PATH HOWTO

Esa Turtiainen, etu@dna.fi Vertaald door: Ellen Bokhorst, bokkie@nl.linux.org v0.4, 15 november 1997

Inhoudsopgave

1 Introductie

Dit document beschrijft algemene truuks en problemen met Unix / Linux omgevingsvariabelen, vooral met de PATH variabele. PATH is een lijst met directory's waarin naar commando's wordt gezocht. De details gelden voor de Debian Linux 1.3 distributie.

Opmerking! Dit document is in beta release status. Stuur alsjeblieft commentaar en correcties op.

2 Copyright

3 Algemeen

Alle Unix processen bevatten een ömgeving". Dit is een lijst met variabelen waarin de naam en waarde staan, beide gewone strings die uit de meeste tekens kan zijn samengesteld. Alle Unix processen hebben een ouder proces - het proces dat dit kindproces aanmaakte. Kindprocessen erven de omgeving van de ouderprocessen. Ze kunnen een aantal wijzigingen in de omgeving aanbrengen voordat ze het doorgeven aan hun kindprocessen.

Een belangrijke omgevingsvariabele is PATH, een lijst met directory's gescheiden door dubbele punten (':'). Deze directory's worden doorzocht om commando's te kunnen vinden. Als je een commando 'foo' probeert aan te roepen, worden alle directory's in PATH (in die volgorde) doorzocht voor een uitvoerbaar bestand met de naam 'foo' (één met x-bit aan). Als het bestand wordt gevonden, wordt het uitgevoerd.

In deze howto gebruik ik de term 'commando' om te verwijzen naar uitvoerbare programma's, waarvan het de bedoeling is dat ze worden aangeroepen met verkorte namen, door gebruik te maken van het padmechanisme.

In Linux doorzoeken zelfs de low level besturingssysteem aanroepen (de exec familie van aanroepen) de directory's in het PATH variabele om processen te starten: je kunt het padmechanisme overal gebruiken daar waar je een commando probeert uit te voeren. Als de exec besturingssysteemaanroep een bestandsnaam krijgt die geen '/' bevat, bepaalt het de waarde van de PATH omgevingsvariabele. Zelfs als er geen variabele PATH in de omgeving voorkomt, worden tenminste de directory's /bin en /usr/bin doorzocht op passende commando's.

In sh gebruik je het export commando om de omgeving in te stellen, in csh gebruik je het setenv commando. Bijvoorbeeld:

sh:

```
PATH=/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games:.
```

csh:

```
setenv PATH /usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games:.
```

C-programma's kunnen de setenv() bibliotheekaanroep gebruiken om de omgeving te wijzigen. Perl heeft z'n omgeving in een daarmee verbonden array %ENV, je kunt PATH instellen als in PATH $ENV{PATH}=/ bin".

Het env commando is de basiswijze om de naar de huidige omgevingsvariabelen te vragen. Het kan ook worden gebruikt om wijzigingen aan te brengen.

Meer informatie over de basis omgevingsvariabelen kun je vinden in de manual pages van 'environ', 'execl', 'setenv', info file 'env' en in de shell-documentatie.

Als Linux opstart, is het initproces het eerste normale proces dat start. Het is een speciaal proces omdat het geen ouder heeft. Het is echter de stamvader van alle andere processen. De init omgeving zal als een omgeving van alle processen blijven als ze het niet expliciet wijzigen. De meeste processen doen dit echter wel.

Init start een groep processen. Het bestand /etc/inittab geeft de processen die het systeem start. Deze processen werken in de omgeving die direct van init is geërfd - - dit zijn typisch processen zoals 'getty', het programma dat 'login:' naar console schrijft. Als je hier PPP verbindingen start, moet je er aan denken dat je in de init omgeving werkt. De systeeminitialisatie is vaak een script dat hier is gestart. In Debian 1.3 is het initialisatiescript /etc/init.d/rc en het roept op zijn beurt andere initialisatiescripts aan.

Het systeem bevat veel draaiende servers (daemons) die wel of niet de standaardomgeving gebruiken. De meeste servers worden vanuit de initialisatiescripts gestart en hebben als gevolg daarvan de init omgeving.

Als een gebruiker op het systeem inlogt, wordt de omgeving beïnvloedt door de instellingen die in de programma's zijn gecompileerd, systeemomvattende initialisatiescripts en gebruikers-initialisatiescripts. Dit is nogal gecompliceerd en de huidige situatie is niet volledig bevredigend. Het is totaal anders als een gebruiker inlogt vanaf de tekstconsole, XDM of vanaf het netwerk.

4 Init

Init is een ouder-proces voor alle andere processen van het systeem. Andere processen erven de omgeving van het init proces en het pad is het init pad in het ongewone geval dat er geen ander pad is ingesteld.

Het 'init pad' staat vast in de bron van het init programma en het is:

```
/usr/local/sbin:/sbin:/bin:/usr/sbin:/usr/bin
```

Merk op dat /usr/local/bin niet in het init pad voorkomt.

Alle programma's die vanuit /etc/inittab zijn gestart werken in de init omgeving, vooral systeeminitialisatiescripts in /etc/init.d (Debian 1.3).

Alles dat vanuit initialisatiescripts wordt gestart heeft de init-omgeving als standaardomgeving. Bijvoorbeeld, syslogd, kerneld, pppd (als bij het opstarten gestart), gpm en het belangrijkst lpd en inetd hebben de init omgeving en veranderen die niet.

Er zijn programma's die vanuit opstartscripts worden opgestart maar waarbij de PATH omgevingsvariabele expliciet wordt ingesteld in het opstartscript. Voorbeelden hiervan zijn: atd, sendmail, apache en squid.

Er zijn andere programma's die vanuit opstartscripts worden gestart en het pad volledig wijzigen. Een dergelijk voorbeeld is cron.

5 Login

In tekstconsole wacht een getty programma op gebruikers login. Het schrijft 'login:' en andere meldingen. Het doet zijn werk in de init omgeving. Als getty een gebruiker krijgt die op het systeem inlogt, roept het de shell aan. Dit programma stelt de gebruikersomgeving in en roept de shell aan.

Het login programma stelt het pad in zoals is gedefinieerd in /usr/include/paths.h. het 'login pad' is voor root gebruikers en andere gebruikers anders.

voor algemene gebruikers (_PATH_DEFPATH):

```
/usr/local/bin:/usr/bin:/bin:.
```

voor root (_PATH_DEFPATH_ROOT):

```
/sbin:/bin:/usr/sbin:/usr/bin
```

Het algemene gebruikerspad bevat geen enkele sbin directory's. Hierin staat echter wel de huidige directory '.', die als gevaarlijk wordt aangemerkt voor de root gebruiker. Zelfs /usr/local/bin is niet beschikbaar voor de root gebruiker.

Het login pad wordt vaak door shell initialisatie overschreven. Het is echter mogelijk andere programma's dan gebruikersshells in /etc/passwd te plaatsen. Bijvoorbeeld, ik heb de volgende regel gebruikt om PPP op te starten als ik inlog door een speciale gebruiker te gebruiken. In dit geval, heeft pppd een exact loginpad.

```
etu-ppp:viYabVlxPwzDl:1000:1000:Esa Turtiainen, PPP:/:/usr/sbin/pppd
```

6 Shells

Vaak zijn gebruikersprocessen kindprocessen van de shell die in /etc/passwd voor deze gebruiker worden genoemd. In de shell-initialisatie bestanden wordt vaak het pad gewijzigd.

In login, wordt de naam van de shell voorafgegaan door een '-', bash wordt bijvoorbeeld '-bash' genoemd. De shell weet hierdoor dat het een 'login'-shell is. In dit geval voert de shell de 'login' initialisatie bestanden uit. Anders wordt er een lichtere initialisatie uitgevoerd. Bovendien controleert de shell of het interactief is, komen de commando's vanuit een bestand of interactieve tty. Dit wijzigt de shell-initialisatie zodanig dat een niet-interactieve niet-login shell zeer licht wordt geïnitialiseerd - bash voert in dit geval geen enkel initialisatiebestand uit!

6.1 bash

Als een normale login-shell haalt bash zijn 'bronnen' uit het systeemomvattende bestand /etc/profile waarin de systeemomgeving en het pad voor bash gebruikers kan worden ingesteld. Het wordt echter niet opgestart als het systeem de shell als een niet-interactieve shell interpreteert. Het belangrijkste geval is in rsh waar een remote commando op de naburige machine wordt uitgevoerd. De /etc/profile wordt niet opgestart en het pad wordt van de rsh daemon geërfd.

bash ontvangt de commando-regelargumenten -login en -i die respectievelijk worden gebruikt om de shell als een login shell of een interactieve shell in te stellen.

De gebruiker kan ingestelde waarden in /etc/profile overschrijven door een bestand ~/.bash_profile, ~/.bash_login of ~/.profile aan te maken. Merk op dat hiervan alleen het eerste bestand wordt uitgevoerd, dus verschilt van de logica van de csh initialisatie. ~/.bash_login wordt niet speciaal voor login shells uitgevoerd en als .bash_profile bestaat, wordt het helemaal niet uitgevoerdi!

Als bash met de naam sh wordt gebruikt in plaats van de naam bash, emuleert het de originele Bourne shell-initialisatie: het haalt zijn bronnen slechts uit de bestanden /etc/profile en ~/.profile en alleen maar voor login shells.

6.2 tcsh

Als een login shell voert tcsh de volgende bestanden in deze volgorde uit:

- /etc/csh.cshrc
- /etc/csh.login
- ~/.tcshrc
- ~/.cshrc (als .tcshrc niet wordt gevonden)
- ~/.history
- ~/.login
- ~/.cshdirs

tcsh kan zodanig worden gecompileerd dat het login scripts voor cshrc scripts uitvoert. Pas op!

Niet-interactieve shells voeren alleen de *cshrc scripts uit. *login scripts kunnen worden gebruikt om slechts éénmalig in de login het pad in te stellen.

7 Gebruikers ID wijzigen

7.1 su

Het commando su stelt een nieuwe te gebruiken gebruikers-id in. Als er geen gebruikers-id wordt opgegeven, wordt root gebruikt.

Normaal gesproken roept su een subshell met een andere gebruikers-id aan. Met het argument '-' (recenter synoniem -l of –login) roept su een shell aan zoals de login shell. Het gebruikt het login programma hier echter niet voor, maar het gebruikt nog een ander ingebouwd pad voor login 'simulatie' (term die wordt gebruikt in de broncode). Dit is:

voor gewone gebruikers

```
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:.
```

voor de root gebruiker

```
/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11:/usr/local/sbin:/usr/local/bin
```

su maakt ook heel wat subtiele omgevings wijzigingen.

7.2 sudo

Er is een groep commando's die veiliger gebruik maakt van de superuser commando's. Ze staan beter inloggen toe, op gebruikers gebaseerde beperkingen en het gebruik van individuele wachtwoorden. Sudo wordt het meest gebruikt.

```
$ sudo env
```

voert het commando env als superuser uit (als het zodanig is geconfigureerd dat dit is toegestaan).

het sudo commando heeft ook weer een andere aanpak voor padafhandeling. Het verandert het zoekpad zodanig dat de huidige directory altijd de laatste directory is. Het wijzigt echter niet de omgevingsvariabele PATH, slechts een paar omgevingsvariabelen zoals SUDO_USER.

8 Netwerk servers

De meeste netwerkservers zouden geen enkele subprocessen moeten aanroepen. Hun pad zou om beveiligingsredenen minimaal moeten zijn.

Een belangrijke uitzondering zijn alle diensten die het inloggen op het systeem vanuit een netwerk toestaan. Deze sectie beschrijft wat de omgeving is in deze situaties. Als het commando op de remote machine met rsh wordt uitgevoerd, krijgt het een ander pad dan wanneer het met ssh wordt uitgevoerd. Vergelijkbaar, het inloggen met rlogin, Telnet of ssh is anders.

8.1 inetd

De meeste netwerkservers hebben geen eigen processen die continue op verzoeken wachten. Dit werk is naar een Internet super server met de naam inetd gedelegeerd. Inetd luistert voor alle gedefinieerde netwerkpoorten en start de passende server als er een verzoek binnenkomt. Dit gedrag is gedefinieerd in /etc/inetd.conf.

inetd wordt vanuit systeem opstartscripts gestart. Het erft gewoon het pad van het init proces. Het wijzigt het geheel niet en alle servers die vanuit inetd worden gestart hebben het init-pad. Een voorbeeld van een dergelijk proces is imapd, de server van het IMAP post office protocol.

Andere voorbeelden van inetd processen zijn telnetd, rlogind, talkd, ftp, popd, veel http servers enzovoort.

Vaak gebruik van inetd is nog steeds gecompliceerd door het gebruiken van een gescheiden tcpd programma om de echte server te starten. Het is een programma dat aanvullende beveiligingscontroles uitvoert voordat de echte applicatie wordt gestart. Het beïnvloedt het pad niet.

8.2 rsh

De rsh daemon stelt het pad vanuit _PATH_DEFPATH (/usr/include/paths.h) in dat hetzelfde pad is als die het login programma voor gewone gebruikers gebruikt. Root krijgt hetzelfde pad als de gewone gebruiker.

In feite voert rshd het commando uit dat het op de commando-regel krijgt:

```
shell -c command-line
```

en shell is geen login shell. Het is wenselijk dat alle shells die in /etc/passwd worden genoemd, de -c optie voor op de commando-regel ondersteunen.

8.3 rlogin

Rlogin roept login aan om de echte login procedure te maken. Als je met rlogin inlogt, krijg je hetzelfde pad als in login. De meeste andere manieren om op een Linux computer in te loggen maken geen gebruik van login. Merk het verschil met rsh op.

Het login commando dat feitelijk wordt gebruikt is

```
login -p -h host-name user-name
```

-p bewaart de omgeving met uitzondering van de variabelen HOME, PATH, SHELL, TERM, MAIL en LOGNAME. met -h geef je de naam van de remote host waarop je inlogt op.

8.4 telnet

Telnet is vergelijkbaar met rlogin. Het gebruikt het login programma en de commando-regel om het op vergelijkbare manier aan te roepen.

8.5 ssh

ssh heeft een eigen padinstelling. Het heeft een vast pad waaraan het de directory waar ssh zich in bevindt aan toevoegt. Dit betekent vaak dat /usr/bin twee keer in het pad voorkomt:

```
/usr/local/bin:/usr/bin:/bin:.:/usr/bin
```

In het pad komt /usr/X11/bin niet voor en de shell die door het ssh commando is aangeroepen is geen login shell. Dus

```
ssh remotehost xterm
```

werkt nooit, en van alles in /etc/profile of /etc/csh.cshrc kan dit veranderen. Je moet het pad /usr/bin/X11/xterm altijd expliciet gebruiken.

ssh zoekt naar omgevingsvariabelen in de vorm VAR=VALUE in het bestand /etc/environment. Helaas veroorzaakt dit een aantal problemen met XFree86.

9 XFree86

9.1 XDM

XDM is de meest gebruikelijke manier om op een grafische terminal in te loggen. Het lijkt een beetje op login maar het is intern totaal anders.

In de directory /etc/X11/xdm zijn configuratiebestanden te vinden die in de verschillende login fases worden uitgevoerd. Xstartup (en Xstartup_0 speciaal voor scherm 0) bevatten commando's om te worden uitgevoerd nadat de gebruiker is ingelogd (commando's worden als root gebruiker uitgevoerd).

Het pad dat voor gebruikers is ingesteld staat in /etc/X11/xdm/xdm-config. Het zijn de regels:

```
DisplayManager*userPath: /usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games
DisplayManager*systemPath: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/bin/X11
```

Dat zal het standaardpad voor respectievelijk de gewone en root gebruikers zijn. Het is erg belangrijk dat /usr/bin/X11 voor X-gebruikers beschikbaar is. Als een X-gebruiker op een andere machine inlogt om een X-client te starten zou hij /usr/bin/X11 aan zijn pad toegewezen moeten krijgen, zelfs als hij niet direct vanaf een X-terminal inlogt.

Na het draaien van Xstartup draait XDM /etc/X11/Xsession dat als de uiteindelijke gebruiker wordt uitgevoerd. Het is de bedoeling dat de bronnen uit /etc/environment worden gehaald voor de lokale configuratie vanuit Xsession als beschikbaar. (Xsession wordt met /bin/sh uitgevoerd en dus moet /etc/environment een sh bestand zijn). Dit botst met ssh die veronderstelt dat /etc/environment een bestand is dat slechts regels in de vorm VAR=VALUE bevat.

9.2 xterm -ls

Standaard is het pad voor alle commando's die vanuit X-window manager menu's zijn aangeroepen, het pad geërfd van XDM. Om iets anders te gebruiken, moet het expliciet worden ingesteld. Om een terminal emulator met een "normaal"pad te starten, moet een speciale optie worden gebruikt. In een xterm moet de optie -ls (login shell) worden gebruikt om een login shell pad zoals in de shell login initialisatiebestanden te verkrijgen.

9.3 Window manager menu's en knoppen

Een Window manager erft zijn omgeving van XDM. Alle programma's die door de window manager worden gestart erven de omgeving van de window manager.

De gebruikers shellomgeving beïnvloedt niet de programma's die vanuit de window manager knoppen en menu's zijn gestart. Als bijvoorbeeld een programma vanuit 'xterm -ls' is gestart, heeft het de standaardomgeving van de login shell maar als het vanuit een menu wordt gestart, heeft het gewoon de omgeving van de window manager.

10 Uitgestelde commando's cron en at

10.1 cron

Cron is een commando dat commando's periodiek uitvoert zoals is aangegeven in /etc/crontab en gebruikersgedefinieerde crontabs. In Debian 1.3 is het standaardmechanisme om commando's in /etc/cron.daily, /etc/cron.weekly en /etc/cron.monthly uit te voeren.

Cron wordt vanuit opstartscripts gestart maar het lijkt zijn PATH in een nogal vreemd pad te wijzigen:

```
/usr/bin:/binn:/sbin:/bin:/usr/sbin:/usr/bin
```

DIT IS WAARSCHIJNLIJK EEN BUG IN CRON. Dit is het init pad waar het begin is overschreven door /usr/bin:/bin zonder af te sluiten met 0! Deze bug komt niet op alle systemen voor.

In crontab kan een PATH definitie voorkomen. In Debian 1.3 staat de volgende standaardregel aan het begin van /etc/crontab:

```
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
```

Om deze reden wordt het PATH van het crond programma nooit gebruikt in gebruikersprogramma's. Alle scripts in de /etc/cron.* directory's krijgen standaard dit pad. Dit pad wordt zelfs gebruikt als een programma als niet-root wordt uitgevoerd.

10.2 at

at is een commando dat kan worden gebruikt om éénmalig een programma op een specifiek tijdstip uit te voeren.

atd wordt uitgevoerd door gebruik te maken van het init pad. De gebruikersprogramma's worden echter altijd in de gebruikersomgeving met gebruik van het sh commando uitgevoerd. Daarom gelden de gebruikelijke regels van de shell. Kijk hiervoor in het hoofdstuk over bash.

11 Een aantal voorbeelden

11.1 magicfilter

magicfilter is een algemeen hulpmiddel om bestanden voor de printer te manipuleren. Het analyseert het type bestand dat moet worden afgedrukt en roept een filterscript aan om een passende mooie-afdruk te maken. Deze scripts worden vanuit lpd aangeroepen dat wordt gestart vanuit /etc/init.d/lpd dat wordt gestart vanuit init. Dus het pad is dat van init. Daarin staat niet /usr/bin/X11!

Mogelijk wil je een afdruk van PDF bestanden aan magicfilter doorgeven. Het is mogelijk dit te doen door gebruik te maken van /usr/bin/X11/xpdf. Nu moet je er aan denken om het volledige pad van de bestandsnaam op te geven omdat magicfilter het anders niet kan vinden. De meeste programma's die in magicfilter worden gebruikt hebben het volledige pad niet nodig, omdat ze in /bin of /usr/bin staan.

11.2 Afdrukken vanuit X applicaties

Je kunt de PRINTER omgevingsvariabele gebruiken om aan te geven welke printer je gebruikt. Echter het zou kunnen dat je bemerkt dat het in een aantal gevallen bij X applicaties verloren is gegaan.

Je moet er aan denken dat als de X sessie vanuit XDM is opgestart, de window manager de waarde van je shell login scripts heeft bepaald. Alle X applicaties die je vanuit een xterm hebt gestart hebben je PRINTER variabele. Als echter dezelfde applicatie vanuit een menu of window manager button is gestart, bevat het je PRINTER variabele niet.

In een aantal gevallen kan dit zijn geërfd van een nog lagere laag: een Netscape hulpapplicatie kan bijvoorbeeld wel of niet de beschikking over je PRINTER definitie hebben.

12 Beveiligingszaken

Het pad is soms een groot beveiligingsprobleem. Het is een zeer algemene manier om een systeem te kraken door een aantal fouten in padinstellingen te gebruiken. Het is makkelijk om Trojan horse aanvallen te genereren als hacker aan root privileges of die van andere gebruikers kan komen om zijn versies van commando's uit te voeren.

Een veelvoorkomende fout in het verleden (?) was door de '.' in het pad van de root te houden. Een kwaadwillige hacker maakt een programma 'ls' in zijn home-directory. Als root dan vervolgens doet

```
# cd ~hacker
# ls
```

voert hij het ls commando van de hacker uit.

Indirect geldt hetzelfde voor alle programma's die als root worden uitgevoerd. Ieder belangrijk daemon proces zou nooit iets moeten uitvoeren waarin een andere gebruiker iets weg kan schrijven. In een aantal systemen is het mogelijk om in /usr/local/bin programma's te plaatsen met minder strikte beveiligings afscherming. - het is slechts uit het pad van de root gebruiker verwijderd. Het is echter bekend dat een aantal daemon processen 'foo' uitvoert door gebruik te maken van het pad '/usr/local/bin/:...', het zou mogelijk kunnen zijn om de daemon om de tuin te leiden om '/usr/local/bin/foo' in plaats van '/bin/foo' uit te voeren. Het is aannemelijk dat voor iemand die naar '/usr/local/bin' kan schrijven het mogelijk is om op het systeem in te breken.

Het is erg belangrijk te overwegen in welke volgorde de directory's in het pad staan. Als /usr/local/bin voor /bin staat, is dit een beveiligings risico - als het erachter staat, is het niet mogelijk het commando /bin/foo met wat plaatselijke wijzigingen in /usr/local/bin/foo te overschrijven.

Onder Linux zou er aan moeten worden gedacht dat de evaluatie van het pad op het system call level wordt afgehandeld. Overal waar een uitvoerbaar bestandspad wordt opgegeven kun je een verkorte naam opgeven waarnaar op z'n minst wordt gezocht vanuit /bin en /usr/bin - waarschijnlijk ook op vele andere plaatsen.

13 Hoe fouten in problemen op te sporen?

Het basiscommando om de omgeving te lezen is /usr/bin/env.

Het is mogelijk om de /proc directory te gebruiken om uit te zoeken wat het pad is van een bepaald programma. Als eerste moet je het procesnummer kennen - gebruik het ps commando om dat te verkrijgen. Als xterm bijvoorbeeld het procesnummer 1088 is, kun je zijn omgeving vinden met het commando

```
# more /proc/1088/environ
```

Dit werkt niet met daemon processen zoals xdm. Om de omgeving van systeemprocessen of andere gebruikersprocessen te benaderen, is root toegang vereist.

Om problemen in Netscape op te sporen, kun je een script /tmp/test aanmaken:

```
$ cat > /tmp/test
#!/bin/sh
/usr/bin/env > /tmp/env
^d
$ chmod +x /tmp/test
```

Stel vervolgens een hulpapplicatie in, bijvoorbeeld RealAudio, audio/x-pn-realaudio om het programma /tmp/testäan te roepen. Als je naar een ReasAudio link probeert te browsen (iets vanuit http://www.realaudio.com/showcase), roept Netscape het dummy programma aan dat de omgeving opslaat in /tmp/env.

14 Een aantal stategiën om hetzelfde pad voor alle gebruikers te verkrijgen

Van de belangrijkste instellingen is het mogelijk om ze in te stellen in de globale shell initialisatie bestanden voor login shells: /etc/csh.login voor tcsh en /etc/profile voor bash.

Uitzonderingen die het juiste pad niet vanuit deze bestanden verkrijgen, zijn rsh commando's, ssh commando's, menu items vanuit de X window manager die de login shell niet expliciet opstarten, commando's aangeroepen vanuit inittab, cron jobs, daemons jobs zoals magic filters gestart vanuit lprd, WWW CGI scripts, enzovoort.

Als het pad in /etc/csh.cshrc is ingesteld, is het pad goed zelfs als rsh of ssh een commando uitvoert op de remote machine met een account die gebruik maakt van tcsh/csh. Het is echter niet mogelijk om het pad in te stellen als de account bash/sh gebruikt.

Het is mogelijk padinstellingen in één bestand te combineren, bijvoorbeeld in een bestand /etc/environment-common. Daarin schrijven we:

```
${EXPORT}PATH${EQ}/bin:/usr/bin:/sbin:/usr/sbin:/usr/bin/X11:/usr/local/bin:/usr/games:.
```

Dit kan vanuit /etc/csh.login (voor tcsh en csh) worden gebruikt.

```
set EQ=" " set EXPORT="setenv " source /etc/environment-common
```

En vanuit /etc/profile (voor bash, werkt niet voor gewone sh)

```
EQ='=' EXPORT="export " . /etc/environment-common
```

En vanuit /etc/environment (voor XDM)

```
EQ="=" EXPORT="export " . /etc/environment-common
```

Deze strategie werkt meestal maar ssh zal klagen over de regels in /etc/environment (en gedefinieerde omgevingsvariabelen EQ en EXPORT). En nog steeds zullen rsh commando's die met bash worden uitgevoerd dit pad niet krijgen.

15 Erkenningen

Een reden om te beginnen met het schrijven van dit document was de grote frustratie van Ari Mujunen. Juha Takal gaf een aantal waardevolle opmerkingen.