De Linux Tips HOWTO

Paul Anderson, paul@geeky1.ebtech.net,
Vertaald door: Ellen Bokhorst, bokkie@nl.linux.org

v3.6, juni 1998

Deze HOWTO bevat die moeilijk te vinden aanwijzingen en fijnafstemmingen die Linux er wat fraaier op maken.

Inhoudsopgave

1 Introductie

Welkom bij de **Linux Tips HOWTO**, een lijst met handige truuks en optimalisaties die Linux er leuker op maken. Alles wat hier nu in staat, zijn tips die ik uit mijn hoofd ken, en tips vanuit mijn oude Tips-HOWTO (Waarom zou je er fatsoenlijke tips uithalen, nietwaar?). Dus stuur al je favoriete tips naar me op, zodat ik ze in de volgende Tips-HOWTO kan plaatsen.

Paul Anderson *Maintainer–Linux TIPS HOWTO*

`panderso@ebtech.net`

2 Kleine Tips

2.1 Handige Syslog Truuk *Paul Anderson, Tips-HOWTO maintainer*

Wijzig je /etc/syslog.conf, en plaats daarin de volgende regel:

```
# Dump alles naar tty8
*.*                                     /dev/tty8
```

Een voorbehoud: *DENK ERAAN TABS TE GEBRUIKEN!* syslog houdt niet van spaties...

2.2 Script om die gecomprimeerde HOWTO's te bekijken. *Didier Juges,* `dj@destin.nfds.net`

Voor de ene aan de andere newbie is hier een klein script die het zoeken naar en het bekijken van howto documenten vereenvoudigt. Mijn howto's staan in /usr/doc/faq/howto/ en zijn met gzip gecomprimeerd. De bestandsnamen zijn XXX-HOWTO.gz, met XXX als het onderwerp. Ik maakte het volgende script aan met de naam "howtoïn de directory /usr/local/sbin:

```
#!/bin/sh
if [ "$1" = "" ]; then
    ls /usr/doc/faq/howto | less
else
    gunzip -c /usr/doc/faq/howto/$1-HOWTO.gz | less
fi
```

Wanneer aangeroepen zonder argument, toont het een directory met de beschikbare howto's. Wanneer vervolgens ingevoerd met het eerste deel van de bestandsnaam (voor het koppelteken) als een argument, toont het dan het gedecomprimeerde bestand (waarbij het origineel intact blijft).

Om bijvoorbeeld de Serial-HOWTO.gz te bekijken, typ je:

$ howto Serial

2.3 Is er voldoende vrije ruimte??? *Hans Zoebelein,* `zocki@goldfish.cube.net`

Dit is een klein script waarmee van tijd tot tijd wordt gecontroleerd of er voldoende vrije ruimte beschikbaar is op alles wat mount laat zien (disks, cdrom, diskette...)

Als de ruimte opraakt, wordt iedere X seconden op het scherm een melding weergegeven en 1 mailbericht per gevuld device afgevuurd.

```
#!/bin/sh

#
# $Id: Tips-HOWTO-NL.sgml,v 1.3 2003/11/22 14:36:34 bokkie Exp $
#

#
# Sinds ik tijdens het compileren mysterieuze foutmeldingen kreeg toen
# tmp bestanden mijn disks opvulden, schreef ik dit om een waarschuwing te
# krijgen voordat de disks vol zijn.
#
# Als hiermee werd voorkomen dat je servers explodeerde
# stuur dan een lovende email naar zocki@goldfish.cube.net.
# Als je site hierdoor afbrandt, dan sorry, maar ik heb je
# gewaarschuwd: geen klachten.
# Vergeef me alsjeblieft als je echt weet hoe met sed om te gaan :)
#

#
# Alle gekheid op een stokje: Plaats 'check_hdspace &' in rc.local.
# Controleer iedere $SLEEPTIME sec. op vrije ruimte op devices.
# Je zou zelfs je diskettes of tape drives erop kunnen controleren. :)
# Als de vrije ruimte onder de $MINFREE (kb) komt, zal er een waarschuwing
# op het scherm weerkaatsen en voor elke device waarop een tekort aan
# ruimte is geconstateerd een mail worden gestuurd aan $MAIL_TO_ME.
# Als er weer meer vrije ruimte is dan de limiet, gaat het ook weer gepaard
# met een mailactie.

# TEDOEN: Verschillende $MINFREE voor elk device.
# Bevrijd /*tmp dirs veilig van oude rommel als er geen vrije ruimte meer is.
```

```
DEVICES='/dev/sda2 /dev/sda8 /dev/sda9'         # device; hier plaats je disks
MINFREE=20480                                   # kb; hieronder een waarschuwing
SLEEPTIME=10                                    # sec; pauze tussen controles
MAIL_TO_ME='root@localhost'                     # dwaas; aan wie de waarschuwingsmail


# ------- geen wijzigingen nodig onder deze regel (hopelijk :) -------

MINMB=0
ISFREE=0
MAILED=""
let MINMB=$MINFREE/1024         # ja, we zijn strict :)

while [ 1 ]; do
        DF="`/bin/df`"
                for DEVICE in $DEVICES ; do
                ISFREE=`echo $DF | sed s#.\*$DEVICE" "\*[0-9]\*""\*[0-9]\*" "\*## | sed s#" ".\*##`

                if [ $ISFREE -le $MINFREE ] ; then
                        let ISMB=$ISFREE/1024
                        echo  "WAARSCHUWING: Slechts $ISMB vrij op $DEVICE." >&2
                        #echo "meer code/tekst hier plaatsen" >&2
                        echo -e "\a\a\a\a"

                        if [ -z  "`echo $MAILED | grep -w $DEVICE`" ] ; then
                                echo "WAARSCHUWING: Slechts $ISMB vrij op $DEVICE.      (Trigger is ingesteld op $MINMB mb)" \
                                | mail -s "WAARSCHUWING: Slechts $ISMB vrij op $DEVICE!" $MAIL_TO_ME
                                MAILEDH="$MAILED $DEVICE"
                                MAILED=$MAILEDH
                                # plaats verdere acties, zoals opschonen van
                                # */tmp dirs hier...
                        fi
                        elif [ -n  "`echo $MAILED | grep -w $DEVICE`" ] ; then
                                # Verwijder mailed markering als er weer voldoende
                                # diskruimte is. Zodat we klaar staan voor nieuwe
                                # mailactie.
                                MAILEDH="`echo $MAILED  | sed s#$DEVICE##`"
                                MAILED=$MAILEDH
                        fi

                done
                sleep $SLEEPTIME

done
```

2.4 Util om je logbestanden op te schonen. *Paul Anderson, Tips-HOWTO Maintainer>*

Ben je net als ik, dan heb je een lijst met 430 subscribers, plus 100+ berichten per dag die via UUCP binnenkomen. Wat moet een hacker met zulke grote logs? Installeer chklogs, dat is wat je kunt doen. Chklogs is geschreven door Emilio Grimaldo, `grimaldo@panama.iaehv.nl`, en de huidige versie 1.8 is beschikbaar vanaf ftp.iaehv.nl:/pub/users/grimaldo/chklogs-1.8.tar.gz. Het is tamelijk eenvoudig te installeren (je zult natuurlijk de info in de doc subdirectory erop nazien). Zodra je het hebt geïnstalleerd, voeg je als volgt een crontab record in:

```
# Voer dagelijks chklogs uit om 9:00PM.
00 21 * * *       /usr/local/sbin/chklogs -m
```

2.5 Handig script om core bestanden op te schonen *Otto Hammersmith,*ohammers@cu-online.com

Maak een bestand aan met de naam rmcores (de auteur noemt het handle-cores) met daarin het volgende:

```
#!/bin/sh
USAGE="$0 <directory> <message-file>"

if [ $# != 2 ] ; then
        echo $USAGE
        exit
fi

 echo Aan het verwijderen...
find $1 -name core -atime 7 -print -type f -exec rm {} \;

echo e-mailen
for name in `find $1 -name core -exec ls -l {} \; | cut -c16-24`
do
        echo $name
        cat $2 | mail $name
done
```

En laat het middels een cron job zeer vaak uitvoeren.

2.6 Directory's van het ene naar het andere bestandssysteem verplaatsen *Alan Cox,*A.Cox@swansea.ac.uk

Snelle manier om een gehele structuur met bestanden van de ene naar de andere disk te verplaatsen

```
(cd /source/directory && tar cf - . ) | (cd /dest/directory && tar xvfp -)
```

[Wijziging van cd /source/directory; tar....enz. ter voorkoming van een ramp waarmee de directory mogelijk wordt verwijderd. Met dank aan Jim Dennis, jim@starshine.org, dat hij me dit liet weten. -Maint.]

2.7 Uitzoeken wat de grootste directory's zijn. *Mick Ghazey,* mick@lowdown.com

Je ooit afgevraagd welke directory's het grootst zijn op je computer? Zo kom je daar achter.

```
du -S | sort -n
```

2.8 De Linux Gazette

Eer komt John Fisk toe, oprichter van de Linux Gazette. Dit is een uitstekend e-zine en het is **GRATIS!!!** Wat valt er meer te wensen? Bekijk het op:

```
http://www.linuxgazette.com
```

BTW, Het schijnt dat (1) LG nu per maand uitkomt, en (2) John Fisk het niet langer onderhoudt, de lui bij SSC doen dit.

2.9 Verwijzer naar patch voor GNU Make 3.70 om het functioneren van VPATH te wijzigen. *Ted Stern,* `stern@amath.washington.edu`

Ik weet niet of veel mensen dit probleem hebben, maar er is een "faciliteit"van GNU make versie 3.70 die ik niet prettig vind. Het gaat erom dat VPATH zich raar gedraagt als je het een absolute padnaam opgeeft. Er is een uiterst degelijke patch waarmee dit wordt gecorrigeerd, die je kunt krijgen van Paul D. Smith <`psmith@wellfleet.com`>. Hij post de documentatie en patch ook na elke revisie van GNU make in de nieuwsgroep "gnu.utils.bug" Over het algemeen pas ik deze patch toe en hercompileer gmake op elk systeem waarop ik toegang heb.

2.10 Hoe stop ik mijn systeem dat het fsck uitvoert bij elke reboot? *Dale Lutz,* `dal@wimsey.com`

V: Hoe stop ik e2fsck dat het mijn disk elke keer bij het booten controleert.

A: Wanneer je de kernel opnieuw bouwt, wordt het bestandssysteem als 'dirty' gemarkeerd en dus zal je disk bij elke boot worden gecontroleerd. Je kunt dit corrigeren door het opstarten van:

rdev -R /zImage 1

Dit corrigeert de kernel zodat het er niet langer van overtuigd is dat het bestandssysteem 'dirty' is.

Noot: Voeg als je lilo gebruikt `read-only` toe *aan je linux setup in je lilo config bestand (gewoonlijk /etc/lilo.conf)*

2.11 Hoe fsck's te voorkomen door een "device busy"tijdens het booten. *Jon Tombs,* `jon@gtex02.us.es`

Als je vaak device busy fouten krijgt bij een shutdown die veroorzaken dat op het bestandssysteem bij een reboot een fsck moet worden toegepast, dan is hier een eenvoudige correctie:

Voeg aan `/etc/rc.d/init.d/halt` of `/etc/rc.d/rc.0` de regel

```
mount -o remount,ro /mount.dir
```

toe voor alle gemounte bestandssystemen, behalve voor /, voor de aanroep naar umount -a. Dit betekent dat als het een shutdown om de een of andere reden niet lukt alle processen te killen en de disks te unmounten, ze bij een reboot toch clean zullen zijn. Het bespaart mij bij een reboot een heleboel tijd.

2.12 Hoe de grootste bestanden op je harddisk te vinden. *Simon Amor,* `simon@foobar.co.uk`

```
ls -l | sort +4n
```

Of voor degenen voor wie de ruimte er werkelijk op aankomt duurt dit wel even, maar het werkt geweldig:

```
cd /
ls -lR | sort +4n
```

2.13 Hoe druk ik pagina's af met een marge voor perforatiegaten. *Mike Dickey,* mdickey@thorplus.lib.purdue.edu

```
#!/bin/sh
# /usr/local/bin/print
# een eenvoudige opgemaakte afdruk, om het iemand mogelijk te maken
# 3 gaten in de uitvoer te ponsen en het samen te binden

cat $1 | pr -t -o 5 -w 85 | lpr
```

2.14 Een manier om bestandsstructuren te doorzoeken op een bepaalde reguliere expressie. *Raul Deluth Miller,* rockwell@nova.umd.edu

Ik noem dit script 'forall'. Gebruik het als volgt:

```
forall /usr/include grep -i ioctl
forall /usr/man grep ioctl
```

Hier is forall:

```
#!/bin/sh
if [ 1 = `expr 2 \> $#` ]
then
        echo Gebruik: $0 dir cmd [optargs]
        exit 1
fi
dir=$1
shift
find $dir -type f -print | xargs "$@"
```

2.15 Een script voor opschonen nadat programma's autosave en backup bestanden hebben aangemaakt *Barry Tolnas,* tolnas@nestor.engr.utk.edu

Hier is een simpel tweeregelig script waarmee directory's worden afgedaald om emacs auto-save (#) en backup (~) bestanden, .o bestanden en TeX .log bestanden te verwijderen. Het comprimeert tevens .tex bestanden en README bestanden. Ik noem het op mijn systeem 'squeeze'.

```
#!/bin/sh
#SQUEEZE verwijdert onnodige bestanden en comprimeert .tex en README bestanden
#Door Barry tolnas, tolnas@sun1.engr.utk.edu
#
echo squeezing $PWD
find  $PWD \( -name \*~ -or -name \*.o -or -name \*.log -or -name \*\#\) -exec
rm -f {} \;
find $PWD \( -name \*.tex -or -name \*README\* -or -name \*readme\* \) -exec gzip -9 {} \;
```

2.16 Hoe kom ik erachter welke processen het meeste geheugen in beslag nemen. *Simon Amor,* simon@foobar.co.uk

```
ps -aux | sort +4n
```

-OF-

```
ps -aux | sort +5n
```

2.17 Vi optuigen voor het programmeren in C, *Paul Anderson*,`Tips-HOWTO Maintainer`

Ik programmeer in mijn vrije tijd nogal wat in C en ik heb er de tijd voor genomen vi op te tuigen zodanig dat het C vriendelijk is. Hier is mijn .exrc:

```
set autoindent
set shiftwidth=4
set backspace=2
set ruler
```

Wat doet het? autoindent zorgt dat vi automatisch elke regel volgend op de eerste inspringt, shiftwidth stelt de afstand van ^T in op 4 spaties, backspace stelt de backspace modus in, en ruler zorgt dat het regelnummer wordt weergegeven. Denk eraan, om naar een specifiek regelnummer te gaan, stel 20, gebruik je:

```
vi +20 myfile.c
```

2.18 Gebruik ctags om het programmeren te vereenvoudigen

De meeste hackers hebben ctags reeds op hun computers, maar gebruiken het niet. Het kan erg handig zijn voor het wijzigen van specifieke functies. Stel dat je een functie hebt in één van de vele bronbestanden in een directory voor een programma dat je aan het schrijven bent, en je wilt deze functie vanwege updates wijzigen. We zullen deze functie foo() noemen. Je weet niet waar het zich in het bronbestand bevindt. Hier komt ctags om de hoek kijken. Wanneer het wordt uitgevoerd, produceert ctags een bestand met de naam tags in de huidige dir, wat uit een opsomming bestaat met alle functies, in welke bestanden deze zich bevinden en waar in het bestand. Het bestand tags ziet er ongeveer zo uit:

```
ActiveIconManager       iconmgr.c       /^void ActiveIconManager(active)$/
AddDefaultBindings      add_window.c    /^AddDefaultBindings ()$/
AddEndResize    resize.c        /^AddEndResize(tmp_win)$/
AddFuncButton   menus.c /^Bool AddFuncButton (num, cont, mods, func, menu, item)$/
AddFuncKey      menus.c /^Bool AddFuncKey (name, cont, mods, func, menu, win_name, action)$/
AddIconManager  iconmgr.c       /^WList *AddIconManager(tmp_win)$/
AddIconRegion   icons.c /^AddIconRegion(geom, grav1, grav2, stepx, stepy)$/
AddStartResize  resize.c        /^AddStartResize(tmp_win, x, y, w, h)$/
AddToClientsList        workmgr.c       /^void AddToClientsList (workspace, client)$/
AddToList       list.c  /^AddToList(list_head, name, ptr)$/
```

Om bijvoorbeeld AddEndResize() met vim te wijzigen, geef je op:

```
vim -t AddEndResize
```

Hierdoor verschijnt het van toepassing zijnde bestand in de editor, met de cursor aan het begin van de functie.

2.19 Waarom hangt sendmail 5 minuten bij het opstarten onder RedHat? *Paul Anderson,* `paul@geeky1.ebtech.net`

Dit is een tamelijk algemeen probleem. Ik weet niet of RedHat deze bug al in hun distributie heeft gecorrigeerd, maar je kunt het zelf repareren. Als je in het bestand /etc/hosts kijkt, zul je zien dat het er ongeveer zo uitziet:

```
127.0.0.1               localhost       jebox
```

Wanneer sendmail start, zoekt het je hostnaam op (in dit voorbeeld, jebox). Het bemerkt dan dat het IP van jebox 127.0.0.1 is. Sendmail heeft hier problemen mee, dus voert het de zoekopdracht nogmaals uit. Het gaat hier een tijdje mee verder totdat het uiteindelijk opgeeft en stopt. Het corrigeren van het probleem is zeer eenvoudig. Wijzig het bestand /etc/hosts zodat het er ongeveer zo uit komt te zien:

```
127.0.0.1               localhost
10.56.142.1             jebox
```

2.20 Hoe configureer ik RedHat voor gebruik van color-ls? *Paul Anderson,* paul@geeky1.ebtech.net

De distributie van RedHat wordt geleverd met color-ls. Waarom ze het echter standaard niet voor kleurengebruik configureren, is me een raadsel. Zo kun je het corrigeren.

Typ als eerste: eval 'DIRCOLORS'

Vervolgens, alias ls='ls –color=auto'

En plaats de 'alias.....' in /etc/bashrc

2.21 Hoe kom ik erachter welke library in /usr/lib een bepaalde functie bevat? *Pawel Veselow,* vps@unicorn.niimm.spb.su

Wat als je aan het compileren bent en je hebt een library gemist die moet worden gelinkt? Alle verslagen van gcc bestaan uit functienamen... Hier is een simpele opdracht dat zal vinden waarnaar je op zoek bent:

```
for i in *; do echo $i:;nm $i|grep tgetnum 2>/dev/null;done
```

tgetnum is hier de naam van de functie waar je naar op zoek bent.

2.22 Ik compileerde een klein testprogramma in C, maar toen ik het uit probeerde te voeren, kreeg ik geen uitvoer!

Je hebt het programma waarschijnlijk in een binary met de naam test gecompileerd, nietwaar? Linux heeft een programma met de naam test, die test of een bepaalde conditie waar is, het produceert nooit enige uitvoer op het scherm. Probeer in plaats van gewoon test, het intikken van: ./test

3 Gedetailleerde Tips

3.1 Delen van swappartities tussen Linux en Windows. *Tony Acero,* ace3@midway.uchicago.edu

1. Formatteer de partitie als een dospartitie, en creëer er het Windows swapbestand op, maar draai windows nog niet. (Je wilt het swapbestand op het moment nog leeghouden, zodat het goed comprimeert).

2. Boot linux en bewaar de partitie in een bestand. Als de partitie bijvoorbeeld /dev/hda8 is:

   ```
   dd if=/dev/hda8 of=/etc/dosswap
   ```

3. Comprimeer het dosswap bestand; aangezien het praktisch allen nullen zijn, zal het zeer goed comprimeren

```
gzip -9 /etc/dosswap
```

4. Voeg het volgende toe aan het /etc/rc bestand om de swapspace onder Linux voor te bereiden en te installeren:

   XXXXX is het aantal blokken in de swappartitie

   ```
   mkswap /dev/hda8 XXXXX
   swapon -av
   ```

   Zorg dat je een regel toevoegt in het /etc/fstab bestand voor de swappartitie

5. Als het package init/reboot /etc/brc of /sbin/brc ondersteunt, voeg je het volgende toe aan /etc/brc, doe dit anders met de hand, wanneer je in dos|os/2 boot en je de swappartitie weer wilt omzetten naar de dos/windows versie:

   ```
   swapoff -av
   zcat /etc/dosswap.gz | dd of=/dev/hda8 bs=1k count=100
   ```

Merk op dat hiermee slechts de eerste 100 blokken naar de partitie worden # teruggeschreven. # Ik heb door ervaring gemerkt dat dit voldoende is

> Wat zijn hier de voors en tegens van?

Voors: je bespaart een substantiële hoeveelheid schijfruimte

Tegens: Als stap 5 niet automatisch gaat, dan moet je eraan denken dit met de hand te doen, en het vertraagt het rebootproces met een nanoseconde :-)

3.2 Wanhopige Undelete. *Michael Hamilton,* `michael@actrix.gen.nz`

Hier is een truuk die ik al een paar keer heb moeten gebruiken

Als je per ongeluk een tekstbestand verwijdert, zoals bijvoorbeeld wat email, of het resultaat van een programmeersessie op de late avond, hoeft alles niet verloren te zijn. Als het bestand het ooit naar disk haalde, d.w.z. dat het daar meer dan 30 seconden was, dan kan het zijn dat de inhoud nog steeds op de diskpartitie voorkomt.

Je kunt de opdracht grep gebruiken om de ruwe diskpartitie te doorzoeken op de inhoud van het bestand.

Ik verwijderde bijvoorbeeld onlangs per ongeluk een deel van m'n email. Dus staakte ik onmiddellijk mijn activiteiten die deze partitie konden wijzigen: in dit geval zag ik gewoon af van het opslaan van mijn bestanden of het uitvoeren van compilaties, enz. Onder andere omstandigheden heb ik me de moeilijkheid op de hals gehaald door het systeem in single user modus te brengen en het bestandssysteem te unmounten.

Ik paste toen de opdracht egrep toe op de diskpartitie: in mijn geval bevond het emailbericht zich in /usr/local/home/michael/, dus aan de uitvoer van df, kon ik zien dat dit op /dev/hdb5 was.

```
sputnik3:~ % df
  Filesystem         1024-blocks  Used Available Capacity Mounted on
  /dev/hda3              18621    9759     7901     55%   /
  /dev/hdb3             308852  258443    34458     88%   /usr
  /dev/hdb5             466896  407062    35720     92%   /usr/local

  sputnik3:~ % su
  Password:
  [michael@sputnik3 michael]# egrep -50 'ftp.+COL' /dev/hdb5 > /tmp/x
```

Nu ben ik extreem voorzichtig wanneer ik met diskpartities aan de gang ga, dus ik pauzeerde om er zeker van te zijn dat ik de syntax van de opdracht begreep VOORDAT ik de return indrukte. In dit geval bevatte de email het woord 'ftp' gevolgd door wat tekst gevolgd door het woord 'COL'. Het bericht bestond uit ongeveer 20 regels, dus gebruikte ik -50 om alle regels rondom de woorden te krijgen. Voorheen gebruikte ik altijd -3000 om er zeker van te zijn dat ik alle regels kreeg van een of andere broncode. Ik stuurde de uitvoer van egrep door naar een andere diskpartitie. Hiermee voorkwam ik dat er over het bericht heengeschreven zou worden waar ik naar aan het zoeken was.

Vervolgens gebruikte ik strings om me te helpen de uitvoer te inspecteren.

```
strings /tmp/x | less
```

Zeker weten dat de email zich daarin bevond.

Deze methode is niet betrouwbaar; alle of een deel van de schijfruimte kan reeds zijn hergebruikt.

Deze truuk is waarschijnlijk alleen bruikbaar op single user systemen. Op multi-user systemen met nogal wat diskactiviteit, kan de ruimte die je hebt vrijgemaakt reeds weer zijn gebruikt. En de meesten van ons kunnen niet zomaar de box vandaan trekken bij onze gebruikers wanneer we ooit een bestand moeten herstellen.

Op mijn systeem thuis is deze truuk me in de afgelopen paar jaar bij ongeveer drie gelegenheden van pas gekomen - gewoonlijk wanneer ik per ongeluk wat van het werk van die dag verwijderde. Als waar ik aan werk het overleeft tot een punt waarvan ik het gevoel heb dat ik een belangrijke voortgang hebt geboekt, wordt er op een diskette een backup van gemaakt, dus ik heb deze truuk nog niet zo vaak nodig gehad.

3.3 Hoe gebruik te maken van de immutable vlag. *Jim Dennis,* `jadestar@rahul.net`

Gebruik de Immutable Flag

Neem direct na de installatie en configuratie van je systeem de /bin, /sbin, /usr/bin, /usr/sbin en /usr/lib door (plus nog een paar van de andere gebruikelijke verdachte bestanden en maak royaal gebruik van de opdracht 'chattr +i '. Voeg dat ook toe aan de kernelbestanden in root. Nu 'mkdir /etc/.dist/' kopieer alles vanuit /etc/ naar beneden (Ik doe dit in twee stappen met /tmp/etcdist.tar ter voorkoming van recursie) in die directory. (Optioneel kun je gewoon /etc/.dist.tar.gz aanmaken) – en dat als immutable markeren.

De reden voor dit alles is het beperken van de schade die je als root kunt aanrichten. Je zal geen bestanden overschrijven door een misplaatst omleidingsteken, en je zal het systeem niet onbruikbaar achterlaten door een verdwaalde spatie in een 'rm -fr' opdracht (je kunt nog steeds heel wat schade aan je gegevens aanrichten – maar je libs en bins zullen veiliger zijn.

Dit maakt ook een diversiteit aan beveiligings- en denial of service uitbuitingen óf onmogelijk óf moeilijker (aangezien veel daarvan erop vertrouwen een bestand te kunnen overschrijven via de acties van een of ander SUID programma die *niet voorziet in een willekeurige shellopdracht*).

Het enige ongerief hierbij onstaat bij het bouwen en uitvoeren van een 'make install' op diverse soorten systeembinary's. Aan de andere kant voorkomt het ook dat een 'make install' de bestanden overschrijft. Wanneer je vergeet de Makefile in te lezen en chattr -i toe te passen op de bestanden die op het punt staan te worden overschreven (en de directory's waaraan je bestanden toe wilt voegen) – mislukt de make. Je past er dan gewoon de chattr opdracht op toe en start het opnieuw op. Je kunt die gelegenheid ook gebruiken om je oude bin's, libs of wat dan ook naar een .old/ directory te verplaatsen of ze hernoemen of er tar op toepassen of wat dan ook.

3.4 Een suggestie waar nieuwe software te plaatsen. *Jim Dennis,* jadestar@rahul.net

Alle nieuwe software begint onder /usr/local! of /usr/local/‘hostname‘

Als je distributie /usr/local leeg laat, creëer dan een /usr/local/src, /usr/local/bin enz en gebruik dat. Plaatst de distributie zaken in de /usr/local structuur dan wil je wellicht een ’mkdir /usr/local/‘hostname‘’ uit laten voeren en er de groep ’wheel’ +w aan toekennen (ik maak het ook SUID en SGID om ervan verzekerd te zijn dat elk lid van de wheel groep daaronder alleen iets met eigen bestanden kan doen, en dat alle aangemaakte bestanden zullen toebehoren aan de ’wheel’ groep.

Disciplineer jezelf nu om *ALTIJD! ALTIJD! ALTIJD!* nieuwe packages onder /usr/local/src/.from/$WAAR_IK_HET_VANDAAN_HAALDE/ plaatst (voor de .tar of wat voor bestanden dan ook) en bouw ze onder /usr/local/src (of .../$HOSTNAME/src). Zorg dat ze onder de lokale hiërarchie worden geïnstalleerd. Plaats een symlink vanuit de lokale hiërarchie naar elk element dat ergens anders naartoe gaat als het *beslist moet"worden geïnstalleerd in /bin, /usr/bin of elders.

De reden hiervoor – ook als is het wat meer werk – is dat het helpt isoleren waarvan een backup moeten worden gemaakt en wat moet worden terruggezet van een backup of opnieuw geïnstalleerd in geval van een volledige herinstallatie vanaf de distributiemedia (tegenwoordig gewoonlijk van een CD). Door gebruik te maken van een /usr/local/.from directory houd je ook een informele log bij van waar je bronnen vandaan komen. – wat helpt wanneer je op zoek bent naar nieuwe updates – en van groot belang kan zijn bij het monitoren van de security announcement lists.

Een van mijn systemen thuis werd samengesteld voordat ik deze maatregelen zelf toepaste. Ik heb nog maar erg weinig met de configuratie van mijn thuissysteem gedaan en ik ben de *enige* persoon die het ooit gebruikt.

Als contrast zijn de systemen die ik op het werk heb ingesteld (toen mij hier de rol van systeembeheerder werd toevertrouwd) allen op deze manier ingesteld — beheerd door veel contractanten en andere MIS mensen, zijn er een groot aantal upgrades en installatie van packages op geïnstalleerd. Niettemin heb ik een zeer goede indruk welke elementen precies werden geplaatst *na* de initiële installatie en configuratie.

3.5 Alle bestanden in een directory naar kleine letters omzetten. *Justin Dossey,* dossey@ou.edu

Ik nam notitie van een paar overmatig moeilijke of onnodige procedures aanbevolen in de 2c tips sectie van Issue 12. Aangezien er meer van zijn, stuur ik het je op:

```
#!/bin/sh
         # lowerit
         # zet alle bestandsnamen in de huidige directory om naar kleine letters
         # werkt alleen met gewone bestanden--wijzigt geen directorynamen
         # zal vragen om verificatie voor een bestaand bestand te overschrijven
         for x in `ls`
           do
           if [ ! -f $x ]; then
             continue
             fi
           lc=`echo $x  | tr '[A-Z]' '[a-z]'`
           if [ $lc != $x ]; then
             mv -i $x $lc
           fi
           done
```

Wauw. Dat is een lang script. Ik zou daarvoor geen script schrijven, ik zou in plaats daarvan deze opdracht gebruiken:

```
for i in * ; do [ -f $i ] && mv -i $i `echo $i | tr '[A-Z]' '[a-z]'`;
done;
```

op de opdrachtregel.

Degene die het aanleverde zei dat de wijze waarop hij het script schreef hij dit voor de leesbaarheid deed (zie hieronder).

Op naar de volgende tip, deze over het toevoegen en verwijderen van gebruikers. Het gaat Geoff goed af tot aan de laatste stap. Reboot? Tjonge, ik hoop niet dat hij reboot elke keer als hij een gebruiker verwijdert. Het enige dat je hoeft te doen, is het uitvoeren van de eerste twee stappen. Welk type processen zou die gebruiker hebben lopen? Een irc bot? Het killen van de processen met een simpel

```
kill -9 `ps -aux |grep ^<username> |tr -s " " |cut -d " " -f2`
```

Voorbeeld, gebruikersnaam is foo

```
kill -9 `ps -aux |grep ^foo |tr -s " " |cut -d " " -f2`
```

Laten we daarmee te hebben afgerekend, verdergaan met het vergeten root-wachtwoord.

De oplossing gegeven in de Gazette is de meest universele, maar niet de eenvoudigste. Met zowel LILO als loadlin, kan met het opgeven van de bootparameter "single" direct in de standaardshell zonder login of password prompt worden geboot. Vanaf daar, kan met elk wachtwoord wijzigen of verwijderen voor het typen van "init 3" om in multiuser modus te starten. Aantal reboots: 1 De andere manier Aantal reboots: 2

Justin Dossey

3.6 Hoe Sendmail upgraden *Paul Anderson,* `paul@geeky1.ebtech.net`

We beginnen vanuit de ruwe, zuivere broncode. Zorg eerst dat je aan de sendmail broncode komt. Ik heb versie 8.9.0, wat zoals je op zal vallen, het nieuwste van het nieuwste is. Ik haalde het vanaf ftp.sendmail.org:/pub/sendmail/sendmail.8.9.0.tar.gz

Het is ongeveer 1Meg, en in overweging nemend dat ik 8.7.6 draai, denk ik dat het de moeite waard is. Als dit werkt, zul je dit ongetwijfeld te horen krijgen, anders kan ik de nieuwe HOWTO versies er niet uitkrijgen zonder e-mail:)

Pak het uit, nu je de broncode hebt. Er zal in de huidige directory een dir met de naam `sendmail-8.9.0` worden aangemaakt. Ga naar die directory en lees de bestanden README en RELEASE_NOTES (en verbaas je over de updates die zijn gedaan). Ga nu met cd naar src. Hier zal je meeste werk worden uitgevoerd.

Een beknopte notitie: Sendmail is een klein, krachtig en goed geschreven programma. De sendmail binary zelf compileert in minder dan 5 minuten op mijn 5x86 133 met 32Megs RAM! De gehele compilatie en installatie nam (zonder config) minder dan 15 minuten in beslag!

Normaal gesproken gebruik ik BIND niet op mijn systeem, dus ik trof de regels

```
# ifndef NAMED_BIND
#  define NAMED_BIND    1       /* gebruik Berkeley Internet Domain Server */
# endif
```

aan en wijzigde de 1 in een 0, ala:

```
# ifndef NAMED_BIND
#  define NAMED_BIND    0       /* gebruik Berkeley Internet Domain Server */
# endif
```

Onder Debian 1.3.1, is db.h standaard geïnstalleerd in /usr/include/db, in plaats van in /usr/include, waar sendmail het hoopt te vinden. Ga naar de src, mailstats, makemap, praliases, rmail en smrsh directory's en voer de volgende opdracht uit:

```
./Build -I/usr/include/db
```

Zodra je dat hebt gedaan, cd .. en typ make install. Dat is het! Sendmail versie 8.9.0 zou nu moeten zijn geïnstalleerd! Dit uiteraard in de veronderstelling dat je reeds een originele configuratie hebt. Om alles op mijn systeem soepel te laten werken, moest ik het volgende aan het begin van /etc/sendmail.cf toevoegen, aangezien ik vrije mailinglists host voor mensen die majordomo gebruiken:

```
O DontBlameSendmail=forwardfileinunsafedirpath, forwardfileinunsafedirpathsafe
```

Sendmail 8.9.0 is tegenwoordig nogal eigenzinnig als het gaat om directory- en bestandspermissies, en het zal meldingen geven over dirs en bestanden in aliases of .forward bestanden die voor de groep of wereld schrijfbaar zijn. Ondanks dat het niet verstandig is deze eigenzinnigheid te deactiveren, draai ik het als enige persoon op de console en ik vond dat het ok was dit kleine beveiligingsgat toe te staan. YMMV.

3.7 Een aantal tips voor nieuwe systeembeheerders. *Jim Dennis,* `jadestar@rahul.net`

Creëer en onderhoud een /README.'hostname' en/of een /etc/README.'hostname' *[Of mogelijk /usr/local/etc/README.'hostname' -Maint.]*

Maak vanaf *de eerste dag* dat je een systeem beheert, notities in een online logbestand. Je zou een "vi /README.$(hostname)" een regel in root's /bash_logout aan kunnen maken. Een andere manier om dit te doen is het schrijven van een su of sudo script die iets dergelijks doet als in:

```
function exit \
        { unset exit; exit; \
          cat ~/tmp/session.$(date +%y%m%d) \
          >> /README.$(hostname) && \
          vi /README.$(hostname)
          }
script -a ~/tmp/session.$(date +%y%m%d)
/bin/su.org -
```

(gebruik de opdracht typescript om een sessielog te creëren en maak een functie aan voor het automatisch toevoegen en bijwerken van de log).

Ik geef toe dat ik het automatiseren van dit beleid niet heb geïmplementeerd. Ik vertrouwde tot dusverre op zelfdiscipline. Ik heb echter met het idee gespeeld (zelfs tot aan het punt vooraf intypen van de scripts en shellfuncties zoals je ze hier ziet). Een ding dat me weerhoudt is de 'script' opdracht zelf. Ik denk dat ik een paar opdrachtregelparameters aan de broncode toe moet voegen (voor een pause/stop van het scriptopname vanaf de opdrachtregel) voordat ik ze aanlever voor gebruik.

Mijn laatste suggestie (voor deze ronde):

Het pad van root zou moeten bestaan uit 'PATH= /bin'

Dat is alles. Niets meer in het pad van root. Alles wat root doet wordt geleverd door een symlink vanuit /bin of door een alias of shellfunctie of is een script of binary in /bin, of wordt uitgetikt met een expliciet pad.

Dit maakt iedereen draaiend als root zich bewust (soms pijnlijk bewust) van hoe hij/zij binaire bestanden vertrouwt. De verstandige beheerder van een multi-user host zal periodiek zijn /bin en /.*history bestanden doorzoeken op bepaalde patronen en loopholes.

De echt gemotiveerde beheerder zal reeksen ontdekken die kunnen worden geautomatiseerd, plaatsen waar veiligheidscontroles kunnen worden ingevoegd, en taken waarvoor "root" privileges tijdelijk zouden moeten worden vermeden (opstarten van editors, MTA's en andere grote interactieve programma's met uitgebreide scriptmogelijkheden die in transparante of gegevensbestanden, *zouden* kunnen worden ingesloten zoals de befaamde vi ./.exrc en emacs ./.emacs en de zelfs meer verraderlijke $EXINIT en ingesloten header/footer macro's). Vanzelfspreken kunnen dergelijke opdrachten worden uitgevoerd met iets als:

```
cp $data $some_users_home/tmp
su -c $origcommand $whatever_switches
cp $some_users_home/tmp $data
```

(...waar de details afhangen van de opdracht).

De meeste van deze voorzorgsmaatregelen zijn voor de home- of voor een "single" user werkstation wat overdreven, maar vormen een erg goed beleid voor de beheerder van een multi-user systeem — in het bijzonder wordt een publiek toegankelijk systeem (zoals die van netcom).

3.8 Hoe xdm's chooser te configureren voor hostselectie *Arrigo Triulzi,* `a.triulzi@ic.ac.uk`

1. Wijzig het bestand waarmee xdm wordt opgestart, naar alle waarschijnlijkheid is dit /etc/rc/rc.6 of /etc/rc.local), zodanig dat in de xdm opstartsectie de volgende regels komen te staan:

```
/usr/bin/X11/xdm
exec /usr/bin/X11/X -indirect hostname
```

2. Wijzig /usr/lib/X11/xdm/Xservers en haal het commentaarteken weg voor de regel die de server start op de lokale machine (d.w.z. de regel beginnend met 0:)

3. Reboot de machine

Ik voeg dit toe omdat het me ongeveer een week kostte om alle problemen de kop in te drukken toen ik het wanhopig probeerde in te stellen voor mijn eigen subnet.

Voorbehoud: met oude SLS (1.1.1) kun je om een of andere reden een -nodaemon weglaten na de xdm regel – dit werkt **NIET** met latere uitgaven.