A Bochs emulátor

A Bochs egy olyan speciális program, amely az indítása után egy virtuális gépet hoz létre, melyet ugyanúgy kezelhetünk, mint ha valós számítógép lenne. Ez azért jó, mert így a más operációs rendszerre írt programokat saját operációs rendszerük alatt futtathatjuk.

Mi több, még virtuális gépünk processzorának típusát is meghatározhatjuk. Ez lehet ***386***, ***486***, ***Pentium***, ***Pentium Pro***, vagy akár ***AMD64*** típusú is, természetesen mindazon speciális utasításkészlettel, amelyek ezekben a processzorcsaládokban megtalálhatók. A perifériák kezelése is nagyon jó, mivel a gazda operációs rendszerek szolgáltatásait használja.
Ez azt jelenti, hogy minden hardvereszközt emulál és a hozzá intézett feladatokat átadja az operációs rendszer megfelelő moduljának. Mivel minden programutasítást és minden hardverhozzáférést emulál, ezért a ***Bochs*** sebessége nem túl nagy. Kellő türelemre van szükség, hogy a feladatok végrehajtását kivárjuk. Emennyiben nagyobb teljesítményre van szükségünk, úgy más emulátor programot célszerű használnunk, mint amilyen például a ***VMware***.

Telepítés

A ***Bochs*** telepítése nem nagy ördöngösség. ***Linux*** gazda operációs rendszert használva el kell döntenünk, hogy előre lefordított binárisból, vagy forrásból szeretnénk-e telepíteni. Mindegyiknek van előnye és hátránya egyaránt. A bináris állománnyal sok dolgunk nincs, egyszerűen telepíteni kell, azonban meg kell elégednünk a fordítást végző által meghatározott funkciókkal.
A forrásból telepítés több munkával jár, de cserébe személyre szabhatjuk a ***Bochs***-ot. Nézzük először a bináris állomány telepítését.
Töltsük le a ➲ ***bochs.sourceforge.net*** oldalról a legfrissebb verziót, ami a cikk írásának időpontjában a 2.2.1. A fájl ***bochs-verziószám.rpm*** nevet viseli, a cikk írásakor a legfrissebb verzió a 2.2.1, vagyis a telepítő készlete ***bochs-2.2.1.rpm*** néven érhetjük el. A telepítéshez rendszergazdai jogosultság szükséges.

```
root# rpm -iv bochs-2.2.1-1.i586.rpm
```

Amennyiben nem kapunk hibaüzenetet, végeztünk is a telepítéssel. Ha valamilyen hibába ütköznénk, olvassuk el a hibaüzenetet, amire az interneten rákeresvevalószínűleg megtaláljuk a megoldást. Sokszor a nem megfelelő függőségek miatt hiúsul meg a telepítés, ilyenkor lehetőségünk van forrásból új bináris állományt készíteni. Ha a telepítés rendben lezajlott, a rendszerünkben megjelent néhány új parancs, a `bochs`, ami maga a virtuális gép, a `bochs-dlx`, ami egy ***DLX Linux*** demó a próbálgatáshoz, a `bximage`, amivel lehetőség van a lemezek állományait létrehozni és a `bxcommit` ami speciális lemezképfájlok elkészítésére használható.
Rögtön próbáljuk is ki a virtuális gépünket, amihez egy terminálban adjuk ki a `bochs-dlx` parancsot. Megjelenik néhány sor, amely az alapvető ellenőrzéseket hajt végre és közli, hogy létrehozza a felhasználó saját könyvtárában a ***.bochsrc*** fájlt, valamint egy 10 MB nagyságú lemez képfájlt.

```
------------------------------------------------
       DLX Linux Demo, for Bochs x86 Emulator
------------------------------------------------
Checking for bochs binary...ok
Checking for DLX linux directory...ok
Checking for /usr/bin/gzip...ok
Checking for /root/.bochsdlx directory...
------------------------------------------------
To run the DLX Linux demo, I need to create
➥a directory called
/home/user/.bochsdlx, and copy some
➥configuration files
and a 10 megabyte disk image into the directory.
------------------------------------------------
Is that okay? [y/n]
```

Miután a fenti sorokat elfogadtuk, elindul a ***DLX Linux*** demója, és be tudunk lépni root-ként (*1. ábra*).
Amennyiben forrásból szeretnénk telepíteni a ***Bochs***-ot, akkor első lépésként töltsük le a ***bochs-2.2.1.tar.gz*** állományt és csomagoljuk ki a

```
tar -xzvf bochs-2.2.1.tar.gz
```

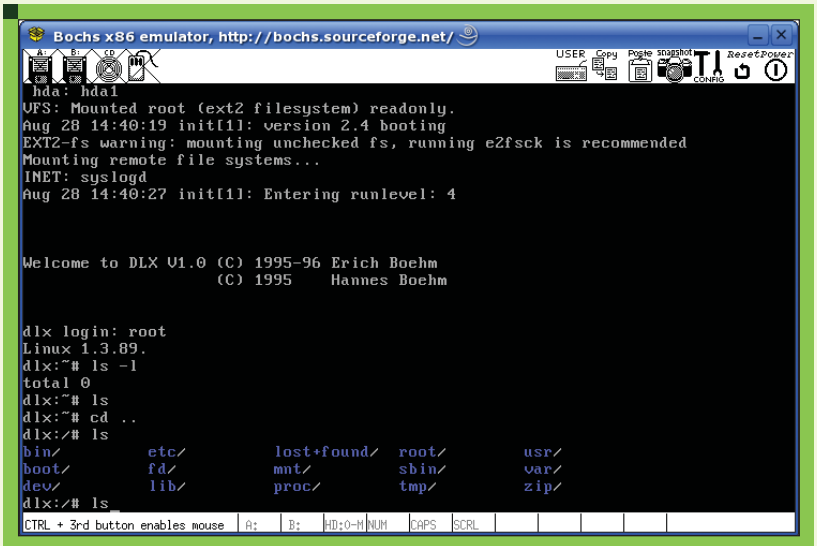
paranccsal. Ha belépünk a létrejött ***bochs-2.2.1*** könyvtárba, első lépésként adjuk ki a

```
./configure --help
```

parancsot és tanulmányozzuk át a hosszú listát, amiből megtudhatjuk, hogy milyen fordítási opciókat állíthatunk be. Miután kiválasztottuk a szükséges opciókat, elindíthatjuk a

```
./configure
```

parancsfájlt. Annak érdekében, hogy megkönnyítsék a felhasználók dolgát, elkészítettek néhány parancsfájlt, amelyek az egyes operációs rendszerekre vannak optimalizálva. Ezeket *.conf.operációs_rendszer* néven találjuk meg. A *Linuxhoz* a *.conf.linux* fájlt kell használni. Amennyiben más opciókat is szeretnénk használni, nyugodtan módosítsuk ezt a fájlt, a későbbiekben is felhasználhatjuk. A listában egy javaslom, hogy módosítsuk a processzor típust (`enabled-cpu-level`) 5-re (*Pentium* osztályú), mivel ez sokkal stabilabb, mint a *Pentium Pro* támogatás.



■ *1. ábra* Indul a DLX Linux


```
sh ./config.linux
```

Ennek hatására konfigurálásra kerül a *Bochs*, amit le kell fordítanunk a `make` paranccsal. Miután hiba nélkül ez is befejeződött, telepítsük fel a programot a `make install` paranccsal. Ne feledjük, hogy ehhez a lépéshez rendszergazdai jogokra lesz szükségünk!
Bármelyik módszert is használjuk, alapértelmezés szerint a *Bochs* a */usr/bin* könyvtárba települ, az osztott fájlokat a */usr/share/bochs* könyvtárban tárolja és a beépülő modulokat a */usr/lib/bochs/plugins* könyvtárba helyezi el.

A Bochs használata

Valószínűleg nem azért telepítettük fel a *Bochs*-ot, hogy egy egyszerű *DLX Linux* demót futtassunk rajta. A rendszer használatához szükségünk lesz magára a *Bochs*-ra, egy *BIOS* képfájlra, egy *VGA BIOS* képfájlra és valamilyen háttértároló képfájljára, vagy esetleg a fizikai eszközre, valamint egy, a beállításokat tartalmazó fájlra.
Ez utóbbi a *.bochsrc* nevet viseli és célszerűen a saját könyvtárunkban helyezhetjük el. A *BIOS* és a *VGA BIOS* a *Bochs* telepítőeszközzel érkezik, de az interneten is találhatunk mások által készített fájlokat.
A következőkben tekintsük át röviden a beállítófájl egyes sorainak jelentését.

```
romimage: file=$BXSHARE/BIOS-bochs-latest,
➥address=0xf0000
```

Ez megadja a *BIOS* képfájlnak a helyét, valamint azt, hogy ezt a *0000h* címre kell betölteni, ami a *RESET* állapotnak megfelelő indulási cím. A *$BXSHARE* változó a */usr/share/bochs* könyvtárra mutat.

```
vgaromimage: file=$BXSHARE/VGABIOS-lgpl-latest
```

A *VGA BIOS* fájl elérési útvonala és neve.

```
vga: extension=vbe
```

VGA kiegészítést kapcsolhatunk be vele, ami lehetővé teszi, hogy a telepített rendszerünket ne csak *640x480*-as felbontásban használhassuk, hanem az *SVGA* felbontások is elérhetők legyenek. Amennyiben ezt szeretnénk használni, vendég operációs rendszerhez be kell szereznünk egy *VBE* meghajtót.
Miután beállítottuk a *BIOS*-okat, meg kell adnunk, hogy mekkora memóriát használhat a virtuális gép. Ehhez a

```
megs: 128
```

sort kell a *.bochsrc*-be beszúrnunk, ami 128 megabájtban korlátozza a memóriahasználatot.
Természetesen a memória méretét a gépünk kiépítettsége alapján kell megállapítsuk. Ha túl kicsire vesszük, akkor nagyon lassú lesz az emulált rendszer, ha pedig túl nagyra, akkor lehet, hogy pazaroljuk a hasznos operatív tárat.
A paraméter maximális értéke `2048` lehet, de `1024` vagy efölötti értékekkel már meggyűlhet a *Bochs* baja, ezért óvatosan bánjunk vele.
Következhet a hajlékony- és merevlemezes meghajtók, valamint optikai tárolók meghatározása.

```
floppya: 1_44=/dev/fd0, status=inserted
floppyb: 1_44=b.img, status=inserted
```

Az A jelű meghajtó a `/dev/fd0` fizikai eszközhöz kapcsolódik, míg a *B* jelű a *b.img* fájlhoz. Ez utóbbi valamilyen hajlékony lemezről készített képfájl. Más lemezméretet is megadhatunk, de nem valószínű, hogy valakinek ennél kisebb méretű lemezekre lesz szüksége. A status kapcsolóval meghatározhatjuk, hogy a lemez behelyezett állapotban van, vagyis használatra kész. Ez a képfájloknál egyértelmű, de fizikai eszközöknél nem feltétlen.

A merevlemezes és az optikai meghajtók megadása előtt meg kell határoznunk, hogy a maximum négy lemezvezérlőből mennyit szeretnénk használni. Ehhez az alábbi sorokat szúrjuk be a .bochsrc fájlba:

```
ata0: enabled=1, ioaddr1=0x1f0, ioaddr2=0x3f0,
➥irq=14
ata1: enabled=1, ioaddr1=0x170, ioaddr2=0x370,
➥irq=15
ata2: enabled=1, ioaddr1=0x1e8, ioaddr2=0x3e0,
➥irq=11
ata3: enabled=1, ioaddr1=0x168, ioaddr2=0x360,
➥irq=9
```

Tiltani az egyes vezérlőket az `enabled = 0` értékkel lehet. Alaphelyzetben csak egy vezérlő aktív, a többi tiltott állapotú. Miután eldöntöttük, hogy mely vezérlőket szeretnénk használni, meg kell határozni, hogy milyen eszközök találhatók a vezérlőkön. Az ***ATA*** eszközökhöz ugyanazt a sort kell megfelelő paraméterekkel használnunk.

```
ata0-master: type=disk, path=/home/jimi/inst/
➥bochs-20050731/c.img, mode=flat cylinders=2031,
➥heads=16, spt=63
ata0-slave: type=cdrom, path=/home/jimi/inst/
➥bochs-20050731/win98.iso, status=inserted
```

Mint látható, a `type` határozza meg, hogy lemezről (`disk`) vagy ***CD-ROM***-ról (`cdrom`) van-e szó. Meg kell adnunk az elérési útvonalat, ami lehet egy képfájl, de lehet akár egy fizikai partíció vagy eszköz is. Ez utóbbival nem árt óvatosnak lenni, mert akár a fájlrendszer is megsérülhet egy esetleges ***Bochs*** összeomlás esetén. Javaslom, hogy a fizikai partíciókról hozzunk létre képfájlt, aminek pontos menetét a cikk későbbi részében természetesen ismertetni fogom.
A lemezeknél használható `mode` kapcsoló adja meg, hogy milyen típusú képfájlokkal dolgozunk.
A leggyakrabban a `flat` típusú fájlokat használjuk, ami egyszerű felépítést és fix méretet jelent, de használhatunk `concat` (több fájlból álló kép) és `growing` (igényeknek megfelelően növekvő méretű) típusokat is. Mint a fenti sorban látható, még a lemezegység geometriáját is megadhatjuk, amit a cilinder (`cylinders`), fej (`head`) és a szektorszámmal (`spt`) jellemeznek.
Miután beállítottuk a háttértárakat, meg kell határoznunk azt is, hogy mi legyen az indítási sorrend. Ehhez szúrjuk be a

```
boot: cdrom, disk, floppy
```

sort. A sorrend természetesen bármi lehet.
Amennyiben szeretnénk, ha a ***Bochs*** a műveleteit folyamatosan naplózza, akkor használhatjuk a

```
log: bochsout.txt
```

sort. ***Linux*** alatt a `/dev/tty` vagy akár a `/dev/null` eszközöket is használhatjuk.
A mai rendszerekben a legtöbb eszköz a ***PCI*** sínrendszert használja a kommunikációhoz, ezért ezt a ***Bochs*** is támogatja. A PCI eszközök használatához szükségünk van a

```
i440fxsupport: enabled=1, slot1=pcivga,
➥slot2=ne2k
```

sorra, ahol a `slot` kapcsolókkal megadhatjuk, hogy melyik csatolóra milyen eszköz kapcsolódik.
Összesen 5 csatlakozó áll rendelkezésünkre, amelyekbe `ne2k` (hálózati kártya), `pcivga` (***VGA*** kártya ***PCI*** csatolóval), `pcidev` (***PCI*** eszközazonosítás és erőforrás kiosztás, de még nagyon kezdeti stádiumban van) és `pcipnic` (hálózati kártya ***PCI*** sínnel) eszközöket „csatlakoztathatunk”.
Előfordulhat, hogy szükségünk van soros portokra, ehhez a

```
com1: enabled=1, mode=mouse
```

sort helyezzük el a ***.bochsrc***-ben. A `mode` kapcsoló értéke `mouse` (szabvány soros egér), `null` (üres be- és kimenet), fájlnév (például `dev=/dev/ttyp9`), valamint `raw` (tényleges hardverport, egyenlőre fejlesztési fázisban van) lehet.
Egy modernebb soros kommunikáció, az ***USB*** támogatás is fontos lehet. Ehhez a

```
usb1: enabled=1, ioaddr=0xFF80, port1=mouse,
➥port2=keypad
```

sort kell használnunk. Ez egy ***i440FX PCI*** lapkakészletet emulál, amelynek része az ***USB*** vezérlő. Mivel minden vezérlő két porttal rendelkezik, ezért a port kapcsolóval megadhatjuk, hogy melyikre milyen eszköz csatlakozik. A használatához szükség van a ***PCI*** támogatás bekapcsolására.
Persze a párhuzamos portot is hasonló módon kezelhetjük, amihez a

```
parport1: enabled=1, fájl="dev/lp0"
```

sort kell használnunk. Ezen portokat az `enabled=0` kapcsolóval lehet tiltani.
Hasonlóan az egér engedélyezéséhez szúrjuk be a

```
mouse: enabled=1, type=imps2
```

sort, míg a tiltáshoz ugyanezt, csak `0` értékkel. A `type` kapcsoló mondja meg az egér típusát, ami `serial` (soros), `serial wheel` (soros egér görgővel), `ps2` (***PS/2***), `imps2` (másik fajta ***PS/2***), vagy `usb` (***USB*** egér) lehet. A soros egér használatához szükség van egy egér típusú (`type=mouse`) soros, míg az ***USB***-s egérnél egy ***USB*** port beállítására.
Amennyiben hangot is szeretnénk a virtuális gépen futó alkalmazásokból kicsiholni, szükség lehet a ***Sound Blaster 16*** hangkártya támogatására. Ehhez az alábbi sort kell beszúrnunk:

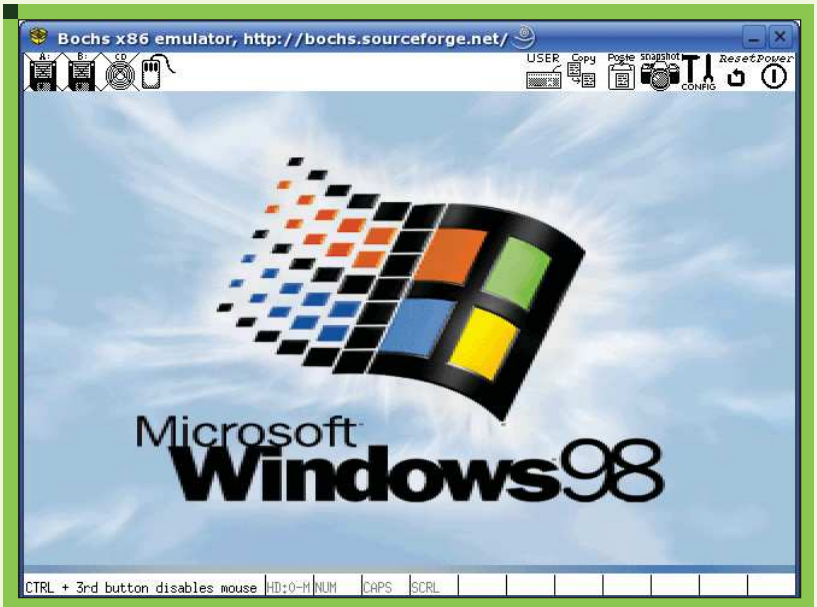
```
sb16: midimode=1, midi=/dev/midi00, wavemode=1,
➥wave=/dev/dsp, loglevel=2, log=sb16.log,
➥dmatimer=600000
```

A `midimode` kapcsoló megadja, hogy milyen módon használjuk az eszközt. Ha értéke `0`, akkor nincs adatkimenet, míg az `1` adatkimenetként használja az eszközt. A `wavemode` hasonló célokat használ, hasonló beállítási lehetőségekkel.
A log kapcsoló határozza meg, hogy a műveleteket a ***Bochs*** hova naplózza. Ehhez szorosan kapcsolódik a `loglevel`

kapcsoló, ami a naplózási eseményeket határozza meg. Értéke leggyakrabban 2 (súlyos hibákat naplóz), vagy 3 (összes hibát naplózza), de 0 és 5 között bármilyen egész szám lehet, a magasabb szám részletesebb naplózást jelent. A *DMA* ciklusok beállítását teszi lehetővé a `dmatimer` kapcsoló.
Bizonyos operációs rendszereknél szükség lehet az emulált gép sebességének a megadására. Ehhez használjuk az

```
ips: 15000000
```

sort, ahol a szám a másodpercenként végrehajtott utasítások számát adja meg. Értékét gyakorlatban tudjuk legjobban meghatározni, amit megkönnyíthet, ha a fordításnál használjuk a `--enable-show-ips` opciót.
Amennyiben szeretnénk hálózatot is használni a vendég operációs rendszerünkben, akkor definiálnunk kell legalább egy hálózati csatolót. Ehhez a



■ *2. ábra* Indul a Windows 98

```
ne2k: ioaddr=0x240, irq=9, mac=b0:c4:20:00:00:00,
➥ethmod=linux, ethdev=eth0
```

sort használhatjuk *Linux* alatt. Az `ioaddr` kapcsoló a perifériacímet, az `irq` a megszakítási vonal számát adja meg. A `mac` kapcsoló a kártya fizikai, adatkapcsolati rétegbeli címét határozza meg, az `ethmod` kapcsolóval állítjuk be az operációs rendszer típusát és végül az `ethdev` kapcsolóval pedig az eszköz nevét. *Windows* alatt ugyanez a sor az alábbiak szerint módosulna:

```
ne2k: ioaddr=0x240, irq=9, mac=b0:c4:20:00:00:01,
➥ethmod=win32, ethdev=KÁRTYANÉV
```

Alapértelmezésben a *Bochs AT* típusú, angol kiosztású billentyűzetet tesz elérhetővé. Ez sok esetben nem megfelelő ezért szükség lehet a ezeknek a megváltoztatására. A

```
keyboard_type: mf
```

megfelelő a legtöbb rendszer számára, de ha nem működne megfelelően, próbálkozhatunk a

```
keyboard_type: at
```

opcióval is. A billentyűzetkiosztás már egy kicsit bonyolultabb, mivel a fizikai gombkiosztáshoz egy más kódkészletet kell párosítanunk. Erre szolgál a

```
keyboard_mapping: enabled=1,
➥map=elérési_útvonal/billenytűzetkiosztás.map
```

sor. Mint látható, a kiosztást egy fájl tárolja, amelyet megpróbálunk begyűjteni az internetről (nekem sajnos nem sikerült megbízható fájlt találni), vagy létrehozunk mi magunk egyet. Ez utóbbi esetben jó szolgálatot tehet az *xkeycaps* nevű program, amelyet forrás formájában letölthetünk a ➲ *http://www.jwz.org/xkeycaps/* oldalról vagy ha segítségül hívjuk a *Google* keresőt, hamar ráakadhatunk a bináris változatra is.
Miután létrehoztuk a *.bochsrc* fájlt, már csak a lemezképet kell létrehoznunk, amit többféleképpen megtehetünk.

Merevlemez képfájl létrehozása

Ebben az esetben a *Bochs*-szal szállított bximage programra lesz szükségünk. A parancs kiadása után első kérdése a programnak, hogy hajlékony- vagy merevlemez fájlt kívánunk létrehozni (`Please type hd or fd. [hd]`), alapértelmezés értéket a szögletes zárójelek között láthatjuk.
A következő kérdés, hogy milyen típusú legyen a képfájl (`Please type flat, sparse or growing. [flat]`), itt maradhatunk a `flat` típusnál, vagy ha nem tudjuk a méretét a fájlnak megtippelni, akkor válasszuk a `growing` típust.
A harmadik kérdés a partíció méretére vonatkozik (`Enter the hard disk size in megabytes, between 1 and 32255`), ami 1 és 32255 MB közötti lehet. Miután ezzel is megvagyunk, kapunk egy összesítést arról, hogy milyen geometria paraméterekkel jön létre a fájl.
Már csak a fájl nevét kell megadnunk (`What should I name the image?`).
Ez egy nyers, formázatlan partíció, amit majd célszerűen az operációs rendszer alatt meg kell formáznunk.
Akkor is hasonló módon kell eljárnunk, ha egy meglévő operációs rendszer partícióját szeretnénk használni, mint lemez képfájl. Első lépésben elkészítünk egy, a fizikai partíciónak megfelelő (nem fontos, de jobb, ha ugyanakkora) lemez képfájlt a `bximage` programmal. Ezt követően a elindítjuk a másik operációs rendszert, kiürítjük a *TEMP* könyvtárakat, eltávolíttatjuk az ideiglenes állományokat és elvégeztetünk egy töredezettség-mentesítést is. Ez ezért fontos, mert így a fájlok folytonosan fognak elhelyezkedni a kép-

■ *3. ábra* Virtuális hardvereszközök

fájlban, ami sebességnövekedést jelent. Igyekezzünk azokat az állományokat is áthelyezni egy másik partícióra, melyek nem képezik az operációs rendszer szerves részét, mivel feleslegesen növelik a képfájl méretét. Miután ezzel is végeztünk, újra ***Linux*** rendszerre váltsunk át és csatoljuk fel a partíciót egy ideiglenes könyvtárba. Például ***Windows*** esetén az alábbi parancsokat adjuk ki:

```
mkdir /windows  #létrehozzuk a /windows
➥könyvtárat
mkdir /windows/c  #azon belül a c könyvtárat
mount -t vfat /dev/hda1 /windows/c/ #felcsatoljuk
➥a hda lemez első partícióját, a fájlrendszer
➥itt vfat (FAT32), de lehet más is
```

Az ***mtools*** csomag segítségével most használhatóvá kell tennünk a nyers képfájlt. Az ***mtools*** olyan parancsoknak a gyűjteménye, amelyeket megszokhattunk a ***DOS*** alatt és mindegyik „m” betűvel kezdődik.
Hozzunk létre a saját könyvtárunkban egy ***.mtoolsrc*** fájlt, amelyben helyezzük el az alábbi sort:

```
drive c: file="/elérési_útvonal/c.img"
➥partition=1
```

Következő lépés, hogy particionálnunk kell a képfájlt. Ehhez az `mpartition` programot kell használnunk, aminek általános alakja:

```
mpartition -I -s szektorszam -t cillinderszám -h
➥fejek_száma c:
mpartition -cpv -s szektorszam -t cillinderszám
➥-h fejek_száma c:
```

Az első parancs beállítja a partíciós táblát és törli a létező partíciókat, míg a második parancs létrehozza az egész képfájlt elfoglaló partíciót. A szektorszámot, a cilinderszámot és a fejek számát a `bximage` kimenetéből tudhattuk meg, amikor létrehoztuk a merevlemez képfájlt.
Következő lépés a létező partíció átmásolása a képfájlba. Ehhez az alábbiakra van szükség:

```
mcopy -s /windows/c/* c:
```

Ezzel meg is vagyunk a képfájl létrehozásával, már csak be kell állítanunk a ***.bochsrc*** fájlban, hogy erről történjen meg a rendszer indulása.

Képfájl létrehozása CD-ROM-ról

Sok esetben előfordul, hogy a telepítőkészlet ***CD***, vagy ***DVD*** lemezen érkezik. Célszerű ebből egy képfájlt készíteni, mert tapasztalataim szerint a ***Bochs*** ezt jobban szereti, nem lehet lemezolvasási hiba, bár helyet mindenképpen igényel.
A képfájl elkészítéséhez használjuk a

```
dd if=/dev/cdrom of=win.iso
```

parancsot, aminek eredményeként a dd egy meglehetősen hosszú munkába kezd, aminek a végeredménye egy ***win.iso*** nevű telepítő képfájl. Ezt már megadhatjuk a ***.bochsrc***-ben a cikkben ismertetett módon.
Az interneten kutakodva, akár csak a ➲ ***bochs.sourceforge.net*** oldalt tekintve jó néhány operációs rendszer képfájlját megtalálhatjuk, ami sok munkától tud megkímélni minket. Nyugodtan próbálgassuk ezeket, hiszen a fizikai rendszerünkben semmiféle változtatásra nincs szükség, így biztonságosan próbálhatók az operációs rendszerek.
A ***Bochs*** indítását a `bochs` paranccsal intézhetjük el. Ennek eredményeként megjelenik egy menü, amelyből az 5-ös menüpontnak kell aktívnak lennie. Amennyiben a 2-tes lenne az, a ***Bochs*** nem találja a ***.bochsrc*** fájlt. Ilyen esetben ellenőrizzük, hogy abban a könyvtárban adtuk-e ki a ***bochs*** parancsot, amelyikben a ***.bochsrc*** található.
Miután elindul a ***Bochs***, megjelenik egy ablak, amelyben elindul a virtuális gép és ha mindent jól állítottunk be, a lemez képfájlokból betöltésre kerül az operációs rendszer (*2. ábra*).
Az ablak tetején egy fejléc található, amely segítségével a ***Bochs*** eszközeit használhatjuk. Az ikonok magukért beszélnek, amely hardvereszköz át van húzva, az jelenleg nem engedélyezett a virtuális gépben (*3. ábra*).
A hardvereszközök munkavégzést az ablak alján látható állapotsor megfelelő része mutatja. Ha használunk egeret és az engedélyezett is a ***Bochs***-ban, akkor amikor az ablak fölé mozgatjuk az egérkurzort, automatikusan átadja a vezérlést a vendég operációs rendszernek. Ahhoz, hogy visszakaphassuk az egeret, használjuk a ***CTRL*** és az egér harmadik gombját egyszerre. Ez tiltja a virtuális gépben az egeret, engedélyezni ugyanezzel a billentyűkombinációval lehet.
A fentiek alapján remélem mindenki be tudja állítani a ***Bochs***-ot és tesztelni tudja a különböző operációs rendszereket. Nem szabad elfeledkeznünk róla, hogy zárt forráskódú, kereskedelmi termékeket csak akkor használhatjuk az emulátorban is, ha rendelkezünk érvényes licenccel. A ***Bochs*** alapvető célokra remekül használható, elsősorban azoknak ajánlható, akik nem engedhetik meg maguknak a drága, nagy teljesítményű kereskedelmi virtuális gépeket (mint amilyen a ***VMware***), de mégis szükségük van más operációs rendszer futtatására.

Markó Imre (linux@akribisbt.hu)
Hardvermérnök és mérnöktanár végzettsége van. Saját cégében Linux rendszerek tervezésével és üzemeltetésével foglalkozik. Ezen kívül egy főiskolán oktat, elsősorban hardveres tantárgyakat.