecimális szám fontos, ez ugyanis a kulcsazonosító, amelyre a jövőben hivatkozni fogunk:

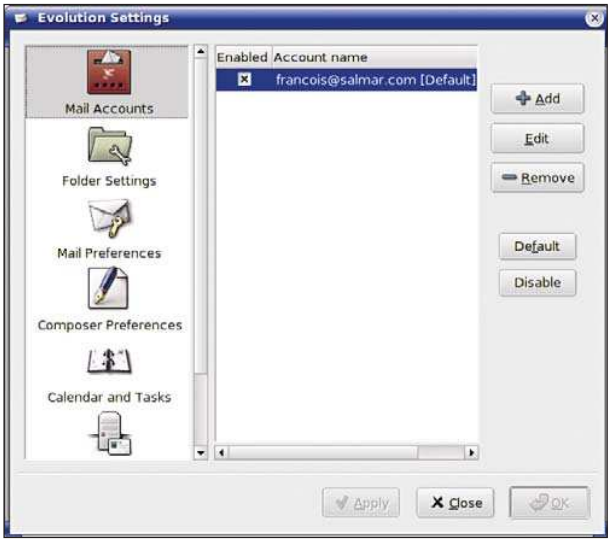
```
/home/marcel/.gnupg/pubring.gpg
-------------------------------
pub  1024D/3E2FCF7D 2004-01-07 Marcel Gagné
(Writer and Free Thinker at Large)
<mggagne@salmar.com>
sub  1024g/B24717BE 2004-01-07

pub  1024D/EE392B87 2004-01-07 Francois (I am but
a humble waiter) <francois@salmar.com>
sub  1024g/F4E07040 2004-01-07
```

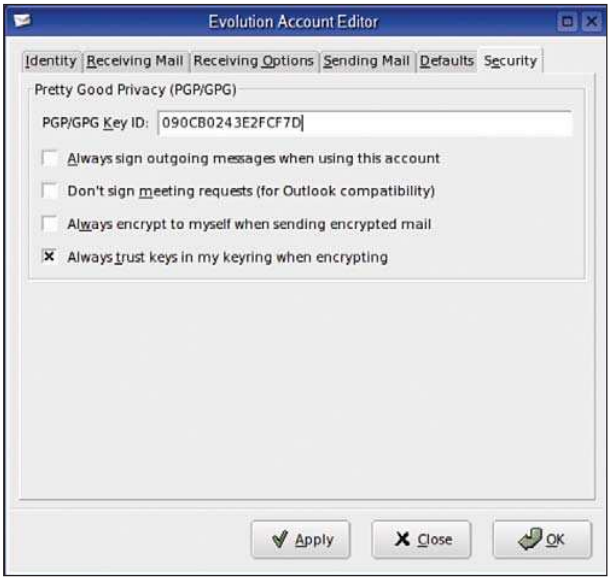
Mielőtt barátaink kulcsaival leveleket kezdenénk kódolni, először az érvényesség ellenőrzésével hitelesítenünk szükséges őket. Ez két lépésből áll. Ha teljesen biztosak vagyunk a kulcs eredetét illetően, az elsőt kihagyhatjuk, ugyanis ez a kulcs ujjlenyomatát szerzi meg:

```
$ gpg --fingerprint francois
pub  1024D/EE392B87 2004-01-07 Francois (I am but
a humble waiter) <francois@salmar.com>
Key fingerprint = 8C5B 775C 33F8 E97C 5ADC  019D
➥C6C8 4B83 EE39 2B
sub  1024g/F4E07040 2004-01-07
```

Láthatjátok, hogy a fenti parancsban a kérdéses személy nevét használtam, ez része a kulcshoz tartozó adatoknak.



5. kép Az Evolution levélfiók-szerkesztője



6. kép A GnuPG-kulcsunk megadása az Evolutionben

Ha a kulcsadatok között több azonos nevű kulcsunk akad, helyette a kulcsazonosítót is használhatjuk. Kérjük meg barátunkat – ennek az ujjlenyomatnak az ellenőrzéséhez –, hogy ugyanezzel a módszerrel ő is ellenőrizze személyes kulcsának ujjlenyomatát. A befejező lépés a kulcs hitelesítése. Ezt a `--edit-key` kapcsolóval tehetjük meg:

```
gpg --edit-key francois
```

Egyszerű az egész folyamat. A program megerősítést kér arra vonatkozóan, hogy tényleg hitelesíteni szeretnénk-e ezt a nyilvános kulcsot, majd végső nyomatékként a jelszókifejezést is bekéri. Ezt minden egyes személlyel végig kell csinálnunk, akivel titkosított leveleket szeretnénk váltani, de ha egyszer elkészültünk, többé semmi sem állhat a titkos üzenetek áramlásának útjába.

Mellesleg számos tetszetős grafikus eszköz létezik a GnuPG-hez, ezek lényegében barátságos külsőt adnak a parancssoros eszköznek. A KDE részeként kapjuk a KGpg-t, amely jól beépül munkakörnyezetünkbe és levelezőrendszerünkbe. Például, ha valaki egy levél mellékleteként egy nyilvános kulcsot küld a KMailen keresztül és a KGpg (a parancs neve `kgpg`) fut, akkor ahelyett, hogy a mellékletet fájlként kellene mentenünk, meghívjuk a parancssort és végrehajtjuk a fenti lépéseket. Az *1. képen* láthatóhoz hasonló barátságos felugró ablaknak kell megjelennie.

Ezzel a tálcán helyet foglaló kis eszközzel szerkeszthetjük is a kulcsokat; újakat vehetünk fel, eltávolíthatunk, megváltoztathatjuk a bizalmi szintet, vagyis mindenre képes, amit a parancssoros GnuPG-vel is végrehajthatunk, de itt csak egy egérkattintásba kerül. Ezenkívül a világhálón lévő kiszolgálókon kulcskeresést is végre tud hajtani, illetve fotóazonosítást használ. Az egyéb KDE-eszközökbe való beépítése annyit jelent, hogy a titkosításhoz a húzd és ejtsd módszert használhatjuk, és a Konquerorból vagy a vágólapról is egyetlen egérkattintással elérjük a GnuPG lehetőségeit.

A KDE 3.1-esnél a KGpg még csak kiegészítés, a KDE 3.2-es változatban azonban már a csomag szerves részét alkotja. A *2. képen* működés közben láthatjuk a KGpg-t.

Egy másik grafikus felügyeleti eszköz, amely figyelmet érdemel, a `gpa`. Ez a GnuPG projekt alapértelmezett GNU Privacy Assistant programja és hivatalos kulcskarika szerkesztője, amely a ➲http://www.gnupg.org címről tölthető le.

Itt az ideje, hogy a titkosított levelek küldését ki is próbáljuk. Nyissuk meg a KMailt, kattintsunk a menüsor ***Settings*** (beállítások) menüjére és válasszuk ki a ***Configure KMail*** (beállítások) menüpontot. Ennek hatására a ***Configure KMail*** (beállítások – KMail) párbeszédablak fog megjelenni. A bal oldalon lévő sávban keressük meg a ***Security*** (biztonság) ikont (*3. kép*).

A jobb oldalon megjelenő ablakban három fület láthatunk: ***General*** (Általános), ***OpenPGP*** és ***Crypto Plugins*** (titkosítási bővítőmodulok). Pillanatnyilag az ***OpenPGP*** fül érdekel bennünket. Ezen a fülön keressük meg a ***Select encryption tool*** (a használni kívánt titkosítási eszköz) feliratú lenyíló listát és ebből válasszuk ki a ***GnuPG – GNU Privacy Guard*** tételt. Egyelőre ne válasszuk ki a ***Sign and encrypt all messages automatically*** (az üzenetek önműködő aláírása és titkosítása) lehetőséget. A jelszókifejezést érdemes megőrizni a memóriában (***Keep the passphrase in memory***), de a program az első titkosított levél előfordulásakor így is kérni fogja. Kattintsunk az ***Apply*** (alkalmazás) gombra.

Ezután kattintsunk a bal oldalon lévő ***Identity*** (azonosító) fülre. Hacsak nem hoztunk létre több levelezési fiókot, itt egyetlen bejegyzést találunk. Kattintsunk a ***Modify*** (módosítás) gombra és a megjelenő párbeszédablakon válasszuk ki az ***Advanced*** (speciális) fület. E panel közepén (*4. kép*) találunk helyet az OpenPGP kulcs számára. Ez a saját, személyes kulcsunk. Kattintsunk a ***Change*** (Módosítás) gombra, ennek hatására egy ablak jelenik meg, amelyben a személyes kulcsunkhoz tartozó adat helyét jelölhetjük ki.

Ha készen vagyunk, kattintsunk az ***OK*** gombra az ***Edit Identity*** (azonosító módosítása) ablak bezárásához, majd ismét az ***OK*** gombra, amellyel ezúttal a ***Configure KMail*** (beállítás – KMail) ablakból lépünk ki. Ahhoz, hogy a KMail megfelelő módon újra beolvassa a beállított értékeket, állítsuk le a programot, majd indítsuk el ismét.

Elvileg készen állunk rá, hogy titkosított leveleket küldjünk. Ha a kulcskarikánkon már szerepel barátunk nyilvános kulcsa és hitelesítettük is, nincs más teendőnk, mint megírni a levelet. Nyissunk egy új levelet és írjuk bele a mondandónkat. Ha küldésre készen állunk, kattintsunk az üzenet menüsorának ***Options*** (jellemzők) menüjére. Itt két titkosítással kapcsolatos lehetőséget vehetünk észre. Az egyik a ***Sign Message*** (az üzenet elektronikus aláírása), a másik pedig az ***Encrypt Message*** (az üzenet titkosítása). A levél aláírása azt jelenti, hogy a nyilvános kulcsunkat elektronikus aláírásként mellékeljük az üzenethez, de nem titkosítjuk. A levél címzettje ily módon rendelkezik egy eszközzel annak ellenőrzésére, hogy a levél valóban a feladótól származik, ha meg akar győződni erről. Mivel az üzenet nem titkosított, bárki elolvashatja. Ez az eszköz csak annak megerősítésére szolgál, hogy valóban attól a személytől kaptuk a levelet, akit feladóként látunk feltüntetve. A titkosítás annyiban jelent további lépést, hogy azt a nyilvános kulcsot használjuk, amit a barátunk bocsátott rendelkezésünkre a titkosításhoz. A KMail mindkét esetben az üzenet elküldése előtt kérni fogja a jelszókifejezést.
Sajnos nem minden levelezőprogram használja ugyanazokat a szabályokat a titkosításhoz. Túl egyszerű is lenne, non? Feltételezett esetünk leírásához használjuk fel a 17-es asztalnál ülő két személyt, Larryt és Michaelt. Mindketten kényes üzleti adatokat szeretnének küldeni, tehát az információknak titkosított formában kell haladniuk. Larry a levelezéséhez az Evolutiont használja, Michael pedig a KMailt. A KMail esetében már megismertük a titkosított levelek kezelésének módját, most pedig vizsgáljuk meg, hogyan történik mindez a Ximian Evolution esetében. Kattintsunk a futó Evolution programunk bejövő üzeneteket tároló könyvtárára. Ezután válasszuk ki a menüsor ***Tools*** (eszközök) menüjének ***Settings*** (beállítások) menüpontját. Ekkor az ***Evolution Settings*** (az Evolution beállításai) ablaknak kell előttünk lennie (*5. kép*). Amennyiben a ***Mail Accounts*** (postafiókok) ikon a bal oldali navigációs sávban még nincs kiválasztva, tegyük meg. A legtöbb embernek egyetlen fiókja van. Ha mégis többel rendelkezünk, válasszuk ki azt, amelyiket a titkosított üzenetek küldésére szeretnénk használni, majd kattintsunk a jobb oldalon lévő ***Edit*** (szerkesztés) gombra. Erre a számos fület tartalmazó ***Account*** (postafiók-szerkesztő) párbeszédablak jelenik meg. A bennünket pillanatnyilag érdeklő fül a ***Security*** (Biztonság) feliratot viseli. Keressük meg a ***PGP/GPG Key ID*** (PGP/GPG kulcs azonosítója) feliratú mezőt és ide írjuk be kulcsazonosítónkat. A párbeszédablak bezárásához kattintsunk az ***OK***-ra, majd az Evolution beállítóablakából való kilépéshez ismét válasszuk az ***OK*** gombot. Feltéve, hogy Larry és Michael kicserélték nyilvános kulcsaikat, Larry készen áll rá, hogy titkosított üzeneteket küldjön. Ehhez először a megszokott módon létrehoz egy üzenetet, majd még az elküldés előtt kipipálja a menüsor ***Security*** (biztonság) menüjéből a ***PGP Encrypt*** (PGP titkosítás) lehetőséget. Amikor Larry a ***Send*** (küldés) menüpontra kattint, a program kérni fogja tőle a GPG jelszókifejezését, amennyiben tudja, az üzenetet azonnal elküldi. Mielőtt továbblépnénk, meg kell említenem, hogy a ***Security*** (biztonság) menüből ugyanígy választható ki a ***PGP Sign*** (PGP-aláírás) lehetőség is, ha az üzenetet csupán alá szeretnénk írni.

Bár ez a kódolás-dekódolás nagyszerű, akadnak azért nehézségek is. Például nagyon sok levelezőprogram (a KMail is ezek közé tartozik) a levélen belül végzi el a kódolást, míg néhány más program, így az Evolution is, más utat követ, ezeknél a kódolt üzenet MIME-csatolmányként jelenik meg. Ebből következően egy KMailből (vagy Eudorából, vagy Outlookból) érkező levelet az Evolutionben csak úgy tudunk megnézni, ha előtte szöveges állományként mentjük, majd a következő paranccsal dekódoljuk:

```
gpg -d message.txt
```

Egy Evolutionban kódolt, KMailnek (vagy néhány másik programnak) küldött levél esetében a csatolt fájlt – és nem a levelet magát – szükséges hasonlóan mentenünk, majd a fenti módszerrel dekódolnunk.
Nagyon sok levelező ügyfélprogram érhető el Linuxra. A KMail és az Evolution népszerű grafikus programok, ugyanakkor a Mozilla sem maradhat ki a sorból. A Mozilla használói számára az Enigmail nevű bővítmény (plugin) teszi lehetővé az üzenetek zökkenőmentes aláírását és titkosítását. A szöveges ügyfélprogramok szintén támogatják a titkosítást, akár önmagukban, akár bővítményeiken keresztül, erre a `mutt` és a `pine` mutat két népszerű példát.
Mon dieu! Ilyen hamar elszaladt az idő? Attól tartok, mes amis, hogy a záróra megint utolért bennünket. François még egyszer teletölti a poharaitokat, mielőtt elválnánk. Miközben kényelmesen szopogatjátok az utolsó kortyokat, hozzátok létre és cseréljétek ki egymással a saját nyilvános kulcsaitokat. Talán valamelyiktekkel megosztom majd a Crème Linuxaise receptjét – természetesen a megfelelő biztonsági intézkedések megtétele mellett. Ki tudja, talán éppen a túlzott biztonság az oka, hogy az az ajtó a pincében még mindig ilyen szorosan zárva van. Viszontlátásra a következő alkalommal! A vôtre santé! Bon appétit!

Linux Journal 2004. április 120. szám

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

KAPCSOLÓDÓ CÍMEK

A Gnu Privacy Guard (GnuPG) oldala
➲ http://www.gnupg.org
A KMail honlapja
➲ http://kmail.kde.org
A Mozilla elérhetősége
➲ http://www.mozilla.org
A Mozilla Enigmail webcíme
➲ http://enigmail.mozdev.org
A Ximian Evolution oldala
➲ http://www.ximian.com/products/evolution

Neverwinter Nights: Hordes of The Underdark

Ritka alkalom, hogy nem játékról, hanem egy kiegészítő lemezről írok. E kivételezésnek személyes „fertőzöttségem” az oka, amit a Nevewinter Nights váltott ki. Ugyanakkor mégsem teszek kivételt, hiszen ez a kiegészítő egy kicsit „új játék”, tele izgalmas és érdekes dolgokkal.

A Neverwinter Nights egy RPG (Role-Playing Game), tehát szerepjáték, amely a D&D 3 szabályrendszerére épül. Ebben a játékban számos varázslat és kaland vár a világba belépő harcosra. Minden játékos először, a szerepjátékok szabályait követve, egy karaktert hoz létre, melynek tulajdonságait és képességeit meg kell határoznia. Ezt a továbbiakban természetesen kedve szerint fejlesztgetheti is. A szerepjátékokban a fejlődés az a vonás, ami a játékosokat megfogja. A számos karakterosztály és a létrehozható saját karakterek sokféleségén túl, a rengeteg kiegészítő, a lények sokszínűsége teszi olyan változatossá a játékot, hogy gyakorlatilag kétszer ugyanúgy sosem tudjuk végigjátszani. A kalandozásunk azonban itt még nem ért véget, hiszen a Bioware csapata eddig két kiegészítőlemezt is készített az alapjátékhoz: a Shadows of The Undertide-ot és a Hordes of The Underdarkot. Természetesen a megjelenésüket követően, nem telt el sok idő, és a linuxos játékosok is élvezhették ezeket, ugyanis a Bioware elkészítette a linuxos telepítőt is. A telepítők a ➲ http://www.bioware.com oldalról tölthetők le.

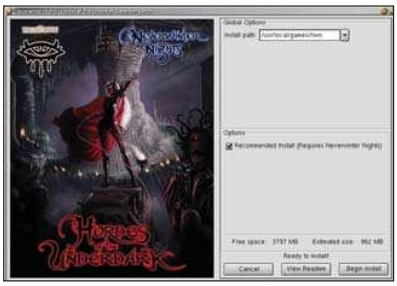
Ha a telepítőt letöltöttük, maga a telepítés gyakorlatilag megegyezik a jól ismert módszerrel, vagyis a CD-t a meghajtóba helyezve, és a telepítőt elindítva tehetjük fel a játékot. Fontos kiemelni a sorrendet: mindig kövessük a kiadási sorrendet, tehát az alapjáték, és a Shadows The Undertide után tegyük fel ezt a kiegészítőt.

Tapasztalataim szerint, azért fontos a sorrend, mert ha nem ilyen módon járunk el, akkor működésképtelen játékot kapunk. Természetesen a játék felpakolásához szükségünk lesz a CD-kulcsra is, amely a leírásában található meg.

Lássuk, mit nyújt a kiegészítő!

Az alapjátékban mindössze a huszadik szintre tudtuk fejleszteni a karaktereket, ezt az értéket tudtuk növelni a Shadows of The Undertide-ban. Nos, most még tovább léphetünk, a Hordes of The Underdarkban a karakterfejlődés (elméletileg elérhető) szintje negyvenre módosult. A karakterosztályok is tovább bővültek, pontosan hat új karakterosztály található a kiegészítésben, például a Red Dragon Disciple, Dwarven Defender, Weapon Master és a Champion of Torm. Most már összesen 16 új lény igyekszik megkeseríteni az életünket, közülük is kiemelném a hárpiát, a mithrill gólemet, valamint az adamtite gólemet, aki igen kellemetlen egyéniség. Két új sárkány is megjelent, az egyik – a prizma sárkány – igen finoman kidolgozott karakternek ígérkezik. Természetesen ez nem azt jelenti, hogy a többi sárkány a fél fogunkra sem elég, hiszen aki harcolt már sárkánnyal, az tudja, hogy mindegyik igen veszélyes. Ennek következtében minden összecsapás előtt ajánlatos alaposan „feltápolni” a karaktert, lehetőleg combos tárgyakkal.

Az új környezetek kialakításáért négy új tileset, azaz grafikai elemkészlet



felel, ezekkel az agyzabálók barlangjait vagy a mélységi gnómok földalatti labirintusait ismerhetjük meg. Természetesen egyéb karaktertulajdonságok is belekerültek, például hősi képességek, elsöprő kritikus ütés vagy fejlesztett kritikus ütés, valamint jó néhány új varázslat is kaptunk, mint a Halálpáncélt vagy Shelgran állandó pengéjét. Természetesen mindezeken kívül a fellelhető tekercsek és tárgyak listája is bővült, ezeket dióhéjban képtelenség lenne felsorolni. Továbbá a már megszokott tárgyakból, fegyverekből is kapunk újakat, illetve az ismertek közül erősebbekhez (például kétkezes kard +5) juthatunk.

Hangulat

A játék hangulata nagyon jó, az eddigiek remek folytatásának bizonyult. A hangok szintén jól sikerültek, bár az előzőekhez képest nem túl sokat változtak. A zene viszont nem mindig illeszkedik a környezethez. Tudom, ez az ítélet az én ízlésemen alapul, tehát vélhetőleg elfogult, de nekem valahogy nem jött be. Az alapjáték zenéje a sok kóborlás során sem volt számomra idegesítő, itt viszont helyenként határozottan zavart. A szörnyek hangjai és a párbeszédek azonban példa értékűek.
A grafika változatlanul nagyon szép, ugyanis a grafikai elemkészletek egységes összképet alkotnak. Egyetlen esetben sem láttam olyan textúrát, amelynél felmerült volna, hogy az adott környezetbe vagy világba nem simul bele. A játékmenet gördülékeny, bár az elgondolkodtató részeknél bizony sokáig el lehet sétálgatni egy-egy helyen, habár sehol sem akadtam el komolyabban. Mindazonáltal itt is érvényes az a szabály, hogy próbáljunk meg a létező összes karaktertől küldetést begyűjteni, mert helyenként nagyon kell az XP. Tehát nem árt a karakter szintjét felhúzni, mert helyenként igen „tápos" ellenfelek jönnek szembe.
A főellenfeleket szinte öngyilkosság oly módon megtámadni, hogy nem aggattuk magunkra a lehető legtöbb varázstárgyat. Olykor bizony még a könnyebben játszható Clericem is megszenvedett a sikerért.

Gépigény

A gépigény szinte semmit nem változott az alapjátékéhoz képest, természetesen figyelmen kívül hagyva azt a tényt, hogy a nagyobb területek vagy a részletesebb textúrák több erőforrást vesznek el. Az én gépemen (Duron 800 MHz-es processzor, 512 DDR memória, GeForce2 videokártya) 800×600-as felbontásban remekül játszható, és a komoly varázslatok közepette sem lassult le nagyon. Azonban mindenkinek erősebb gépet ajánlok, ugyanis az 1024×768-as felbontáshoz elkél a lóerő.

Összegzés

A Bioware csapat ismét egy remek játékot alkotott, melynek rendkívül jó a hangulata, valamint a játékmenete, ezért mindenkinek csak ajánlani tudom. Amennyiben megszállott szerepjátékos vagy és a Neverwinter Nights és a Shadows Of Undertide már megvan, bátran vedd meg ezt is, mert megéri, nagyon jó kikapcsolódás. Ha azonban csak most szeretnéd beszerezni, szerintem a Gold Editionnel jársz a legjobban. Egyrészt ilyen módon olcsóbb is, mintha különkülön vennéd meg őket, másrészt egyszerre kapod meg az összes kiegészítőt, és azonnal végig kalandozhatod az egész „hőseposzt".
További jó játékot!

Dancsok Zoltán
(strogg@mail.tvnet.hu)
Jelenleg technikai szerkesztőként dolgozik a BME-OMIKK-nál, ahol oktat is.