Készítsünk Live CD-t!

Készítsük el saját, különleges live CD terjesztésünket az itt bemutatásra kerülő nem túl ismert rendszerindító CD trükkök alapján.

Biztosan mindenki hallott már a ***Knoppixról***, arról a ***Debian*** alapú terjesztésről, amely 2GB alakalmazást zsúfol egyetlen CD lemezre. Használják ***Linux*** bemutató eszközként, mentőlemezként, sőt még ***Debian*** telepítőként is. Hasonló projektek egész kis seregét ihlette, az egy-két plusz és mínusz csomagot tartalmazó ***Knoppix*** CD-ktől kezdve egészen a rendszer teljes újratervezéséig.
Jómagam mostanában fogtam neki egy termékbemutató CD készítésének. Kíváncsiságtól hajtva darabokra szedtem a ***Knoppix*** CD-t míg végül egy ***Makefile*** és néhány mellékes állományra jutottam, amelyek ugyan elég egyértelműen ***Knoppix*** ihletésűek de azért van bennük némi új kód is. Itt tartok most.

Rövid Körséta

Ha a ***Knoppix*** CD-t betesszük a meghajtóba és felcsatoljuk, hamar feltűnik, hogy nem nagyon hasonlít a hagyományos ***Linux*** telepítésekre. Van ugyan néhány grafikus fájl és néhány zenesáv, de nincs ***init***, nincs ***/dev*** és nincs ***/bin***. A mágia a ***/KNOPPIX/KNOPPIX*** nevű, igen méretes állományban van, amely a ***cloop*** eszközzel tömörített ***ISO9660*** lenyomat.
A rendszermagban található megszokott ***loop*** eszköz lehetővé teszi, hogy néhány fájlrendszeren úgy érjük el az állományokat mintha azok valódi eszközök lennének. Az eszköz blokkjainak elérési kérelmei az alatta elhelyezkedő állomány blokkjaira fordítódnak le. Mivel az eszközt fel tudjuk csatolni a fájlrendszerek másolatait lényegében ugyanúgy érhetjük el mintha valódi lemezes eszközök lennének. Amennyiben a hálózatról töltöttük le a ***Knoppixot***, akkor minden bizonnyal van egy ***ISO9660*** lenyomatunk amit a ***loop*** segítségével fel tudunk csatolni, ha kíváncsiak vagyunk a tartalmára:

```
# mkdir /tmp/knoppix-cd
# mount -o loop -r \
$HOME/KNOPPIX_V3.3-2003-09-24-EN.iso /tmp/knoppix-cd
```

A `cloop` tömörített `loop` eszköz még egy lépéssel tovább megy. Ez a `loop` eszköz úgy modellezi az alkatrészt, hogy minden blokkot a `gzip`-pel tömörít, amit eléréskor átlátszó módon kibont. A ***/KNOPPIX/KNOPPIX*** pontosan ilyen másolat, amelyet rendszerindításkor csatolunk fel – ezzel a trükkel tud a ***Knoppix*** 2GB-ot elhelyezni egy 650MB-os CD-n.
Ha csak egyszerűen körbe szeretnénk nézni a fájlrendszeren, nem muszáj a rendszermagunkba fordítanunk a ***cloop*** képességet. Elegendő a ***cloop-utils*** csomagot feltennünk és az ***extract_compressed_fs*** segédeszközt használnunk. Az állomány elhelyezéséhez körülbelül 2GB szabad helyre lesz szükségünk a ***/var/tmp*** vagy valamely egyéb könyvtárban:

```
# mkdir /tmp/knoppix-cloop
# extract_compressed_fs \
   /tmp/knoppix-cd/KNOPPIX/KNOPPIX \
    >/var/tmp/KNOPPIX-cloop
# mount -o loop /var/tmp/KNOPPIX-cloop \
   /tmp/knoppix-cloop
# find /tmp/knoppix-cloop -print
```

Mindent a szemnek, semmit a kéznek – az ***ISO9660*** fájlrendszer csak olvasható. A terjesztés módosításához először is mindkét fájlrendszer szerkezetet egy hagyományos könyvtárba kell másolnunk:

```
# mkdir $HOME/my-knoppix-tree  \
    $HOME/my-knoppix-cd-tree
# tar -C /tmp/knoppix-cloop -cf - . | \
    tar -C $HOME/my-knoppix-tree -xvpf -
# tar -C /tmp/knoppix-cd -cf - . | \
    tar -C $HOME/my-knoppix-cd-tree -xvpf -
# umount /tmp/knoppix-cd /tmp/knoppix-cloop
```

Most már nekiláthatunk megvalósítani szívünk vágyát. A legkényelmesebb módszer, ha a ***Knoppix*** belső fában a `chroot` paranccsal megváltoztatjuk a gyökeret:

```
# mount -t proc none $HOME/my-knoppix-tree/proc
# cp /etc/resolv.conf \
$HOME/my-knoppix-tree/etc/resolv.conf
# chroot $HOME/my-knoppix-tree /bin/sh
```

Ettől kezdve a szokásos ***Debian*** csomagkezelő parancsok segítségével (`dpkg`, `apt-get` és a többiek) telepíthetünk és törölhetünk tetszés szerint. Miután elkészültünk, lépjünk ki

a chroot-ból és csatoljuk le a *proc*-ot (hacsak nem akarjuk halhatatlanná tenni a fejlesztői rendszerünk folyamatlistáját a CD-n). A create_compressed_tree és mkisofs parancsokkal készítsük el a külső és belső fát:

```
# mkisofs -L -R -l -V "KNOPPIX ISO9660" -v \
-allow-multidot $HOME/my-knoppix-tree | \
create_compressed_fs - 65536 > \
$HOME/my-knoppix-cd/KNOPPIX/KNOPPIX
# mkisofs -l -r -J -V "KNOPPIX with local stuff" \
-hide-rr-moved -v -b KNOPPIX/boot-en.img \
-c KNOPPIX/boot.cat -o knoppix.iso \
$HOME/my-knoppix-cd
```

Végül, írjuk ki a *knoppix.iso*-t egy *CD-ROM*-ra és indíthatjuk a rendszert. Akár írás nélkül is kipróbálhatjuk *Bochs* vagy *VmWare* segítségével.

Következő lépés

Ez az egyszerű megközelítés azonban kezd nehézkessé válni amikor komolyabb vállalkozásba fogunk. Például, ha azt szeretnénk, hogy az X egy adott ablakkezelőt indítson de nem akarjuk, hogy *GNOME* vagy *KDE*-t használjon, kénytelenek leszünk belenyúlni a parancsfájlokba. Ez persze nem nehéz feladat, viszont azt jelenti, hogy lényegében *Knoppix* leágazást készítünk. Amikor egy új *Knoppix* verzió jön ki, mindent újra meg kell csinálnunk. Továbbá, ha üzleti céllal szeretnénk eladni a *Knoppix* alalpú CD-nket, meg kell felelnünk valamennyi terjesztésre szánt program engedélyének, ami feltételezi hogy pontosan tudjuk mi van a lemezen. Az általam vizsgált *Knoppix* verzió tartalmazott néhány olyan állományt amelyek nem Debian csomagokból származtak, sőt, némelyik még csak nem is volt szabad program.
Nem közelíthetnénk meg a problémát más oldalról? Szerencsére megtehetjük. Hála a *Progeny* kezdeményezésnek, amely telepítőjét ajándékozta a *Debian Projektnek*, *Klaus Knoppernek* a *Knoppix* szerzőjének és a *cloop* meghajtó alkotójának, és más Debian fejlesztőknek akik kódját a fő *Debian* tárhelybe felvették, ma már nulláról össze tudunk állítani egy teljes live CD rendszert kizárólag *Debian* csomagok segítségével. A cikk további részében ezzel fogunk foglalkozni.

Letöltések

A cikkünkben szereplő parancsfájlok és állományok tarlabdája a következő címről tölthető le: *ftp.linux.org.uk/~dan/livecd*. Helyszűke miatt cikkünkben a legtöbb kódot nem mutatjuk be. Többnyire *Makefile* által vezérelt, pár héjprogrammal és néhány egyszerű *Perl* kóddal megtűzdelve, úgy gondolom elég jól követhető. Kicsit nehezebb dolga van annak aki nem *Debiant* használ. Ha sikerül más terjesztés alatt is működésre bírni a dolgot, kérem küldjék el a foltot.
A *debootstrap* program az további munkához szükséges *Debian* alaprendszert biztosítja. Ha megadjuk a *Debian* kiadás nevét és a csomagtükör *URL*-t, a *debootstrap* a kiválasztott alkönyvtárba tölti le és telepíti az alaprendszer. Ez igen rugalmas megoldás. Beléphetünk *chroottal*, használhatjuk *UML* gyökérként, avagy, ha a kiválasztott alkönyvtár külön fájlrendszer volt, újraindíthatjuk a gépet és közvetlenül is használhatjuk. Akár CD-re is kiírhatjuk, és pontosan ezt akarjuk majd tenni. De addig még van egy két elvégzendő feladatunk.
A parancsfájljaink próbálgatása folyamán jó sok *debootstrap* és csomagtelepítésre számíthatunk. Mielőtt továbblépnénk, egy kis időt és sávszélességet takaríthatunk meg, ha feltesszünk valamilyen *proxy* csomagarchívumot (például az *apt-proxy*-t) egy kényelmesen elérhető gépre.

Csomagok felvétele

A *Makefile*-ban található fix_inner cél csomagokat ad az alaprendszerhez. Első lépésként lecseréljük a start-stop-daemon programot a */bin/true*-ra, nehogy az *telepítés utáni (post-installation) parancsfájlok* szolgáltatásokat indítsanak el a *chroot* környezetünkben. Ez után többször *chroot*-olunk a rendszerbe és futtatjuk az apt-get és dpkg parancsokat. Tesztelési és kísérletezési célokra a *run-chroot.pl* Perl parancsfájlt használhatjuk, amely rendszerindítást szimulál *chroot* környezetben. A legtöbb szolgáltatást nem indítja el, hiszen azok már elve futnak a gépen és ütközést okozhatnának, de az *SSH* kiszolgálót és az X indítószkripteket lefuttatja. Sokkal kényelmesebb mint mindig kiírni egy CD-t és újraindítani a gépet valahányszor ki akarjuk próbálni.

autologin

Nem sok értelme van bejelentkezésre kényszeríteni az embereket egy egyfelhasználós, bemutatóra szánt rendszeren. Úgyis meg kell mondani nekik a jelszót, és mivel a CD csak olvasható legfeljebb ideiglenesen tudják megváltoztatni azt. A *GDM* rendelkezik ugyan *autologin* képességgel, de mivel a méretet nem szeretnénk túlságosan megnövelni, jobb lenne elkerülni a rengeteg *GNOME* függőséget. Ehelyett egyszerűen a su paranccsal indítjuk az X-et nem-root felhasználóként, és lefuttatjuk az *.xsession* parancsfájlt, amely beindítja az *xterm*-et, az *Emacs*-ot és az alkalmazásainkat. Az *autologin-x* parancsfájl */etc/init.d.autologin-x* néven települ, és elkészíti a rendszerindítás folyamatába illesztéshez szükséges közvetett hivatkozásokat.
A parancsfájl a DISPLAY változó értéke alapján dönti el, hogy melyik X kiszolgálót szükséges futtatni. Ha már be van állítva, az *Xvnc*-t indítja az *XFree86* helyett. Ez a tesztelést segíti: amikor az *autologin-x*-et a *run-chroot.pl* futtatja egy *xterm* ablakban, hozzákapcsolódhatunk egy *VNC* ügyféllel és meggyőződhetünk róla, hogy valamennyi X alkalmazás megfelelően indul. Természetesen ha az CD-ROM-ról indítva szeretnénk futtatni az X-et, tudnunk kell, a felhasználó milyen videókártyával rendelkezik.

Alkatrészfelderítés

A *Linux* alkatrész-felderítési képességei igen sokat fejlődtek az utóbbi tíz évben, amit az alkatrész-technológiák fejlődése is elősegített. Egy mai *PCI* vagy *USB* alkatrészt sokkal könnyebb megbízhatóan és biztonságosan azonosítani mint egy korabeli *ISA* csatolót.
A legtöbb *Linux* terjesztésben találunk valamilyen megoldást, amely végigvizsgálja a rendszer *PCI* és *USB* eszközeit és betölti a megfelelő modulokat. A *Knoppix* a *Kudzut* használja, amit eredetileg *Red Hat*-hez írtak, az eredeti *Debian*

pedig a `discover` parancsot alkalmazza. A két megoldás alkatrész lefedettsége közel azonos; tekintve, hogy mind a kettő nyílt forrású, nyugodtan másolhatnak egymás alkatrész adatbázisából. A ***Debian*** X kiszolgáló csomagja eleve a `discover` segítségével állapítja meg az X beállítási kérdések alapértékét, így inkább ennél maradunk.

debconf

Mit kezdünk a felderített alkatrészekkel? A ***Debian*** csomagoknak kézzel szerkeszthető beállításállományuk van, de általában telepítés-utáni parancsfájlokkal interaktív módon hozzák létre azok kezdeti verzióját telepítéskor. Amikor megoldható, mint például az X és hálózati beállítások esetében, a parancsfájlok lefuttatják az alkatrész kereső eszközöket.
A gond csak az, hogy mi jelenleg ***chroot*** környezetben telepítünk a gazdagépen, és a gazda alkatrészeinek felderítése nem sokat segít a célgépen. Ezért a ***debconf*** adatbázisát valami írható helyre kell tennünk, így indításkor a `debconf-communicate` segítségével törölhetjük a csomagok beállításait, majd a ***.config*** parancsfájlok futtatásával elhitetjük velük, hogy első ízben állítjuk be őket. Ez alaposabb megközelítés mint a `dpkg-reconfigure` futtatása, ami néha olyanokat kérdez mint: „Biztos, hogy újra be kívánja állítani a csomagot?”. Ez megzavarhatja a végfelhasználót aki természetesen még semmit sem állított be.
A `debconf-communicate` kézikönyvoldalain és a tarlabda ***target/etc/init.d/configure-xserver*** állományában további részleteket találunk.

Maradandó tároló: Hotplug

A CD-ROM csak olvasható, a ramdisk megsemmisül amint a gépet lekapcsoljuk. Az emberek viszont el szeretnék menteni az állományaikat, illetve el akarják érni a meglévő merevlemezeiken és cserélhető tárolóikon (például ***USB*** kulcson vagy ***ZIP*** meghajtón) őrzött állományokat. Akárcsak az előbb, a munka legnehezebb részét már elvégezték helyettünk; most a ***hotplug*** és az ***autofs*** lesz a megmentőnk.
A ***hotplug*** az új eszközök behelyezését és eltávolítását figyeli. Amikor új ***USB*** tárolóeszközt vesz észre, betölti a megfelelő modulokat és emulált ***SCSI*** gazdát hoz létre. Továbbra is nekünk kell tudnunk azonban, hogy milyen eszközök állnak a rendelkezésünkre és nekünk kell kézzel felcsatolni őket, itt jön a képbe az ***autofs***.
Az ***autofs*** igény szerint fel- és lecsatolja a fájlrendszereket. A `map` program használatával egy ***Perl*** parancsfájlt futtatnunk valahányszor a felhasználó lekérdezi a ***/media/list*** könyvtárat; itt létrehoz egy könyvtárat, valamint elkészíti a csatlakoztatott eszközök nevének megfelelő közvetett hivatkozásokat. A hivatkozások ***autofs*** csatlakoztatási pontokra mutatnak melyeken keresztül elérhetjük a fájlrendszert.
Vizsgáljuk meg a ***target/etc/auto.master*** és ***target/usr/local/sbin/autofs-device-list*** állományokat a tarlabdában.

A rendszermag

Lényegében ugyanazt a rendszermag beállítást használjuk mint a ***Knoppix*** (vizsgáljuk meg egy futó ***Knoppix*** rendszer ***/usr/src/linux/.config*** állományát, vagy a tarlabdánk ***kernel-config*** állományát), de eltávolítunk belőle néhány számunkra nyilvánvalóan felesleges dolgot, például a ***ZISOFS***-t.

A szabványos ***Debian*** `make-kpkg` eszközcsomagok felépítik és telepítik a rendszermagot. A gazdagépen található Debian függőségről van szó (szükségünk lesz a ***cloop-src*** csomagra), és mivel valószínűleg ez az egyetlen ilyen nemegyértelmű függőség, a későbbi verziókban esetleg érdemes lehet áthozni a ***chroot***-ba.

A fájlrendszer

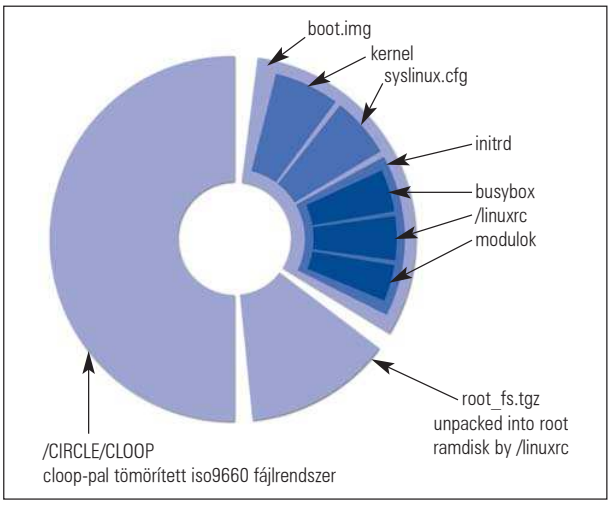
A legtöbb ***UNIX*** fájlrendszer szívesen elindul csak olvasható módban is, ám nekünk néhol szükségünk van fájlírásra. Például az X kiszolgáló beállításállományait indításkor a használt alkatrészeknek megfelelően át kell alakítanunk, a ***debconf*** adatbázist frissítenünk kellene illetve vannak különféle napló és zár-állományaink is.
A ***tmpfs*** fájlrendszer segítségével ***RAM***-alapú fájlrendszert hozunk létre. A rendszer ezt a memórialemezt fogja használni gyökérként és a /ro alkönyvtárban várja a CD lemez felcsatolását. A csak olvasható könyvtárakhoz közvetett hivatkozásokat hozunk létre, például a ***/usr***-t a ***/ro/usr***-hez rendeljük.
A csak olvasható könyvtárakról listát készítünk, és kétszeresen ellenőrizzük. Először egy tarlabdát készítünk a rendszerről amely nem tartalmazza ezeket a könyvtárakat, helyükön a megfelelő közvetett hivatkozásokkal. Ezt a tarlabdát aztán a futó rendszer gyökerébe másoljuk. Másodszor, amikor a ISO9660 képmást ***cloop***-tömörítéssel kiírjuk, ezt a könyvtárlistát fogjuk csatolni.

initrd

Mielőtt a rendszer beindulna két igen fontos dolgot kell elintéznünk. Először is fel kell csatolnunk a ***cloop*** másolatot, szükség szerint be kell töltenünk a ***CD-ROM*** modulokat, majd meg kell találnunk és fel kell csatolnunk a CD-t. Ezek után telepítjük a ***cloop*** eszközt majd felcsatoljuk rá a belső fájlrendszert. Másodszor elkészítjük a gyökér fájlrendszerhez a memórialemezt és rámásoljuk a ***root_fs.tgz*** tartalmát a CD-ről.
Az ***initrd (initial ramdisk)*** támogatás segítségével készítünk egy mini gyökér rendszert amit a rendszermag felcsatol és lefuttat mielőtt a valódi `init` elindulna. Ez gzippelt fájlrendszer lesz. Amikor ***initrd*** támogatással rendelkező rendszermagot indítunk a `initrd=filename` paraméterrel, a megadott fájlnév tartalmát fogja betölteni és memórialemezt készít belőle. Ez után memórialemezen elindítja a ***/linuxrc*** állományt.
Amikor a ***linuxrc*** befejeződött, a ***pivot_root*** hívást használja a valódi gyökér fájlrendszerbe váltáshoz (amely addig a /ramdisk volt) majd lefuttatja a valódi init-et.
A rendszermag és az ***initrd***, az indítómásolat valamennyi állományát beleszámolva, elég kicsi kell legyen, hogy beleférjen 1.44MB memóriában. Ez nem valami sok hely, hiszen a ***GNU*** önmagában 1,200K, úgyhogy kreatívnak kell lennünk.

dietlibc, Busybox

Aki soha még csak nem is vágyott ***Linuxos PDA***-ra vagy autóba építhető MP3 lejátszóra, most az is igazán hálás lehet a beágyazott ***Linux*** programozóinak.
A ***Busybox*** és ***dietlibc*** segítségével fogjuk ugyanis átjuttatni a tevét tű fokán. A ***busybox*** apró héjprogram amit fordí-



1. ábra Doboz a dobozban: a kész CD fájlrendszer másolatba ágyazott fájlrendszer másolatban megbúvó fájlrendszer másolatokat tartalmaz

táskor beállíthatunk úgy, hogy beépítettként tartalmazzon sok általános eszközt, a ***dietlibc*** pedig egy kis méretre optimalizált C könyvtár változat. Örömmel vehetjük észre, hogy a ***busybox*** mindent tud, amire az ***initrd***-ben szükségünk lehet, valamint hogy statikusan fordítva a ***dietlibc***-vel mindössze 100K helyet foglal. Összehasonlításképpen ugyanezekkel a ***Busybox*** kapcsolókkal statikusan fordítva a ***glibc***-vel 500K méretű végrehajtható állományt kapunk.
A ***Busybox*** bővítményeit a (tarlabdában található) ***Config.h*** állomány `#define` kulcsszavaival kapcsolhatjuk be. A kikapcsolt értékek talán önkényesnek tűnhetnek, de amikor könyvtárlista kérésére használhatjuk az `echo *` és `tar cvf /dev/null` parancsokat is, az `ls` igazán luxusnak tűnik.
Az ***initrd***-t a `genext2fs` programmal készítjük, így nem lesz szükségünk ***loopback*** felcsatolásra. A program egy könyvtárszerkezetből készít ***ext2*** fájlrendszert, amit ***gzip***-el tömörítünk és a rendszerindító lemezmásolatba mozgatunk (*1.ábra*).

Rendszerindítás

A CD-ROM-ról történő rendszerindítás szabványa ***El Torito*** néven vált ismertté amit eredetileg a ***Phoenix BIOS*** készítői hoztak létre. Az ***El Torito*** egy vagy több lemezképmás létrehozását teszi lehetővé a CD lemezen. Rendszerindításkor a ***BIOS*** felderíti ezeket, szimulált lemezt hoz belőlük létre és erről indítja a rendszert. A másolatok hajlékony lemez (1.44MB vagy 2.88MB) illetve merevlemez másolatok lehetnek. Ezen kívül létezik nem-emulációs mód is, ahol a ***BIOS*** egy megadott állományból tölt be szektorokat és emulált lemez kialakítása nélkül hajtja végre azokat.
Természetesen itt is van buktató: Az ***El Torito*** rendszert ***BIOS*** írók készítették. A laptoppal vagy egyéb érdekes alkatrésszel rendelkező ***Linux*** tulajdonosok jól tudják, hogy a ***BIOS***-ok nem éppen a leginkább hibamentes programok ezen a földön. Nem ok nélkül feltételezhetjük, hogy bizonyos gyártók nyugodt szívvel hagyják figyelmen kívül a hivatalos rendszerleírásokat mindaddig, míg szerkezetük az éppen aktuális ***Windows*** rendszer alatt valahogy elindul.
Így aztán akármilyen fájdalmas is a méretbeli korlát, a maximális hordozhatóság érdekében a ***Knoppix*** által kijárt utat fogjuk követni és egyetlen 1.44MB hajlékonylemezes másolatot alkalmazunk.

boot.img

Mit tegyünk ebbe az 1.44MB-ba? Indíthatunk nyers Linux rendszermagot, esetleg használhatjuk a megszokott ***LILO*** vagy ***Grub Linux*** rendszerindítót. ***H Peter Anvin SYSLINUX*** eszköze azonban használhatóság terén mindkét lehetőséget könnyedén veri. A ***SYSLINUX MS-DOS*** fájlrendszert használó indítólemezeket készít, így a hajlékonylemez másolatot a felhasználói ***mtools*** segítségével is létrehozhatjuk.
A lemeznek a rendszermag ***vmlinuz*** állományra, a ***syslinux.cfg***, csatolt segítségnyújtó állományokra és az ***initrd*** másolatra van csak szüksége. Amint készen vagyunk futtathatjuk rajta a ***SYSLINUX***-ot.
Már csak az maradt hátra, hogy elkészítsük és kiírjuk a fájlrendszerünket, amint azt korábban is tettük. A belső fájlrendszer a `$(SCRATCH)/CLOOP`-ban található. A külső fájlrendszerünk a ***boot.img*** és ***root_fs.tgz*** állományokat tartalmazza majd. Ezt kiírjuk a CD-re (egy-két CD-RW lemez sem árthat) majd elindítjuk róla a gépet. Hacsak nincs nagy balszerencsénk, működni fog.

Végszó

Számomra, régi ***Linux*** felhasználó számára, aki szokásos telepítést már évek óta nem csinált, igazán lenyűgöző, hogy milyen sokat fejlődött az automatikus alkatrész felismerés és támogatás. És ahogy telik az idő, biztos vagyok benne, hogy még ennél is csak jobb lesz.
Merre lehet továbblépni? Az ***automount*** támogatáson kell kicsit csiszolni; esetleg kipróbálhatunk valami mást, például a ***Volumatic***-ot. A többi leginkább a készülő termékünktől függ. Azonban minden parancsfájl szabad program, és érdeklődve várom a visszajelzéseket.

Linux Journal 2005. április, 132. szám

Daniel Barlow független szakértő az Angliai Oxfordban, ahol Linux és Common Lisp fordítókkal dolgozik. Szabadidejében, nagyon szeret elektromos gitáron rosszul játszani, ami nagy szerencse, ugyanis csak így tud. Megjegyzéseket a dan@metacircles.com címen várja.

KAPCSOLÓDÓ CÍMEK

debconf:
➲ www.kitenet.net/programs/debconf

El Torito:
➲ www.phoenix.com/resources/specs-cdrom.pdf

Hotplug:
➲ linux-hotplug.sf.net

SYSLINUX:
➲ syslinux.zytor.com

VNC:
➲ www.realvnc.com