Jedro Linuxa, HOWTO

Brian Ward, `bri@cs.uchicago.edu`
v1.0, 5. junij 1999

To je slovenski prevod datoteke `Kernel-HOWTO`, podrobnega vodnika nastavitev jedra operacijskega sistema Linux, prevajanja, nadgradnje in odpravljanja problemov za sisteme s procesorji ix86.

Kazalo

1 Uvod

Naj sploh berete ta dokument? No, poglejte, če imate katerega od naštetih simptomov:

- „Aaa! Tale paket wizzo-46.5.6 pravi, da potrebuje jedro izdaje vsaj 2.8.193, jaz pa imam še vedno izdajo 1.0.9¡'
- V enem novejših jeder je gonilnik za neko napravo, ki ga preprosto morate imeti.
- Nimate najmanjšega pojma, kako prevesti jedro.
- „Je stvar v datoteki `README` res vsa zgodba¿'
- Prišli ste, poskusili, ni delovalo.
- Potrebujete nekaj, kar bi dali ljudem, ki vas mučijo s prošnjami, da namesto njih namestite njihova jedra.

1.1 Najprej preberite tole! (In to res mislim!)

Nekateri primeri v tem spisu predpostavljajo, da imate GNU `tar`, `find` in `xargs`. Ti so kar standardni; to vam ne bi smelo povzročati preglavic. Predvidevam tudi, da poznate datotečno strukturo vašega sistema; če je ne, je pomembno, da imate pri roki izpisano kopijo izhoda ukaza `mount` med normalnim delovanjem sistema (ali izpis datoteke `/etc/fstab`, če ga znate brati). Te informacije so pomembne in se ne spreminjajo, razen, če razparcionirate svoj disk, dodate novega, ponovno namestite svoj sistem, ali kaj podobnega.

Zadnja stabilna različica jedra je bila v času pisanja tega spisa 2.2.9, torej se sklicevanja in primeri nanašajo na to izdajo. Čeprav poskušam narediti ta spis karseda neodvisen od različice, se jedro nenehno razvija, torej bo novejša različica neizogibno vsebovala nekaj sprememb. To vam ne bi smelo povzročati večjih težav, lahko pa vas malce zmede.

Obstajata dve različici izvorne kode jedra Linuxa; stabilna (angl. „production") in razvojna (angl. „development"). Stabilne izdaje imajo sodo malo število: 1.2.x je bila stabilna, 2.0.x prav tako, in tudi 2.2.x. Ta jedra se smatrajo za najbolj stabilne in razhroščene različice v času njihove izdaje. Razvojna jedra (2.1.x, 2.3.x itd.) so jedra, namenjena preizkušanju, za ljudi, ki bi radi preizkusili nova in morda zelo hroščata jedra. Bili ste opozorjeni!

1.2 O stilu

`Besedilo, ki je videti takole` je nekaj, kar se prikaže na vašem zaslonu, ime datoteke ali nekaj, kar lahko neposredno vtipkate, na primer ukaz ali izbire pri ukazu (če gledate tekstovno različico tega spisa, zgornje besedilo ne izgleda nič drugače). Ukazi in drugi vnosi so pogosto citirani (z „ “), v slovenskem prevodu takole: „`make config`“.

2 Pomembna vprašanja in njihovi odgovori

2.1 Kaj pravzaprav sploh počne jedro?

Jedro Unixa deluje kot posrednik med vašimi programi in strojno opremo. Najprej, dela (ali poskrbi za) upravljanje s pomnilnikom za vse tekoče programe (procese) in poskrbi, da vsi dobijo pošten (ali nepošten, če tako želite) delež ciklov procesorja. Poleg tega daje programom prijazen, precej prenosljiv, vmesnik za pogovor z vašo strojno opremo.

Gotovo ima jedro več dolžnosti kot samo ti dve, a najpomembneje je, da poznate ti, osnovni.

2.2 Zakaj bi si želeli nadgraditi svoje jedro?

Novejša jedra v splošnem ponujajo možnost za pogovor z več vrstami opreme (to se pravi, imajo več gonilnikov naprav), imajo morda boljše procesno upravljanje, lahko delujejo hitreje kot starejše različice, so bolj stabilna od starejših različic in odpravijo neumne hrošče v starejših različicah. Večina ljudi nadgradi jedra, ker želijo gonilnike naprav in popravke hroščev.

2.3 Katero strojno opremo podpirajo nova jedra?

Preberite `Hardware-HOWTO`. Kot alternativo lahko pogledate datoteko `config.in`, v Linuxovi izvorni kodi, lahko pa tudi izveste, ko poskusite „`make config`“. To vam pokaže vso opremo, ki je podprta v standardni izdaji jedra, a ne vse, kar podpira Linux; veliko pogostih gonilnikov (kot npr. gonilniki za PCMCIA in za nekatere tračne enote) je nalagalnih modulov, ki se urejajo in distribuirajo posebej.

2.4 Katero različico prevajalnika gcc in knjižnice libc potrebujem?

Linus priporoča različico gcc v datoteki `README`, priloženi izvorni kodi Linuxa. Če nimate te različice, vam dokumentacija poleg priporočene različice gcc pove, če morate nadgraditi knjižnico libc. To ni težko opravilo, pomembno pa je, da sledite navodilom.

2.5 Kaj je nalagalni modul (loadable module)?

To so delčki kode jedra, ki niso povezani (vključeni) direktno v jedro. Prevedemo jih posebej in jih lahko vključimo ali odstranimo v delujoče jedro skoraj kadarkoli. Zaradi njihove fleksibilnosti je to zdaj priporočen način za uporabo določenih lastnosti jedra. Mnogi popularni gonilniki, na primer gonilniki za PCMCIA in za tračno enoto QIC-80/40, so nalagalni moduli.

2.6 Koliko prostora potrebujem na disku?

Odvisno od vaše konkretne konfiguracije sistema. Komprimirana izvorna koda Linuxa različice 2.2.9 je velika približno 14 MB. Večina računalnikov jo obdrži tudi po odpakiranju. Odpakirana in zgrajena izvorna koda jedra za povprečno sestavo zasede dodatnih 67 MB.

2.7 Kako dolgo traja?

Na novejših strojih traja prevajanje znatno manj časa kot na starejših; AMD K6-2/300 s hitrim diskom lahko prevede jedro 2.2.x v približno štirih minutah. Če pa želite prevesti jedro na starih računalnikih Pentium, 486 in 386, bodite pripravljeni čakati, morda ure ali celo dneve ...

Če vas to moti in imate na voljo tudi hitrejši stroj, na katerem lahko prevajate, lahko gradite jedro na hitrejših strojih (če podate pravilna določila, če so vaši pripomočki osveženi, in tako naprej) in potem prenesete sliko jedra na počasnejši stroj.

3 Kako zares sestaviti jedro

3.1 Nabava izvorne kode

Izvorno kodo lahko dobite po anonimnem FTP-ju s strežnika `ftp.kernel.org` v imeniku `/pub/linux/kernel/vx.y`, kjer je `x.y` različica (npr. 2.2), in kot je bilo že omenjeno, so lahko razvojne različice jedra, označene z lihimi končnimi številkami, nestabilne. Izvorna koda jedra je navadno označena kot `linux-x.y.z.tar.gz`, kjer je `x.y.z` številka različice. Strežniki navadno premorejo tudi datoteke s podaljškom `.bz2`, ki so bile stisnjene s pripomočkom `bzip2` (te datoteke bodo manjše, zato bo za njihov prenos potrebno manj časa).

Najboljše bo, če za prenos uporabite `ftp.xx.kernel.org`, kjer je `xx` koda vaše države; v Sloveniji boste tako uporabili *`ftp.si.kernel.org`*, v Avstriji `ftp.at.kernel.org`, v Združenih državah Amerike pa `ftp.us.kernel.org`.

3.2 Odpakiranje izvorne kode

Prijavite se kot „`root`“ (ali pa uporabite ukaz `su`) in s `cd` spremenite imenik na `/usr/src`. Če ste ob prvi namestitvi Linuxa namestili tudi izvorno kodo jedra (večina jih stori tako), bo tam že imenik „`linux`“, ki vsebuje celotno staro drevo izvorne kode. Če imate dovolj diskovnega prostora in bi se radi počutili varno, ohranite ta imenik. Dobra ideja je, da izveste različico sistema, ki jo trenutno uporabljate, in ustrezno preimenujete imenik. Trenutno različico jedra izveste z ukazom „`uname -r`“. Če torej „`uname -r`“ pravi „1.0.9“, boste preimenovali (z „`mv`“) „`linux`“ v „`linux-1.0.9`“. Če se počutite malo bolj lahkomiselne, preprosto pobrišite celoten imenik. V vsakem primeru se prepričajte, da v `/usr/src` nimate imenika „`linux`“, preden odpakirate celotno izvorno kodo.

Zdaj v imeniku `/usr/src` odpakirajte izvorno kodo z ukazom „`tar zxpvf linux-x.y.z.tar.gz`“ (če imate samo datoteko `.tar` brez končnega `.gz` pa z ukazom „`tar xpvf linux-x.y.z.tar`“). Na zaslonu boste videli izpisane datoteke izvorne kode. Ko `tar` konča, boste imeli nov imenik `/usr/src/linux`. Naredite `cd` v `linux` in preberite datoteko `README`. Nekje bo razdelek z naslovom „`INSTALLING the kernel`“ („NAMESTITEV jedra“). Upoštevajte navodila, kjer je to primerno – simbolne povezave, ki morajo biti na pravih mestih, brisanje ostalih datotek `.o` itd.

Če imate datoteko `.bz2` in pripomoček `bzip2` (več o tem preberite na `<http://www.muraroa.demon.co.uk/>`), naredite tole:

```
bz2cat linux-x.y.z.tar.bz2 | tar xvf -
```

3.3 Nastavitev jedra

Opomba: Nekaj tega je reiteracija/razjasnitev podobnega razdelka v Linusovi datoteki `README`.

Ukaz „`make config`“ v imeniku `/usr/src/linux` požene nastavitveni skript, ki vas vpraša veliko vprašanj. Potrebuje ukazno lupino `bash`, zato preverite, če ta obstaja v `/bin/bash`, `/bin/sh`, ali `$BASH`.

Verjetno boste raje uporabljali katero od alternativ ukazu „`make config`“. Tisti, ki poganjate grafični sistem X, lahko poskusite „`make xconfig`“, če imate nameščen programski paket Tk („klik-o-rama“ – Nat). „`make menuconfig`“ je za tiste, ki imate (n)curses in bi radi imeli tekstovne menuje. Ta dva vmesnika imata bistveno prednost pred standardnim: če zamočite in med konfiguracijo izberete napačno izbiro, lahko greste nazaj in zadevo popravite.

Z uporabo „`make menuconfig`“ ali „`make xconfig`“ bodo nastavitvene izbire urejene hierarhično.

Pripravljeni ste na odgovarjanje nekaj vprašanj, navadno z „`y`“ (da) ali „`n`“ (ne). Gonilnik naprav imajo tipično izbiro „`m`“. Ta pomeni „modul“, se pravi, da ga bo sistem prevedel, ne pa tudi vključil neposredno v jedro. Na voljo bo kot nalagalni modul. Bolj duhovit način za opis te izbire bi bil „mogoče“. Nekatere bolj očitne in ne-kritične izbire tukaj niso opisane; glejte razdelek 3.3.11 („Druge nastavitvene izbire“) za kratek opis nekaterih posameznih izbir. Pri „`make menuconfig`“ s presledkom spreminjate izbiro.

V jedrih 2.0.x in poznejših je na voljo tudi izbira „?“, ki poda kratek opis posameznega nastavitvenega parametra. Ta informacija je verjetno najbolj sveža. Tukaj je seznam nekaterih najpomembnejših odlik, s hierarhijo, v kateri jih najdete, in kratkim opisom.

3.3.1 Kernel math emulation (Processor type and features)

[Emulacija matematičnih operacij v jedru (Vrsta in lastnosti procesorja)]

Če nimate matematičnega koprocesorja (imate le goli procesor 386 ali 486SX), morate tukaj reči „`y`“. Če imate koprocesor in rečete „`y`“, ne skrbite preveč – koprocesor se bo še vedno uporabljal, emulacija pa ignorirala. Za vsak napol sodoben stroj bo odgovor „ne“, a ne skrbite, če boste pomotoma rekli „da“; če emulacija ni potrebna, se ne uporablja.

3.3.2 Enhanced (MFM/RLL) disk and IDE disk/cdrom support (Block Devices)

[Podpora izboljšanim diskom (MFM/RLL) in diskom/CD-ROM-om tipa IDE (Blokovne naprave)]

Verjetno morate to podpreti; pomeni, da bo jedro podpiralo standardne trde diske, ki jih najdemo v osebnih računalnikih večine ljudi. Ta gonilnik ne vključuje pogonov SCSI; v nastavitvah pridejo ti na vrsto kasneje.

Nastavitveni program vas bo nato vprašal ali želite podporo le starim diskom („old disk-only“) in novim diskom IDE („new IDE“). Izbrati morate eno od teh možnosti; glavna razlika je v tem, da stari gonilnik podpira le dva diska na enem vmesniku, medtem ko novi podpira drugi vmesnik in CD-ROM-e IDE/ATAPI. Novi gonilnik je 4 KB večji od starejšega in naj bi bil „izpopolnjen“, kar pomeni, da poleg vsebovanja različnega števila hroščev verjetno tudi izboljša obnašanje vašega diska, posebej, če imate novejšo strojno opremo tipa EIDE.

3.3.3 Networking support (General Setup)

[Omrežna podpora (Splošne nastavitve)]

Tukaj boste zelo verjetno rekli „`y`“, saj želite, da bo vaš računalnik omrežen na Internet ali da bo dostopal vanj preko SLIP, PPP, term itd., torej s klicnim dostopom. Vendar, ker veliko paketov (kot na primer sistem X window) potrebuje omrežno podporo tudi, če vaš računalnik ne živi v pravem omrežju, boste tukaj vseeno rekli „`y`“. Pozneje vas bo program vprašal, če želite omrežno podporo protokolu TCP/IP; spet boste odgovorili z „`y`“, če niste absolutno prepričani v nasprotno.

3.3.4 System V IPC (General Setup)

[Medprocesna komunikacija Systema V (Splošne nastavitve)]

Ena najboljših definicij IPC (Interprocess Communication, medprocesna komunikacija) je v slovarčku knjige Programming Perl: „včasih se mora proces le pogovoriti z drugim procesom“. Ne preseneča nas torej, da nekateri perlovski programerji dovoljujejo procesom, da se pogovarjajo drug z drugim, kot to počne tudi veliko drugih paketov (najbolj znan je DOOM), torej izbira „`n`“ ni dobra ideja, razen, če točno veste, kaj počnete.

3.3.5 Processor family (Processor type and features)

[Procesorska družina (Vrsta in lastnosti procesorja)]

(v starejših jedrih: uporabite zastavico `-m486` za optimizacije za 486)

Včasih je to vključilo posebne optimizacije za določen procesor; jedro je teklo povsem dobro na ostalih čipih, a je bilo morda malo večje. V novejših jedrih pa to ni več res, zato vnesite procesor, za katerega prevajate jedro. Jedro za „`386`“ bo delovalo na vseh strojih.

3.3.6 SCSI support

[Podpora SCSI]

Če imate naprave tipa SCSI, recite „`y`“. Vprašani boste po nadaljnih podatkih, kot je podpora CD-ROM-om, diskom in katere vrste vmesnik SCSI imate. Za več podrobnosti preberite `SCSI-HOWTO`.

3.3.7 Network device support

[Podpora omrežnim napravam]

Če imate mrežno kartico ali bi radi uporabljali SLIP, PPP ali paralelni vmesnik za dostop na Internet, recite „`y`“. Nastavitveni skript vas bo vprašal o tipu kartice in protokolu, ki ga boste uporabljali.

3.3.8 Filesystems

[Datotečni sistemi]

Nastavitveni skript vas bo vprašal, če želite imeti naslednje datotečne sisteme podprte v jedru:

- `Standard (minix)` – novejše distribucije ne delajo datotečnih sistemov minix in veliko ljudi jih ne uporablja, a mogoče je vseeno dobra zamisel, da bi jih podprli. Nekateri programi za izdelavo „rešilnih diskov“ jih uporabljajo in veliko disket je formatiranih kot minix, saj je minixov datotečni sistem na disketah manj mukotrpen.
- `Second extended` – To je standardni datotečni sistem Linuxa. Skoraj zagotovo ga imate in morate reči „`y`“.
- `msdos` – Če želite uporabljate particije MS-DOS-a na trdem disku ali nameščati dosovske formatirane diskete, recite „`y`“. Dostopnih je še mnogo drugih tujih datotečnih sistemov.
- `/proc` – (zamisel iz Bell Labs, domnevam). Datotečnega sistema `/proc` se ne ustvari na disku; to je datotečni vmesnik do jedra in procesov. Veliko izpisovalcev procesov (npr. „`ps`“) ga uporablja. Poskusite kdaj „`cat /proc/meminfo`“ ali „`cat /proc/devices`“. Nekatere ukazne lupine (posebej `rc`) uporabljajo `/proc/self/fd` (na drugih sistemih znan kot `/dev/fd`) za vhodno/izhodne (V/I) operacije. Skoraj gotovo morate reči „`y`“; veliko pomembnih orodij za Linux je odvisnih od tega.
- `NFS` – Če vaš stroj biva na omrežju in želite uporabljati datotečne sisteme, ki ležijo na drugih sistemih z NFS, recite „`y`“.
- `ISO9660` – Najdete ga na večini CD-ROM-ov. Če imate pogon za CD-ROM in ga želite uporabljati v Linuxu, recite „`y`“.

Vendar jaz ne vem, katere datotečne sisteme potrebujem! Prav, napišite „`mount`“. Izpis bo približno takšen:

```
blah:# mount
/dev/hda1 on / type ext2 (defaults)
/dev/hda3 on /usr type ext2 (defaults)
none on /proc type proc (defaults)
/dev/fd0 on /mnt type msdos (defaults)
```

Poglejte v vsako vrstico; beseda poleg „`type`“ je ime datotečnega sistema. V tem primeru sta moja datotečna sistema `/` in `/usr` tipa „second extended“, uporabljam `/proc` in nameščena je disketa z datotečnim sistemom msdos (bljak).

Poskusite tudi „`cat /proc/filesystems`“, če imate trenutno vklopljen `/proc`; to bo izpisalo vaše trenutne datotečne sisteme v jedru.

Namestitev redko uporabljanih, ne-nujnih datotečnih sistemov lahko povzroči napihnjeno jedro; glejte razdelek 9 (o modulih) za način, kako se temu izognete in razdelek 7.2 („Velika ali počasna jedra“) o tem, zakaj je napihnjeno jedro nezaželjen pojav.

3.3.9 Character devices

[Znakovne naprave]

Tukaj vključite gonilnike za vaš tiskalnik (pravzaprav, tiskalnik na vzporednih vratih), miško, priklopljeno na vrata busmouse ali PS/2 (veliko notesnikov uporablja miškovni protokol PS/2 za njihove vgrajene sledilne kroglice), nekatere tračne enote in druge takšne „znakovne“ naprave. Recite „`y`“, če je tako prav.

Opomba: `gpm` je program, ki vam omogoča uporabo miške izven sistema X window za izrezovanje in prilepljanje besedila med navideznimi zasloni. Dobro je, če imate miško na zaporednih vratih, saj ta lepo shaja z Okni X, za druge miške pa morate uporabiti posebne trike.

3.3.10 Sound

[Zvok]

Če imate veliko željo slišati, kako `biff` laja, recite „`y`“, in nastavitvenemu programu lahko poveste vse o vaši zvočni kartici. (Opomba glede nastavitve zvočne kartice: ko vas vpraša, če želite namestiti popolno različico gonilnikov, lahko rečete „`n`“ in prihranite nekaj pomnilnika jedra z izbiro le tistih lastnosti, ki se vam zdijo potrebne.)

Če zares potrebujete dobro podporo zvočne kartice, poglejte proste gonilnike na naslovu `<http://www.linux.org.uk/OSS/>` in komercialni Open Sound System na `<http://www.opensound.com/>`.

3.3.11 Druge nastavitvene možnosti

Vse nastavitvene izbire tukaj niso naštete, saj se prepogosto spreminjajo ali so same po sebi razvidne (na primer, podpora 3Com 3C509 za točno to mrežno kartico). Obstaja precej obsežen seznam vseh izbir (in način, kako jih uvrstimo v skript `Configure`); projekt je začel in vzdrževal Axel Boldt (`boldt@math.ucsb.edu`) in je dostopen kot pomoč na zvezi. Na voljo je tudi kot ena sama velika datoteka `Documentation/Configure.help` v izvorni kodi jedra od različice Linuxa 2.0 naprej.

3.3.12 Kernel hacking

[Hekanje jedra]

Iz Linusove datoteke `README`:

Izbira „hekanje jedra" navadno vodi v večje in počasnejše jedro (ali v oboje) in lahko naredi jedro manj stabilno tako, da prekodira nekatere rutine, ki aktivno poskušajo sesuti slabo kodo in s tem najti jedrne probleme (`kmalloc()`). Torej boste, če ste navadni smrtnik, tukaj odgovorili z „`n`".

3.4 Pa zdaj? (Datoteka **`Makefile`**)

Ko opravite nastavljanje, vam sporočilo pove, da je jedro nastavljeno in da naj pogledate „najvišje-nivojsko datoteko Makefile za dodatno nastavitev" itd.

Poglejte torej `Makefile`. Verjetno vam je ne bo treba spreminjati, a nikoli ne škodi, če pogledate. Po namestitvi novega jedra lahko spreminjate izbire tudi z ukazom „`rdev`". Če se ob ogledu te datoteke počutite izgubljeni, jo pač pozabite.

4 Prevajanje jedra

4.1 Čiščenje in urejanje odvisnosti

Ko konfiguracijski skript konča z delom, vam pove, da napravite „`make dep`" in (morda) „`clean`". Torej napišete „`make dep`". To vam zagotovi, da so vse odvisnosti, na primer vključne datoteke, na svojem mestu. To ne traja dolgo, razen, če imate zelo počasen računalnik. Pri starejših različicah jedra morate po koncu delanja odvisnosti napisati še „`make clean`". To odstrani vse objektne datoteke in druge stvari, ki so jih pustile za sabo stare različice. V vsakem primeru ne pozabite narediti tega koraka preden začnete prevajati jedro.

4.2 Čas za prevajanje

Po urejanju odvisnosti in čiščenju lahko napišete „`make zImage`" ali „`make zdisk`" (ta del traja veliko časa). „`make zImage`" prevede jedro in v imeniku `arch/i386/boot` pusti datoteko, imenovano „`bzImage`" (med drugim). To je novo komprimirano jedro. „`make bzdisk`" naredi isto stvar, le da prepiše novo datoteko `bzImage` na disketo, ki ste jo, upajmo, vstavili v pogon „A:" (`/dev/fd0`). „`bzdisk`" je priročno orodje za testiranje novih jeder; če novo jedro ne deluje v redu, preprosto odstranite disketo in zaženite staro jedro. Včasih boste to disketo lahko uporabili tudi, če boste po pomoti odstranili svoje jedro (ali naredili kaj podobno groznega). Disketo lahko uporabite tudi pri inštalaciji novih sistemov, ko preprosto prepišete vsebino enega diska na drugega („Vse to in še več! Koliko bi plačali **zdaj¿**‘). Vsa, vsaj na pol razumno nova, jedra so komprimirana, odtod črka „`bz`" pred njihovimi imeni. Komprimirano jedro se samodejno odkomprimira, ko se izvaja.

V starejših jedrih ni izbire za gradnjo `bzImage`; le `zImage`. Ta izbira je trenutno še vedno dostopna, vendar je glede na velikost kode novejših jeder uporaba `bzImage` bolj ali manj obvezna, saj starejše metode ne znajo uporabljati prevelikega jedra.

4.3 Drugi cilji „make"

„`make mrproper`" naredi bolj intenzivno čiščenje („`clean`"). Včasih je ta cilj potreben; morda ga želite uporabiti ob vsakem popravku. „`make mrproper`" bo tudi pobrisal vašo konfiguracijsko datoteko, zato shranite njeno rezervno kopijo (`.config`), če se vam zdi pomembna.

„`make oldconfig`" bo poskušal nastaviti jedro s stare konfiguracijske datoteke; namesto vas bo šel skozi proces „`make config`". Če še nikoli niste prevedli jedra ali nimate stare konfiguracijske datoteke, verjetno nočete tega, saj hočete spremeniti privzeto nastavitev.

Za ukaz „`make modules`" glejte razdelek o modulih.

4.4 Namestitev jedra

Ko imate novo jedro, za katerega menite, da deluje, kot želite, je čas za njegovo namestitev. Večina ljudi za ta korak uporablja LILO (Linux Loader). Jedro namestite, poženete čez njega LILO in ga pripravite za zaganjanje z ukazom „`make bzlilo`“. **Vendar le**, če je `lilo` nastavljen takole: jedro je `/vmlinuz`, lilo je v imeniku `/sbin`, in vaša nastavitvena datoteka `/etc/lilo.conf` se s tem strinja.

V vseh drugih primerih morate pognati LILO neposredno. Paket je precej enostaven za inštalacijo in delo, a zna zmesti ljudi s konfiguracijsko datoteko. Glejte nastavitveno datoteko (v starejših različicah je to `/etc/lilo/config`, v novejših pa `/etc/lilo.conf`) in poglejte, kakšne nastavitve imate. Konfiguracijska datoteka mora izgledati podobno:

```
image = /vmlinuz
    label = Linux
    root = /dev/hda1
    ...
```

Nastavitev „`image =`“ kaže na na novo nameščeno jedro. Večina ljudi uporablja `/vmlinuz`. Lilo potrebuje oznako „`label`“, da ugotovi, katero jedro ali operacijski sistem naj zažene, oznaka „`root`“ je korenski imenik `/` določenega operacijskega sistema. Napravite rezervno kopijo vašega starega jedra in prepišite datoteko `bzImage`, ki ste jo pravkar naredili na to mesto (napišete npr. „`cp bzImage /vmlinuz`“, če uporabljate „`/vmlinuz`“). Potem še enkrat zaženete lilo - na novejših sistemih le napišete „`lilo`“, na starejših morate morda narediti `/etc/lilo/install` ali celo `/etc/lilo/lilo -C /etc/lilo/config`.

Če bi radi izvedeli več o nastavitvi programa LILO ali če nimate programa LILO, dobite najnovejšo različico z vašega priljubljenega mesta za FTP in upoštevate navodila.

Za zaganjanje enega vaših starejših jeder s trdega diska (še en način, kako si opomorete, če ste uničili novo jedro), prekopirajte vrstice pod (in vključno z) „`image = xxx`“ v LILO-vi nastavitveni datoteki na konec datoteke in spremenite „`image = xxx`“ v „`image = yyy`“, kjer je „yyy“ polna pot do datoteke, v katero ste shranili rezervno jedro. Potem spremenite „`label = zzz`“ v „`label = linux-backup`“ in še enkrat poženite `lilo`. V konfiguracijsko datoteko lahko dodate tudi vrstico z „`delay=x`“, kjer je „x“ časovni interval v desetinkah sekunde, v katerem LILO čaka, da ga boste lahko prekinili (npr. s tipko Shift) in vpisali oznako rezerve zaganjalne kopije (če se zgodi kaj neprijetnega).

5 Popravljanje jedra

5.1 Uporaba popravka

Zaporedne nadgradnje jedra se distribuirajo kot popravki (patches). Na primer, če imate različico 1.1.45 in opazite, da obstaja nekje datoteka „`patch46.gz`“ za njo, to pomeni, da lahko z uporabo programa `patch` nadgradite jedro na različico 1.1.46. Morda boste najprej želeli napraviti rezervno kopijo drevesa izvorne kode (komprimiran arhiv naredite z „`make clean`“ in potem „`cd /usr/src; tar zcvf old-tree.tar.gz linux`“).

Nadaljujmo zgornji primer in predpostavimo, da imate datoteko „`patch46.gz`“ v imeniku `/usr/src`.Naredite `cd /usr/src` in potem „`zcat patch46.gz | patch -p0`“ (ali „`patch -p0 < patch46`“, če popravek ni komprimiran). Na zaslonu bodo mimo vas letele stvari, ki vam bodo sporočale, da `patch` poskuša uporabiti določene popravke in uspeh teh poskusov. Navadno se vse odvija prehitro, da bi lahko brali. Če niste prepričani, ali je šlo vse po sreči, boste morda uporabili zastavico `-s` za program `patch`, kar pove `patch`u naj sporoča le sporočila o napakah (v tem primeru ne boste imeli občutka „Hej, moj računalnik za spremembo nekaj počne¡‘, a boste morda vseeno raje storili tako). Če vas zanima, kateri deli se niso popravili povsem gladko, napravite cd`/usr/src/linux` in poiščite datoteke s podaljškom `.rej`. Nekatere starejše različice `patch`a pustijo podaljšek `#`. Za iskanje uporabite „`find`“:

```
 # find .
-name '*.rej' -print
```

To izpiše vse datoteke s podaljškom `.rej`, ki prebivajo v trenutnem imeniku ali podimenikih, na standardni izhod.

Če je šlo vse kot po maslu, napravite „`make clean`“, „`config`“, in „`dep`“ kot je opisano v razdelkih 3 („Kako zares sestaviti jedro“) in 4 („Prevajanje jedra“).

Ukaz `patch` ima še precej dodatnih izbir. Zgoraj smo že omenili `patch -s`, ki zadrži izpis vseh sporočil, razen napak. Če imate izvorno kodo jedra v kakšnem drugem imeniku kot `/usr/src/linux`, uporabite v tem imeniku `patch -p1`. Ostale izbire najdete z `man patch`.

5.2 Če se kje zalomi

(Opomba: Ta razdelek se nanaša predvsem na zelo stara jedra.)

Najpogostejši problem, ki se je včasih pojavljal, je bil, ko je popravek spremenil datoteko „`config.in`“ in ta ni bila čisto prava, saj ste spremenili izbire, da bi opisali opremo svojega stroja. To se je uredilo, a lahko v starejših izdajah še vedno srečate. Popravite pa tako, da pogledate datoteko `config.in.rej` tako, da odgovarja originalnemu popravku. Popravki bodo navadno označeni s simboloma „+“ in „-“ na začetku vrstice. Glejte okoliške vrstice in se spomnite, ali so bile nastavljene kot „y“ ali kot „n“. Zdaj popravite `config.in` in spremenite „y“ v „n“ in obratno, kjer je to primerno. Naredite:

```
# patch -p0 < config.in.rej
```

in če `patch` sporoči, da je uspel, lahko nadaljujete z nastavitvami in prevajanjem. Datoteka `config.in.rej` vam bo ostala, a jo lahko pobrišete.

Če imate še vedno probleme, ste morda namestili popravek prek vrste. Če `patch` pravi „`previously applied patch detected: Assume -R?`“, verjetno poskušate namestiti popravek, ki je starejši od trenutne različice jedra; če odgovorite z „y“, bo `patch` poskušal podgraditi vašo izvorno kodo, in najverjetneje mu bo spodletelo. Potrebovali boste popolno čisto novo drevo izvorne kode (kar je morda tako ali tako dobra ideja).

Za razveljavitev popravkov uporabite ukaz „`patch -R`“ na originalnem popravku.

Najboljša stvar, ki jo lahko naredite, ko ne morete uporabiti popravkov, je, da začnete znova s čistim drevesom izvorne kode (na primer, z eno od datotek `linux-x.y.z.tar.gz`), in znova začnete.

5.3 Kako se znebite datotek `.orig`

Po nekaj popravkih se bodo začele kopičiti datoteke `.orig`. Npr. neko drevo jedra 1.1.51, ki sem ga nekoč imel, je bilo zadnjič očiščeno pri različici 1.1.48. Odstranitev datotek `.orig` je prihranila več kot pol megabyta.

Zadevo uredite z ukazom:

```
# find . -name '*.orig' -exec rm -f {} ';'
```

Različice programa `patch`, ki uporabljajo `#` za zavrnitvene datoteke, uporabljajo za podaljšek tildo („`.~`“) namesto „`.orig`“.

Obstajajo tudi boljši načini za odpravo datotek `.orig`, ki slonijo na GNU `xargs`:

```
# find . -name '*.orig' | xargs rm
```

ali „precej varna, a malo bolj izčrpna“ metoda:

```
# find . -name '*.orig' -print0 | xargs --null rm --
```

5.4 Drugi popravki

Razen Linusovih, obstajajo tudi drugi popravki (rekel jim bom „nestandardni“). Če jih uporabite, Linusovi popravki morda ne bodo delovali pravilno in jih boste morali obnoviti, urediti izvorno kodo popravka, namestiti novo drevo izvorne kode ali kombinacijo naštetega. To lahko postane zelo frustrirajoče, zato, če ne želite spreminjati izvorne kode (z morda slabim izidom), napravite rezervne kopije nestandardnih popravkov, preden uporabite Linusove, ali le namestite novo drevo. Potem lahko pogledate, če nestandardni popravki delujejo. Če ne, morate ostati pri starem jedru in se igrati s popravki ali izvorno kodo, da bi dosegli delovanje, ali čakati (morda celo prosjačiti) za novo različico popravkov.

Kako pogosti so popravki, ki niso v standardnih distribucijah? Verjetno boste slišali o njih. Jaz uporabljam popravek `noblink` za moje navidezne zaslone, ker sovražim utripajoče kazalce (ta popravek je (ali je vsaj bil) redno osvežen ob vsaki novi izdaji jedra). Z razvojem vse več novejših gonilnikov naprav kot modulov pa uporaba „nestandardnih“ popravkov znatno upada.

6 Dodatni paketi

Vaše Linuxovo jedro ima veliko odlik, ki niso pojasnjene v sami izvorni kodi jedra; te posebnosti se navadno dosežejo z uporabo zunanjih paketov. Tukaj naštevam nekaj najpogostejših.

6.1 kbd

Linuxov zaslon (console) ima najbrž več zmožnosti, kot si jih zasluži (op.: prevajalec se ne strinja). Med temi so možnost preklaplanja znakov, preslikave tipkovnice, preklop video načinov (v novejših jedrih) itd. Paket kbd vsebuje programe, ki uporabniku omogočajo vse to, in še veliko znakov in načrtov tipkovnic za skoraj vsako tipkovnico. Paket dobite na istih mestih kot izvorno kodo jedra.

6.2 util-linux

Rik Faith <*faith@cs.unc.edu*> je sestavil veliko zbirko uporabnih programov za Linux, ki se imenujejo util-linux. Trenutno jih vzdržuje Andries Brouwer <*util-linux@math.uio.no*>. Po anonimnem FTP-ju dobite na <ftp://metalab.unc.edu/pub/Linux/system/misc/> programe kot so `setterm`, `rdev`, in `ctrlaltdel`, ki se nanašajo na jedro. Kot pravi Rik, ničesar ne inštalirajte, ne da bi prej premislili, ni vam treba inštalirati vsega in lahko imate resne probleme, če bi radi inštalirali vse.

6.3 hdparm

Kot veliko paketov je bil tudi ta nekoč popravek za jedro in podporni programi. Popravki so se prebili v uradno jedro, programi za optimizacijo in igranje z vašim trdim diskom pa se razširjajo posebej.

6.4 gpm

gpm pomeni „splošno uporabna miška“ (ang. general purpose mouse). S tem programom lahko besedilo izrezujete in prilepljate med posameznimi navideznimi zasloni in počnete druge stvari s celo paleto različnih mišk.

7 Nekatere pasti

7.1 make clean

Če po rutinski nadgradnji jedra vaše jedro počne zares čudne reči, ste morda pozabili napisati `make clean` pred prevajanjem novega jedra. Simptomi so lahko karkoli, od zmrznjenega sistema, čudnih V/I problemov, do slabega (počasnega) delovanja. Prepričajte se tudi, da boste ukazali `make dep`.

7.2 Velika ali počasna jedra

Če vaše jedro požira velike količine pomnilnika, je preveliko, in/ali le traja neskončno dolgo, da se prevede, čeprav imate nov procesor Quadbazillium-III/440, ste najverjetneje vključili podporo veliko nepotrebnih zadev (gonilnikov naprav, datotečnih sistemov, itd.). Če naprave ne uporabljate, je ne podprite v jedru, saj to zavzema pomnilnik. Najbolj očiten simptom prenapihnjenega jedra je ekstremno izmenjavanje pomnilnika z diskom sem ter tja; če vaš disk nenehno oglaša in ni eden od tistih starih Fujitsujevih Eagles, katerih zvok lahko primerjamo s pristajanjem reaktivnih letal, preglejte nastavitve vašega jedra.

Koliko pomnilnika zaseda jedro lahko izveste z odštevanjem vrednosti „total mem" v izpisu `/proc/meminfo` ali izhodom ukaza „`free`", od količine vsega pomnilnika.

7.3 Vzporedna vrata ne delujejo/tiskalnik ne deluje

Nastavitvene izbire za PC-je so: najprej v kategoriji Splošnih nastavitev (angl. General Setup) vključite podporo zaporednih vrat (angl. Parallel port support) in strojno opremo osebnih računalnikov (angl. PC-style hardware). Nato v Znakovnih napravah (angl. Character devices) podprite tiskalnik na vzporednih vratih (angl. Parallel printer support).

Potem so tu še imena. Linux 2.2 poimenuje tiskalniške naprave drugače od prejšnjih izdaj. Posledica tega je, da napravi `lp1` v vašem starem jedru v novem jedru verjetno ustreza naprava `lp0`. Uporabite `dmesg` ali poglejte v dnevnik v imeniku `/var/log` ter ugotovite novo ime.

7.4 Jedro se ne prevede

Če se ne prevede, je to verjetno zato, ker je popravek spodletel, ali je vaša izvorna koda nekako pokvarjena. Morda nimate prave različice prevajalnika `gcc`, ali je tudi z njim kaj narobe (na primer, vključne datoteke so lahko napačne). Prepričajte se, da so simbolične povezave, ki jih Linus priporoča v datoteki `README` pravilno narejene. V splošnem, če se standardna jedra ne prevajajo, je nekaj resno narobe s sistemom in ponovna inštalacija nekaterih orodij je neizogibna.

V nekaterih primerih lahko `gcc` odpove zaradi strojnih problemov. Sporočila o takšnih napakah so nekaj kot „xxx exited with signal 15" in navadno izgledajo zelo skrivnostna. Tega sploh ne bi omenil, a se mi je nekoč zgodilo – imel sem nekaj slabega predpomnilnika in prevajalnik se je pritoževal povsem naključno. Če imate težave, poskusite najprej ponovno namestiti gcc. Sumničavi postanite samo, če se vaše jedro lepo prevede z izključenim zunanjim predpomnilnikom, zmanjšano količino RAM-a, ipd.

Ljudje so navadno vznemirjeni, ko izvejo, da bi lahko imeli tudi težave s strojno opremo. Hja, tega si ne izmišljujem. Obstajajo tudi pogosto zastavljena vprašanja o tej temi – najdete jih na `<http://www.bitwizard.nl/sig11/>`.

7.5 Novo jedro se noče zagnati

Niste pognali programa LILO, ali pa ga niste pravilno nastavili. Nekoč me je zafrkavala vrstica v LILO-vi konfiguracijski datoteki, ki je bila „`boot = /dev/hda1`“ namesto „`boot = /dev/hda`“. (To je lahko sprva zelo moteče, a ko imate enkrat delujočo nastavitveno datoteko, vam je ni treba spreminjati.)

7.6 Pozabili ste pognati LILO, ali pa se sistem sploh ne zažene

Ups! Najboljše, kar lahko storite ta hip, je, da zaženete operacijski sistem z diskete ali CD-ROM-a in potem pripravite še eno zaganjalno disketo (kot bi jo naredil ukaz „`make zdisk`“). Vedeti morate, kje je vaš korenski (`/`) datotečni sistem in katerega tipa je (npr. second extended, minix). V spodnjem primeru morate vedeti tudi na kakšnem datotečnem sistemu leži vaše drevo izvorne kode `/usr/src/linux`, njegov tip, in kje je navadno nameščen (z `mount`).

V naslednjem primeru je `/` enak `/dev/hda1`, in datotečni sistem, ki vsebuje `/usr/src/linux` na `/dev/hda3`, navadno nameščen na `/usr`. Oba sta datotečna sistema tipa ext2 (second extended). Delujoča slika jedra v imeniku `/usr/src/linux/arch/i386/boot` se imenuje `bzImage`.

Zamisel je takšna, da uporabimo delujoče jedro `zImage` na novi disketi. Še ena možnost, ki lahko deluje bolje, ali pa tudi ne (odvisno od konkretne metode, s katero ste zavozili svoj sistem), je opisana po tem primeru.

Najprej zaženite sistem s kombinacije disket boot in root ali z reševalne diskete in namestite delujočo sliko jedra:

```
# mkdir /mnt
# mount -t ext2 /dev/hda3 /mnt
```

Če vam `mkdir` pravi, da imenik že obstaja, ga ignorirajte. Zdaj pojdite z ukazom `cd` na imenik, v katerem je delujoče jedro. Pozorni bodite na to, da je

```
/mnt + /usr/src/linux/arch/i386/boot - /usr = /mnt/src/linux/arch/i386/boot
```

V disketni pogon „A:“ vložite formatirano disketo (ne vaših diskov boot ali root!), prepišite sliko jedra na disketo in jo nastavite za svoj korenski datotečni sistem.

```
# cd /mnt/src/linux/arch/i386/boot
# dd if=bzImage of=/dev/fd0
# rdev /dev/fd0 /dev/hda1
```

Naredite `cd` na `/` in odmestite običajni datotečni sistem `/usr`:

```
# cd /
# umount /mnt
```

Zdaj lahko še enkrat zaženete svoj sistem z nove diskete. Ne pozabite tokrat po zagonu pognati `lilo` (ali karkoli je bilo že narobe)!

Kot smo omenili zgoraj, obstaja še ena običajna pot. Če imate delujočo sliko jedra v `/` (`/vmlinuz`, na primer), jo lahko uporabite za zagonsko disketo. Če veljajo vsi zgoraj našteti pogoji in je slika jedra `/vmlinuz`, naredite le te spremembe v zgoraj opisanem primeru: spremenite `/dev/hda3` v `/dev/hda1` (datotečni sistem `/`), `/mnt/src/linux` v `/mnt`, in `if=bzImage` v `if=vmlinuz`. Opombo o tem, kako dobimo `/mnt/src/linux` lahko spregledate.

Uporaba programa LILO na velikih diskih (večjih od 1024 cilindrov) lahko povzroča probleme. Preberite LILO mini-HOWTO ali LILO-vo dokumentacijo, če potrebujete pomoč pri tem.

7.7 Izpiše „`warning: bdflush not running`“

To je lahko resen problem. Od jedra izdaje po 1.0 (okoli 20. aprila 1994) se je program „`update`“, ki periodično izplakne vmesni pomnilnik datotečnega sistema, posodabljal in nadomestil. Dobite izvorno kodo „`bdflush`“ (najdete jo tam, kjer ste našli jedro) in namestite ta program (verjetno boste medtem pognati vaš sistem pod starim jedrom). Ta program se sam namesti kot „`update`“ in po ponovnem zagonu se novo jedro ne bo več pritoževalo.

7.8 Mojega CD-ROM-a IDE/ATAPI ne prepričam, da bi deloval

Čudno, a veliko ljudi ne more pripraviti svoje pogone ATAPI k delovanju, verjetno zato, ker gre lahko veliko stvari narobe.

Če je vaš CD-ROM edina naprava na konkretnem vmesniku IDE, morate nastaviti skakače kot „master“ ali „single“. To je menda najbolj pogosta napaka.

Creative Labs (na primer) je postavil vmesnik IDE na njihove zvočne kartice. A to vodi k zanimivem problemu, da imajo nekateri ljudje en sam vmesnik, veliko jih ima dva vmesnika IDE na njihovih matičnih ploščah (navadno na IRQ15), torej je splošna praksa označiti vmesnik SoundBlaster-ja kot tretji IDE port (IRQ11, mi pravijo).

To povzroča probleme z Linuxom, saj različice 1.2.x ne podpirajo tretjega vmesnika IDE (obstaja podpora v serijah 1.3.x, a to je razvojna različica, se še spomnite, in ne izvaja avtomatskega iskanja). Temu se lahko izognemo na več načinov.

Če že imate druga vrata IDE, jih morda ne uporabljate ali še nimajo na sebi dveh naprav. Vzemite pogon ATAPI z zvočne kartice in ga povežite na drugi vmesnik. Potem lahko onemogočite vmesnik zvočne kartice, kar tako ali tako privarčuje IRQ.

Če nimate drugega vmesnika, nastavite skakač na vmesniku zvočne kartice (ne na zvočnem delu zvočne kartice) kot IRQ15, drugi vmesnik. Moralo bi delovati.

7.9 Izpisuje čudne reči o zastarelih zahtevah za usmerjanje (obsolete routing requests)

Poiščite novo različico programa `route` in vseh drugih usmerjevalnih programov. Datoteka `/usr/include/linux/route.h` (ki je pravzaprav v imeniku `/usr/src/linux`) se je spremenila.

7.10 Požarni zid ne deluje v 1.2.0

Nadgradite vsaj na različico 1.2.1.

7.11 „`Not a compressed kernel Image file`“ (datoteka s sliko jedra ni komprimirana)

Ne uporabljajte datoteke `vmlinux`, ki je narejena v imeniku `/usr/src/linux`, kot vašo zaganjalno sliko; `[..]/arch/i386/boot/bzImage` je prava datoteka.

7.12 Težave z zaslonskim terminalom po nadgradnji na 1.3.x

Spremenite besedo `dumb` v `linux` v opisu zaslonskega terminala v datoteki `/etc/termcap`. Morda boste morali tudi narediti nov zapis.

7.13 Po nadgradnji jedra ne morem prevajati zadev

Linuxova izvorna koda jedra vsebuje veliko vključnih datotek (datoteke, ki se končujejo na `.h`), na katere se sklicujejo standardne datoteke v imeniku `/usr/include`. Na njih se navadno sklicujemo takole (tukaj je `xyzzy.h` nekaj v imeniku `/usr/include/linux`):

```
#include <linux/xyzzy.h>
```

Navadno je v imeniku `/usr/include` povezava, imenovana `linux`, na imenik `include/linux` vaše izvorne kode jedra (`/usr/src/linux/include/linux` v tipičnem sistemu). Če te povezave ni tam, ali kaže na napačen kraj, se večina stvari sploh ne bo prevedla. Če ste se odločili, da porablja izvorna koda jedra preveč prostora na disku in ste jo pobrisali, je očitno to problem. Lahko pa, da je kaj narobe z dovoljenji datotek; če ima vaš `root` nastavitev umask, ki ne dovoljuje drugim uporabnikom, da bi kot privzeto lahko gledali njegove datoteke, in ste izluščili izvorno kodo jedra brez izbire `p` (ohrani datotečne načine), ti uporabniki ne bodo mogli uporabljati prevajalnika za C. Čeprav lahko uporabite ukaz `chmod` in to popravite, je verjetno lažje še enkrat izvleči vključne datoteke. To lahko storite enako kot ste storili na začetku z vso izvorno kodo, le z dodatnim argumentom:

```
# tar zxvpf linux.x.y.z.tar.gz linux/include
```

Opomba: „`make config`“ bo naredil povezavo `/usr/src/linux`, če je še nimate.

7.14 Povečanje omejitev

Naslednji primer ukazov je lahko koristen za tiste, ki se sprašujete, kako povečati nekatere mehke omejitve, ki jih privzame jedro:

```
# echo 4096 > /proc/sys/kernel/file-max
# echo 12288 > /proc/sys/kernel/inode-max
# echo 300 400 500 > /proc/sys/vm/freepages
```

8 Opomba o nadgradnji na različice 2.0.x, 2.2.x

Jedra različic 2.0.x in 2.2.x so uvedla precej sprememb pri njihovi namestitvi. Berite datoteko `Documentation/Changes` v drevesu izvorne kode jedra za znanje, ki ga morate imeti, ko nadgrajujete na ta jedra. Verjetno boste morali nadgraditi veliko ključnih paketov, kot so gcc, libc in SysVInit, in spremeniti veliko sistemskih datotek, zato bodite na to pripravljeni. A brez panike, prosim.

9 Moduli

Nalagalni moduli lahko prihranijo pomnilnik in poenostavijo konfiguracijo. Domet modulov je razširjen na datotečne sisteme, gonilnike omrežnih kratic, tračnih enot, tiskalnikov in še več.

9.1 Namestitev modulskih pripomočkov

Modulski pripomočki so na voljo, kadarkoli dobite izvorno kodo vašega jedra kot `modutils-x.y.z.tar.gz`; izberite najvišjo številko različice `x.y.z`, ki je enaka ali manjša vašemu jedru. Odpakirajte jih z „`tar zxvf modutils-x.y.z.tar.gz`“, pojdite s `cd` na imenik, ki ga ustvari `tar` (`modutils-x.y.z`), preglejte datoteko `README`, in upoštevajte navodila (kar je navadno nekaj preprostega, kot, denimo, `make install`). Zdaj morate imeti programe insmod, rmmod, ksyms, lsmod, genksyms, modprobe, in depmod v imeniku /sbin. Če želite, lahko preskusite pripomočke s preizkuševalnim gonilnikom „hw" v programu insmod; preberite datoteko INSTALL v tem podimeniku za podrobnosti.

insmod vključi modul v tekoče jedro. Moduli imajo navadno podaljšek .o; preizkuševalni gonilnik, omenjen zgoraj, se imenuje drv_hello.o, torej morate napisati „insmod drv_hello.o", če ga želite vključiti. Module, ki jih jedro trenutno uporablja, lahko izpišete z lsmod. Izhod izgleda takole:

```
blah:# lsmod
Module:        #pages:  Used by:
drv_hello          1
```

„drv_hello" je ime modula, uporablja eno stran (4 KB) pomnilnika in noben drug jedrni modul trenutno ni odvisen od njega. Ta modul odstranite z ukazom „rmmod drv_hello". Paziti morate, ker hoče rmmod ime modula, ne ime datoteke; dobite ga z izpisom lsmod. Nameni drugih modulskih pripomočkov so našteti v njihovih referenčnih priročnikih (npr. man ksyms).

9.2 Moduli, distribuirani poleg jedra

Od različice 2.0.30 je večina vsega dostopna kot nalagalni modul. Če jih želite uporabiti, morate nastaviti podatke o njih v običajnem jedru; to se pravi, ne rečete „y" med „make config", temveč „m". Prevedite novo jedro in z njim zaženite sistem. Potem naredite „cd /usr/src/linux" in ukažite „make modules". To prevede vse module, ki jih niste že navedli v konfiguraciji jedra in v imenik /usr/src/linux/modules namesti povezave na njih. Uporabite jih lahko v tem imeniku ali pa izvedete „make modules_install" in jih s tem namestite v imenik /lib/modules/x.y.z, kjer je x.y.z številka izdaje jedra.

To je lahko še posebno uporabno z datotečnimi sistemi. Morda ne uporabljate pogosto datotečnih sistemov minix in/ali msdos. Na primer, kadar dobim dosovsko (brrr) disketo, naredim insmod /usr/src/linux/modules/msdos.o, in potem rmmod msdos, ko opravim z njo. Ta postopek privarčuje okoli 50 KB RAM-a v jedru med normalnim delovanjem. Pri datotečnem sistemu minix ne bo odveč majhna pripomba: vedno ga podprite neposredno v jedru, da boste lahko uporabljali „reševalne" diskete.

10 Nasveti in triki

10.1 Preusmeritev izhoda ukazov make in patch

Če želite videti, kaj je naredil ukaz „make" ali „patch", lahko preusmerite standardni izhod programa v datoteko. Najprej ugotovite katero ukazno lupino uporabljate: „grep root /etc/passwd" in glejte nekaj podobnega temu: „/bin/csh".

Če uporabljate sh ali bash, boste takole preusmerili izhod ukaza *(ukaz)* v datoteko *(izhodna_datoteka)*:

```
# (ukaz) 2>&1 | tee (izhodna_datoteka)
```

Za csh ali tcsh uporabite:

```
# (ukaz) |& tee (izhodna_datoteka)
```

Za lupino rc (verjetno je ne uporabljate) je ustrezen ukaz

```
# (ukaz) >[2=1] | tee (izhodna_datoteka)
```

10.2 Pogojna inštalacija jedra

Razen z uporabo disket je še več metod preizkušanja novega jedra, ne da bi se dotaknili starega. Za razliko od mnogih Unixov je LILO sposoben zagnati jedro s kateregakoli mesta na disku (če imate disk večji od 500 MB, preberite LILO-vo dokumentacijo, kako preprečite težave). Če torej na konec konfiguracijske datoteke dodate nekaj podobnega:

```
image = /usr/src/linux/arch/i386/boot/bzImage
    label = new_kernel
```

lahko izberete zagon novega jedra ne da bi se dotaknili vašega starega jedra `/vmlinuz` (seveda morate še pognati `lilo`). Najpreprostejši način za zagon novega jedra je, da pritisnete ob zagonu tipko Shift (ko se na zaslonu izpiše `LILO` in nič drugega), kar vam da pozivnik. Zdaj lahko vnesete „`new_kernel`" in zagnalo se bo novo jedro.

Če želite obdržati več dreves izvorne kode različnih jeder (to lahko sicer zaseda veliko diskovnega prostora), je najpogostejši način ta, da jih preimenujete v `/usr/src/linux-x.y.z`, kjer je `x.y.z` različica jedra. Potem lahko „izberete" drevo izvorne kode s simbolično povezavo, npr. „`ln -sf linux-1.2.2 /usr/src/linux`" naredi drevo 1.2.2 za trenutno aktualno drevo. Preden naredite to simbolično povezavo, se prepričajte, da zadnji argument programu `ln` ni pravi imenik (stare simbolične povezave so v redu); rezultat ne bo tak, kot bi želeli.

10.3 Nadgradnje jedra

Russell Nelson <nelson@crynwr.com> zbira spremembe v novih izdajah jedra. Te so kratke, lahko jih pogledate, preden nadgradite svoje jedro. Najdete jih na <ftp://ftp.emlist.com/pub/kchanges/> ali prek svetovnega spleta na naslovu `url url="http://www.crynwr.com/kchanges"`>.

11 Ostali HOWTO-ji, ki bi lahko bili uporabni

- `Sound-HOWTO`: zvočne kartice in pripomočki,
- `SCSI-HOWTO`: vse o krmilnikih in napravah SCSI,
- `NET-2-HOWTO`: omreženost,
- `PPP-HOWTO`: omreženost s PPPjem, posebej,
- `PCMCIA-HOWTO`: o gonilnikih za vaš notesnik,
- `ELF-HOWTO`: ELF: kaj je to, prenos,
- `Hardware-HOWTO`: pregled podprte strojne opreme,
- `Module mini-HOWTO`: več o modulih jedra,
- Kerneld mini-HOWTO: o demonu `kerneld`,
- BogoMips mini-HOWTO: če se slučajno sprašujete.

12 Razno

12.1 Avtor

Avtor in vzdrževalec priročnika Linux Kernel-HOWTO je Brian Ward <*bri@cs.uchicago.edu*>. Prosim, pošljite mi vse pripombe, dodatke, popravke (predvsem popravki so zame najbolj pomembni).

Mojo domačo stran najdete na enem od teh dveh URL-jev:

- <http://www.math.psu.edu/bri/>
- <http://blah.math.tu-graz.ac.at/~bri/>

Čeprav poskušam biti po pošti pozoren kot se le da, se, prosim, zavedajte, da dobim vsak dan veliko pisem, zato lahko traja dolgo, preden vam odgovorim. Posebno, kadar me po pošti kaj sprašujete, prosim, poskusite biti še posebej jasni in podrobni v svojem sporočilu. Če pišete o nedelujoči strojni opremi (ali kaj takega), moram vedeti, kakšna je vaša celotna strojna konfiguracija. Če poročate o napaki, ne recite le „Poskusil sem tole, pa mi je javil napako“; vedeti moram tudi, katera napaka je to bila. Želim tudi vedeti različico jedra, prevajalnika gcc in knjižnice libc, ki jih uporabljate. Če le poveste, da uporabljate to-in-to distribucijo, mi s tem ne boste povedali kaj dosti. Ne moti me, če vprašujete preprosta vprašanja; vedite, če nikoli ne vprašate, morda ne boste nikoli dobili odgovora! Želim se zahvaliti vsem, ki so mi posredovali povratne informacije.

Če vaše vprašanje ni povezano z jedrom ali je v jeziku, ki ga ne razumem, morda ne bom odgovoril.

Če ste mi pisali in vam nisem odgovoril v razumnem časovnem roku (trije tedni ali več), sem morda pomotoma pobrisal vaše sporočilo ali kaj takega (oprostite). Prosim, poskusite še enkrat.

Dobivam veliko pošte o stvareh, ki imajo pravzaprav opraviti s strojno opremo. To je v redu, a, prosim, zavedajte se, da nisem seznanjen z vso obstoječo strojno opremo tega sveta. Osebno uporabljam procesorje AMD, krmilnike SCSI proizvajalcev Adaptec in Sybios, ter diske SCSI proizvajalca IBM.

Različica -0.1 angleškega izvirnika je bila napisana 3. oktobra 1994. Izvirnik je dostopen kot SGML, PostScript, TeX, roff, in kot navaden tekst.

Avtor slovenskega prevoda z dne 24. julija 1999 je Roman Maurer <*roman.maurer@hermes.si*>. Prosim, pošljite mi pripombe na prevod. Slovenski prevod je dostopen kot SGML DTD LinuxDoc, HTML, DVI, PDF, PostScript in navaden tekst na strežniku slovenskega Društva uporabnikov Linuxa v imeniku <ftp://ftp.lugos.si/pub/lugos/doc/HOWTO-sl/> ali na spletnem naslovu <http://www.lugos.si/delo/slo/HOWTO-sl/Kernel-HOWTO-sl.html>.

12.2 Narediti

Razdelek 10 („Nasveti in triki“) je bolj majhen. Upam, da ga bom razširil s predlogi drugih.

Tako je tudi z razdelkom 6 („Dodatni paketi“).

Potrebujemo več podatkov o razhroščevanju/odpravljanju posledic sesutja sistema.

12.3 Prispevki

Vključen je majhen del Linusove datoteke *README* (izbire za hekiranje jedra). (Hvala, Linus!)

- *uc@brian.lunetix.de* (Ulrich Callmeier): `patch -s` in `xargs`,
- *quinlan@yggdrasil.com* (Daniel Quinlan): popravki in dodatki več razdelkov,
- *nat@nataa.fr.eu.org* (Nat Makarevitch): `mrproper`, `tar -p`, več drugih reči,
- *boldt@math.ucsb.edu* (Axel Boldt): po omrežju je zbral opise konfiguracijskih izbir jedra; potem mi je poslal seznam,
- *lembark@wrkhors.psyber.com* (Steve Lembark): predlog različnega zaganjanja,
- *kbriggs@earwax.pd.uwa.edu.au* (Keith Briggs): nekateri popravki in predlogi,
- *rmcguire@freenet.columbus.oh.us* (Ryan McGuire): dodatki ciljev `make`,

- *dumas@excalibur.ibp.fr* (Eric Dumas): francoski prevod,
- *simazaki@ab11.yamanashi.ac.jp* (Yasutada Shimazaki): japonski prevod,
- *jjamor@lml.ls.fi.upm.es* (Juan Jose Amor Iglesias): španski prevod,
- *mva@sbbs.se* (Martin Wahlen): švedski prevod,
- *jzp1218@stud.u-szeged.hu* (Zoltan Vamosi): madžarski prevod,
- *bart@mat.uni.torun.pl* (Bartosz Maruszewski): poljski prevod,
- *roman.maurer@hermes.si* (Roman Maurer): slovenski prevod,
- *donahue@tiber.nist.gov* (Michael J. Donahue): tipkarske napake, zmagovalec „tekmovanja narezanega kruha",
- *rms@gnu.ai.mit.edu* (Richard Stallman): zamisel in distribucija „proste" dokumentacije,
- *dak@Pool.Informatik.RWTH-Aachen.DE* (David Kastrup): reč o NFS,
- *esr@snark.thyrsus.com* (Eric Raymond): različni delčki.

Pomagali so mi tudi ljudje, ki so mi poslali pošto z vprašanji in problemi.

12.4 Pravice razširjanja, licenca, in te stvari

Priporočila: Komercialno razširjanje je dovoljeno in celo zaželeno; vendar se močno priporoča, da distributer stopi v stik z avtorjem še pred distribucijo, da bi obdržali osveženo stanje stvari (lahko mi pošljete tudi kopijo stvari, ki jo izdelujete, če ste že pri tem). Avtor svetuje tudi prevajalcem, da stopijo v stik z njim, preden začnejo prevajati. Natisnjene različice so videti bolje. Lahko jih recikliramo.