Despre felul în care lucrurile vechi pot deveni noi

Mandrake Linux

Mai multe frag-uri ca oricând

UT 2003

Pe coama valului

Mozilla 1.6

Ripping şi encoding

De la CD-DA la MP3

linux360

februarie - martie 2004

06

.coperta

O altă Viziune

Mai întâi a fost, ca la orice început, ideea. O idee de mai bine pentru utilizatorii români de Linux, pentru delăsarea cu care asamblatorii români instalatează o distribuție Linux pe un un sistem doar pentru a-i scădea prețul şi a atrage cu ofertele din supermarket-uri, pentru comunitatea Linux din România. I s-a spus Vision.

Pentru fundație a fost ales un arbore de dezvoltare dintre cele mai dinamice, încorporând permanent ultimele versiuni ale programelor şi pachetelor. Din acesta, s-a dezvoltat propria ramură, prin încorporarea propriilor pachete personalizate, a îmbunătățirii uneltelor ce vor fi puse la dispoziția utilizatorilor şi, poate cel mai important, a creării unei interfețe personalizată, localizată şi modificată pentru a atinge un grad înalt de ergonomie.

Sună destul de simplu, dar, pornind ca un proiect închis, a fost o muncă sisifică din partea coordonatorului proiectului. Neavând prea mulți dintre noi experiența unei creări de distribuție, de multe ori el a centrat, el a dat cu capul. Când am văzut rezultatul, am rămas uimit. Era mai frumos decât orice văzusem până atunci şi pe care l-aş fi vrut pe desktop-ul propriu.

Încotro vom merge de aici? Se poate spune că utilizatorii ne vor hotărî paşii. linux360 a oferit o revistă, acum va oferi şi o distribuție destinată aceloraşi categorii de utilizatori ca şi revista. Dacă utilizatorii se vor implica atât în testare şi evaluare, cât şi, de ce nu, în dezvoltare, va ieşi cu un lucru nemaipomenit pentru toți cei care vor face primul pas în această lume a Pinguinului cu surse libere.

Viață lungă şi prosperă, Vision!

Ovidiu

linux360 - nr. 06 februarie - martie 2004

www.linux360.ro / redactia@linux360.ro

Noutăți

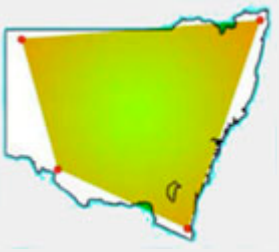
O nouă versiune de **Open Office**, ajunsă la numărul 1.1.1, este disponibilăa pentru download pe site-ul oficial www.openoffice.org. Ea nu aduce multe noutăți ci mai mult rezolvări ale anumitor bug-uri ce au fost descoperite în versiunea anterioară. După lansarea OpenOffice.org 1.1.1, pe site-ul madpenguin.org a apărut o ştire în care se menționa că versiunea finală nu diferă cu nimic de versiunea precedenta RC3 după ce au fost verificate sumele de control MD5 al fişierului de instalare. Aceasta se poate explica prin faptul că în lumea dezvoltătorilor GNU atât a programelor cât şi a Linux Kernel sunt cazuri în care, după ce a fost prezentat un ultim RC, nu au mai fost descoperite bug-uri semnificative şi este lansată versiunea finală ce nu diferă de RC.

Sistemul de fişiere **Reiser4** este aproape gata pentru utilizare. După o perioadă îndelungată de dezvoltare şi testare, bine cunoscutul şi popularul sistem de fişiere este pregătit pentru a se instala în PC-urile utilizatorilor. Până la anunțul oficial vor trebui rezolvate două bug-uri. Unul este legat de NFS şi altul de mmap. După rezolvarea acestora, se poate începe implementarea şi trecerea în producție la nivel de utilizator. Momentan nu se recomandă trecerea sistemelor de producție pe noua versiune reiserfs4 din cauza necesității efectuării mai multor teste.

O noua versiune de test a Fedora Core, şi anume **Fedora Core 2 Test 2**, este disponibilă pentru publicul larg. Dintre noutățile pe care le aduce noua versiune se află un kernel 2.6 peste care putem găsi rulând GNOME 2.5 şi KDE 3.2. Această versiune de test este disponibilă numai pentru platformele x86 şi x86-64 şi nu se recomandă a se folosi în sistemele de producție.

Elveția adoptă GNU pe scară largă. Consiliul IT al federației elveține a adoptat la sfârşitul lunii februarie în unanimitate versiunea 1.0 a "Federal Administration's open source software (OSS) strategy". În această strategie se menționează Linux ca un potențial sistem de operare pentru desktop. În strategie se specifică că nu vor exista discriminări între closed-source şi open-source, ci vor fi tratate în egală măsură în analiza pentru implementarea în proiecte de interes național.

Cea mai mare rețea de sateliți rulând Linux. Peste 75 de oraşe din regiunea New South Wales au de acum acces la Internet prin sateliți unde traficul este rutat de Linux. Este cea mai mare rețea de sateliți din Australia ce acoperă o suprafață de 800.000 kilometri pătrați. Această soluție este mai ieftină în regiunile îndepărtate ale Australiei decât utilizarea liniilor ISDN. Această rețea face parte din inițiativa Rural Link Network ce este promovată de guvernul australian şi de mai multe companii private. Pe routerele terestre este instalat Debian ce în momentul de față poate oferi servicii de firewall, QoS, IPSec VPN, mail şi DNS care sunt administrate de la distanță utilizând o interfață web. Rețeaua foloseşte o stivă TCP/IP proprietară ce nu este compatibilă cu standardul TCP/IP, din cauza latenței ce se manifestă în comunicațiile satelit.



A apărut o nouă versiune a e2fsprogs, **e2fsprogs 1.35**. Aceasta include utilitare pentru mentenanța sistemelor de fişiere ext2 şi ext3. Noua versiune este compatibilă cu kernelul Linux 2.6 la care s-au adăugat mai multe noi funcționalități cum sunt directoarele htree din Kernel şi patch-uri backport-ate din kernelurile seria 2.4.

SUN Microsystems este pe cale să redescopere lumea cu viitorul său desktop. Proiectul **Looking Glass** la care lucrează programatorii companiei SUN este în întregime bazat pe limbajul de programare Java. Din primele impresii arată foarte promițător. Proiectul încearcă să aducă pe desktop-urile utilizatorilor senzația tridimensională a lumii virtuale unde manipularea obiectelor şi navigarea se face 3D. Sperăm să-l vedem pe desktop-urile noastre cât mai curând.

Introducere în administrare

Radu - Eosif Mihăilescu

Bun găsit în capitolul de introducere în administrarea sistemelor Linux! În numărul trecut am început descrierea unei infrastructuri folosite pe scară largă pentru autentificare în sisteme Linux, şi anume PAM (Pluggable Authentication Module) cu o descriere arhitectural-funcțională şi un exemplu succint de utilizare.

Am amintit atunci că această infrastructură este (după cum şi denumirea o sugerează) bazată pe module - module ce pot îndeplini patru "roluri de bază" (account, auth, password şi session) în procesul de autentificare şi nu numai. În acest număr vom trece în revistă unele astfel de module - cele mai cunoscute şi des folosite în practică, dar mai întâi să dăm câteva detalii ce țin de funcționarea acestui sistem complex.

Cum lucrează de fapt PAM?

Atunci când un program proiectat să folosească PAM are de făcut autentificare, el începe un "dialog" cu infrastructura prin intermediul API-ului specific. Nu voi intra în detalii ce țin de programare aici, ci voi spune doar că prima funcție apelată, cea care inițiază dialogul are un parametru care este interesant pentru noi şi anume "numele serviciului".

În context PAM, numele serviciului este un şir de caractere care este folosit de infrastructură pentru a selecta (şi pune în aplicare) configurația specifică acelei aplicații. Aceste configurări pot fi date în două feluri (mutual exclusive):

- ca linii de configurare în /etc/pam.conf ce au ca prim câmp numele serviciului
- ca fişiere în `/etc/pam.d` ce au ca denumire numele serviciului.

În cazul în care configurația cerută de un program nu există sau nu este validă, PAM va considera că i-a fost pasat numele de serviciu `"other"` ca argument (vezi `/etc/pam.d/other`). De cele mai multe ori, numele de serviciu în sens PAM este identic (sau foarte apropiat) cu cel al fişierului executabil principal al respectivului program.

După ce a selectat configurația cerută de programul apelant, PAM este în aşteptarea unei cereri de la acesta. În momentul în care programul face o cerere, ea se înscrie într-una din cele patru categorii enumerate anterior (`account`, `auth`, `password` şi `session`) şi PAM porneşte la executarea ei parcurgând din configurarea selectată liniile care se referă la categoria în cauză, de sus în jos (deci ordinea contează).

Fiecare modul astfel declarat este încărcat, inițializat şi apoi "întrebat" de PAM "ce părere are" despre cererea programului apelant. Modulele PAM se comportă ca un juriu: răspund doar cu "vinovat" sau "nevinovat". În funcție de specificațiile de sens din configurare (`required`, `optional` şi `sufficient`), PAM poate "calcula" un "răspuns final" în urma interogării tuturor modulelor aflate în listă, răspuns final pe care îl întoarce programului apelant care este în acest moment liber sa înterpreteze decizia PAM după cum crede de cuviință.

Aşa funcționează, în mare, PAM. Să trecem acum mai departe.

Module PAM "celebre"

- `pam_stack`: după cum poate bănuiți, acest modul are un rol foarte simplu (dar în acelaşi timp foarte puternic!), şi anume acela de a "stivui" (engl. "to stack") liste de module din configurări diferite. El ia un singur parametru, "service", ce are ca valoare numele de serviciu a cărui listă de module se doreşte procesată "în numele" modulului
- `pam_unix`: acest modul este cel care se ocupă de semanticile UNIX ale autentificării. De exemplu, caută/schimbă parolele din `/etc/passwd` sau `/etc/shadow` (după configurare)
- `pam_permit`, `pam_deny`: perechi funcționale ale celebrelor `/bin/true` şi `/bin/false`, aceste module se folosesc pentru a da (impune) un anume curs lanțului de decizie. Primul modul întoarce (folosind metafora anterioară a juriului) "nevinovat" indiferent de context iar cel de al doilea, respectiv, "vinovat"
- `pam_nologin`, `pam_securetty`, `pam_shells`: aceste trei module reprezintă tot "reîncarnări" ale semanticilor UNIX in PAM. Astfel, primul interzice accesul oricărui alt utilizator în afară de root daca fişierul `/etc/nologin` există, cel de al doilea interzice accesul utilizatorului root dacă accesul se face de pe un TTY care nu apare în `/etc/securetty` iar ultimul interzice accesul utilizatorilor al căror shell nu apare în `/etc/shells`
- `pam_cracklib`: acest modul are rolul de a testa parola utilizatorului împotriva unui dicționar de cuvinte uzuale, interzicând schimbarea de parolă daca aceasta se găseşte în dicționar
- `pam_limits`, `pam_env`, `pam_chroot`, `pam_console`: acestea sunt doar câteva dintre modulele care au ca rol efectuarea de "manevre de culise" la începutul (respectiv sfârşitul) unei sesiuni de lucru a unui utilizator. Primul este o automatizare a funcționalității

prezentate de comanda "ulimit", cel de al doilea se ocupă de pregătirea si setarea unui set predefinit de variabile de mediu, cel de al treilea este o automatizare a apelului "chroot();" iar ultimul se ocupă cu schimbarea proprietarului şi/sau a drepturilor pe anumite fişiere

- pam_access, pam_time, pam_localuser: sunt trei module care implementează restricții complexe de acces. Astfel, primul permite accesul numai daca tranzacția curentă satisface regulile din /etc/security/access.conf, cel de al doilea permite accesul numai dacă sunt respectate restricțiile temporale din /etc/security/time.conf iar ultimul permite accesul numai dacă autorul cererii este un utilizator local acestui sistem.

Acestea sunt numai câteva module - există mult mai multe, chiar şi gata instalate. În urma unei căutări pe Internet vom găsi şi mai multe - pentru cele mai diverse şi suprinzătoare scopuri, trebuie doar să ne alegem ceea ce ne trebuie şi să le folosim în configurațiile PAM ale diferitelor programe.

Pentru mai multe informații, consultați /etc/pam.d pentru lista de configurații curente, /lib/security pentru lista de module curente şi documentația PAM pentru orice alte detalii.

Cam acestea ar fi de spus în contextul unei prezentări generale a infrastructurii PAM. Să trecem acum la cel de al doilea subiect "fierbinte" şi anume:

Interfețe de rețea în Linux

Înainte de a discuta de configurare, trebuie să facem o mică discuție teoretică. Aşadar, în context Linux, o interfață de rețea este un "obiect" care are proprietățile următoare:

- reprezintă o cale de acces către un mediu de transmisie compatibil cu conceptul de rețea (fie el chiar şi virtual)
- suportă unul sau mai multe protocoale de rețea - ca atare, acceptă una sau mai multe adrese specifice respectivelor protocoale deschizând astfel "porți de acces" spre acele tipuri de rețele.

O interfață de rețea se poate afla în starea "activată" (up) sau "dezactivată" (down) - şi aceasta indiferent de configurația ei de la un moment dat. O interfață de rețea este mai întâi creeată, apoi configurată şi în cele din urmă activată.

Interfețele de rețea au modalități diverse de creere: cele reprezentând plăci fizice de rețea sunt creeate în momentul inițializării driver-ului lor iar cele reprezentând dispozitive virtuale de obicei sunt creeate cu ajutorul unui program de control specific.

Configurarea interfețelor de rețea (după ce au fost creeate) se face cu /sbin/ifconfig - şi tot cu acest utilitar se execută şi controlul activității lor.

După cum poate v-ați obişnuit, există nişte denumiri consacrate şi pentru diferitele tipuri de interfețe de rețea, după cum urmează:

- lo: aceasta este interfața zisă "loopback" (buclă de intoarcere), adică cea care poartă celebra adresă 127.0.0.1 (sau ::1 in notație IPv6) şi serveşte traficului ce nu părăseşte sistemul
- eth0: aceasta este interfața corespunzătoare primei plăci de rețea Ethernet - pentru următoarele se creşte numărul
- ppp0: aceasta este prima interfață corespunzătoare unei legături PPP (indiferent de mediu)
- sl0: analog pentru legături SLIP
- plip0: analog pentru legături PLIP
- gre0, tunl0, sit0: acestea sunt interfețe virtuale folosite în tehnica "tunelării". Ele reprezintă interiorul unui tunel ce are un capăt pe acest sistem
- tap0, dummy0: acestea sunt interfețe virtuale cu scopuri speciale. Prima este similară cu un PTY (permite unui program să se comporte ca o placă de rețea) iar cea de a doua este similară cu /dev/null ... adică nu face nimic
- ax0: această interfată corespune unui echipament AX.25 (folosit de radioamatori în rețelele radio pachet)
- dvb0: această interfață corespunde unui receptor de satelit digita folosit pentru date.

Cam atât despre interfețe de rețea in Linux acum, data viitoare vom vorbi despre alias-uri la interfețe, despre protocoale (familii de adrese, cum mai sunt ele denumite), despre resurse legate de interfețe (buclucaşele apeluri bind (); şi select();) şi, în fine, despre rutare.

Autor:

radu.mihailescu@linux360.ro

Criptografie - RC2

Radu - Eosif Mihăilescu

În continuarea seriei dedicată criptografiei (aşa cum este ea folosită în domeniul informatic), vom vorbi acum despre un algoritm de cifrare.

Este vorba de un algoritm simplu dar eficient şi în acelaşi timp destul de vechi, precursor al celebrului astăzi RC4 (măcar din auzite). Doamnelor şi domnilor, vă prezint pe:

Algoritmul de cifrare RC2

Mai întâi trebuie să spunem că acest algoritm este proprietatea intelectuală a RSA Data Security, Inc.

Algoritmul discutat este un algoritm de cifrare convențional (cu cheie secretă) bazat pe blocuri. Dimensiunile blocurilor de intrare, respectiv ieşire ale algoritmului sunt egale între ele şi cu 64 de biți. Lungimea cheii este variabilă de la un octet la 128, totuşi, implementarea actuală foloseşte 8 octeți.

Acest algoritm a fost conceput pentru a putea fi uşor implementat pe procesoare cu lățimea cuvântului de date de 16 biți.

Descrierea algoritmului

Pe parcursul acestui document vom folosi termenul "cuvânt" pentru a denumi o valoare pe 16 biți. De asemenea, vor fi folosite următoarele operații caracteristice algebrei Boole: adunarea, modulo, conjuncția, disjuncția exclusivă, rotația la dreapta, respectiv stânga şi negația.

Acest algoritm are trei părți funcționale:

- *expansiunea cheii*: această parte are ca intrare o cheie de lungime variabilă şi produce o cheie de criptare de 64 de cuvinte K[0], ..., K[63]
- *criptarea*: aceasta ia ca intrare o cantitate pe 64 de biți stocată în cuvintele R[0], ..., R[3] şi o criptează "în loc" (rezultatul este obținut tot în R[0], ..., R[3])
- *decriptarea*: operația inversă criptării.

1. Expansiunea cheii

Înainte de expansiunea propriu-zisă a cheii, se construieşte un vector de 256 valori numerice de câte un octet denumit în continuare `sBox[]`. Fiecare element din acest vector este obținut prin următoarea operație:

$$sBox[i] = (B[i] \bmod 256) \oplus p[i]$$

unde B[] este un vector ce conține primele 256 de valori din Cifrul lui Beale numărul 1 iar p[] este un vector ce conține 256 de valori pentru ajustare.

Cifrul lui Beale numărul 1, vectorul de ajustare precum şi vectorul `sBox[]` rezultant precalculat sunt date la sfârşitul acestui document.

Având construit pe `sBox[]`, putem trece acum la expansiunea la 128 de octeți a cheii care se efectuează după algoritmul următor: se iau primul şi ultimul octet din cheia furnizată inițial, se adună modulo 256 şi rezultatul se caută în `sBox[]`, valoarea ce îi corespunde acolo fiind adăugată cheii furnizate. Această operație se repetă apoi pentru al doilea octet al cheii furnizate şi octetul proaspăt adăugat - şi tot aşa până când cheia atinge 128 de octeți. O ultimă operație mai are loc în acest moment şi anume înlocuirea primului octet din cheie cu valoarea corespunzătoare din `sBox[]` - în acest moment cheia fiind gata pentru criptare. În cele ce urmează, cheia de criptare astfel obținută va fi notată cu `S[]` şi va fi considerată un vector de 64 de cuvinte (în locul unui vector de 128 de octeți).

{Algoritmul de criptare}

```
for i := 0 to 15 do begin
j := i * 4;
temp := ( R[0] + ( R[1] and not R[3] ) + ( R[2] and R[3] ) + S[j + 0]);
R[0] := ( temp shl 1 ) or ( temp shr 15 ); { Pascal nu are rol }
temp := ( R[1] + ( R[2] and not R[0] ) + ( R[3] and R[0] ) + S[j + 1]);
R[1] := ( temp shl 2 ) or ( temp shr 14 );
temp := ( R[2] + ( R[3] and not R[1] ) + ( R[0] and R[1] ) + S[j + 2]);
R[2] := ( temp shl 3 ) or ( temp shr 13 );
temp := ( R[3] + ( R[0] and not R[2] ) + ( R[1] and R[2] ) + S[j + 3]);
R[3] := ( temp shl 5 ) or ( temp shr 11 );
if (i = 4) or (i = 10) then begin
R[0] := R[0] + S[R[3] and 63]; R[1] := R[1] + S[R[0] and 63];
R[2] := R[2] + S[R[1] and 63]; R[3] := R[3] + S[R[2] and 63];
end;
end;
```

{Algoritmul de decriptare}

```
for i := 15 to 0 do begin
j := i * 4;
temp := ( R[3] shr 5 ) or ( R[3] shl 11 ); { Pascal nu are rol }
R[3] := temp - ( R[0] and not R[2] ) - ( R[1] and R[2] ) - S[j + 3];
temp := ( R[2] shr 3 ) or ( R[2] shl 13 );
R[2] := temp - ( R[3] and not R[1] ) - ( R[0] and R[1] ) - S[j + 2];
temp := ( R[1] shr 2 ) or ( R[1] shl 14 );
R[1] := temp - ( R[2] and not R[0] ) - ( R[3] and R[0] ) - S[j + 1];
temp := ( R[0] shr 1 ) or ( R[0] shl 15 );
R[0] := temp - ( R[1] and not R[3] ) - ( R[2] and R[3] ) - S[j + 0];
if (i = 4) or (i = 10) then begin
R[3] := R[3] - S[R[2] and 63]; R[2] := R[2] - S[R[1] and 63];
R[1] := R[1] + S[R[0] and 63]; R[0] := R[0] - S[R[3] and 63];
end;
end;
```

{Cifrul lui Beale numarul 1}

```
const Beale1:array[0..255] of Word = (71, 194, 38, 1701, 89, 76, 11, 83, 1629, 48, 94,
63, 132, 16, 111, 95, 84, 341, 975, 14, 40, 64, 27, 81, 139, 213, 63, 90, 1120, 8, 15, 3,
126, 2018, 40, 74, 758, 485, 604, 230, 436, 664, 582, 150, 251, 284, 308, 231, 124,
211, 486, 225, 401, 370, 11, 101, 305, 139, 189, 17, 33, 88, 208, 193, 145, 1, 94, 73,
416, 918, 263, 28, 500, 538, 356, 117, 136, 219, 27, 176, 130, 10, 460, 25, 485, 18,
436, 65, 84, 200, 283, 118, 320, 138, 36, 416, 280, 15, 71, 224, 961, 44, 16, 401, 39,
88, 61, 304, 12, 21, 24, 283, 134, 92, 63, 246, 486, 682, 7, 219, 184, 360, 780, 18, 64,
463, 474, 131, 160, 79, 73, 440, 95, 18, 64, 581, 34, 69, 128, 367, 460, 17, 81, 12,
103, 820, 62, 110, 97, 103, 862, 70, 60, 1317, 471, 540, 208, 121, 890, 346, 36, 150,
59, 568, 614, 13, 120, 63, 219, 812, 2160, 1780, 99, 35, 18, 21, 136, 872, 15, 28, 170,
88, 4, 30, 44, 112, 18, 147, 436, 195, 320, 37, 122, 113, 6, 140, 8, 120, 305, 42, 58,
461, 44, 106, 301, 13, 408, 680, 93, 86, 116, 530, 82, 568, 9, 102, 38, 416, 89, 71,
216, 728, 965, 818, 2, 38, 121, 195, 14, 326, 148, 234, 18, 55, 131, 234, 361, 824, 5,
81, 623, 48, 961, 19, 26, 33, 10, 1101, 365, 92, 88, 181, 275, 346, 201, 206);
```

{Vectorul de ajustare}

```
const RC2Pad:array[0..255] of Byte = (158, 186, 223, 97, 64, 145, 190, 190, 117, 217,
163, 70, 206, 176, 183, 194, 146, 43, 248, 141, 3, 54, 72, 223, 233, 153, 91, 210, 36,
131, 244, 161, 105, 120, 113, 191, 113, 86, 19, 245, 213, 221, 43, 27, 242, 157, 73,
213, 193, 92, 166, 10, 23, 197, 112, 110, 193, 30, 156, 51, 125, 51, 158, 67, 197, 215,
59, 218, 110, 246, 181, 0, 135, 76, 164, 97, 47, 87, 234, 108, 144, 127, 6, 6, 222, 172,
80, 144, 22, 245, 207, 70, 227, 182, 146, 134, 119, 176, 73, 58, 135, 69, 23, 198, 0,
170, 32, 171, 176, 129, 91, 24, 126, 77, 248, 0, 118, 69, 57, 60, 190, 171, 217, 61,
136, 169, 196, 84, 168, 167, 163, 102, 223, 64, 174, 178, 166, 239, 242, 195, 249, 92,
59, 38, 241, 46, 236, 31, 59, 114, 23, 50, 119, 186, 7, 66, 212, 97, 222, 182, 230, 118,
122, 86, 105, 92, 179, 243, 255, 189, 223, 164, 194, 215, 98, 44, 17, 20, 53, 153, 137,
224, 176, 100, 208, 114, 36, 200, 145, 150, 215, 20, 87, 44, 252, 20, 235, 242, 163,
132, 63, 18, 5, 122, 74, 97, 34, 97,142, 86, 146, 221, 179, 166, 161, 74, 69, 182, 88,
120, 128, 58, 76, 155, 15, 30, 77, 216, 165, 117, 107, 90, 169, 127, 143, 181, 208,
137, 200, 127, 170, 195, 26, 84, 255, 132, 150, 58, 103, 250, 120, 221, 237, 37, 8,
99);
```

2. Criptarea

Cifrul are 16 parcurgeri complete, fiecare subdivizată în 4 etape. Două din trecerile complete fac o procesare suplimentară a datelor de intrare. Algoritmul (folosind notațiile anterioare) este prezentat mai jos.

3. Decriptarea

Aceasta este pur-şi-simplu inversa operației de criptare. Algoritmul este prezentat mai jos folosind semantici Pascal (ca şi cel anterior).

Aceasta este descrierea completă a algoritmului de criptare RC2. Acest algorim este folosit astăzi pe scară largă în criptografia informatică, în cadrul infrastructurilor care au legătură cu criptografia, cum ar fi: SSL, TLS, PGP, PKI

În numărul viitor vom descrie pe fratele său "mai mare" şi anume pe RC4 (acesta din urmă fiind mai popular astăzi decât RC2) şi vom da câteva explicații privitor la câteva moduri de utilizare standard a unui cifru în contextul criptografiei informatice.

Până atunci, ca de obicei cu speranța utilității, vă urezi să aveți cifrurile neinversabile si o zi bună, al dumneavoastră:

Autor:
radu.mihailescu@linux360.ro

Kernel Logo

Răzvan Vilt

Schimbarea logo-ului în kernel-ul Linux nu este o necesitate, dar este o plăcere. Într-un recent sondaj făcut de echipa Max-IT (dacă nu mă înşel) s-a arătat că foarte multă lume ştie de Linux, de performanța lui, de stabilitatea lui, şi de securitatea lui, dar toți ştiu că este dificil, are doar o interfață text neprietenoasă. Adevărul este ca odată ce eşti sigur că un kernel îți merge bine, acele mesaje pe care le arată sunt complet nenecesare (pentru majoritatea userilor).

Pentru a-l face mai prietenos am început aici o serie de mini-articole prin care să prezentăm modalitățile de a-i da o interfață vizuală plăcută ochiului. Paşii care ar trebui luați sunt: logo pentru kernel, front-end pentru init şi un login-manager complet customizabil. În acest articol ne vom ocupa de prima parte a acestor probleme.

Soluțiile pentru un logo de kernel sunt variate şi diferite între versiunile curente de kernel. Am încercat să găsesc una care este aproape identică pentru vesiunile 2.4 şi 2.6 de kernel. Dacă kernel-ul din seria 2.6 are o metodă integrată foarte lejeră de schimbare a logo-ului, 2.4 are o metodă mai dificilă, care implică modificarea surselor.

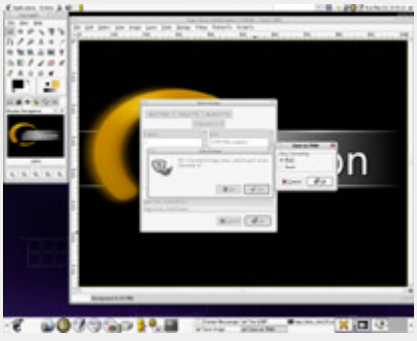
Soluții pentru 2.4.

Cea mai cunoscută soluție este Bootsplash. Deşi efectul vizual poate fii spectaculos in cazul Bootsplash, am ales să nu-l folosesc datorită complicațiilor pe care le aduce şi penalizărilor aduse performanței, dar şi datorită marimii spectaculoase a kernel-ului generat şi nu îm ultimul rând, din cauza riscului de securitate pe care-l aduce. Acest risc nu s-a materializat încă într-o metodă de spargere a sistemelor cu Bootsplash instalat, dar din moment ce Alan Cox este foarte îngrijorat de securitatea acestui patch, sunt şi eu. O altă soluție este "Linux Kernel Logo Patch Project" (http://www.arnor.net/linuxlogo/). Am ales această soluție pentru că era una care implementa strict ceea ce era necesar şi într-un mod foarte asemănător cu kernel-ul 2.6.

Pregătiri.

Un logo nu trebuie să fie de 80x80. Eu de exemplu mi-am făcut unul de 1024x768, aceasta fiind rezoluția preferată în consolă, deci putem să începem prin realizarea unui logo la rezoluția dorită, cu următoarele mențiuni:

1. Numărul de pixeli pe orizontală cât şi pe verticală trebuie să fie multiplu întreg de 8.
2. Numărul maxim de culori este 224.
3. Asta nu este obligatoriu, dar este recomandat: imaginea să fie simpla şi cu puține culori, pentru ca să poată fii comprimat kernel-ul cât mai mult.
4. Imaginea trebuie salvată în formatul pnm ASCII numit logo_linux_clut224.ppm pentru o versiune 2.6 a nucleului, şi pnm RAW cu numele de linux_logo.ppm pentru o versiune 2.4 a nucleului.



Kernel 2.4

OK, dacă lucrăm cu un kernel 2.4 trebuie să ne apucăm de download-at. Sunt 6 patch-uri mici care trebuiesc scoase de pe pagina de download a "Linux Kernel Logo Patch Project" (http://www.arnor.net /linuxlogo/download.html).

Acum hai să le aplicăm:

```
# cd /usr/src/linux-2.4.##

# gzip -cd $path/$către/
$download-uri/logo-01
.patch.gz | patch -p1

# gzip -cd $path/$către/
$download-uri/logo-02.
patch.gz | patch -p1

# gzip -cd $path/$către/
$download-uri/logo-03.
patch.gz | patch -p1

# gzip -cd $path/$către/
$download-uri/logo-04.
patch.gz | patch -p1

# gzip -cd $path/$către/
$download-uri/logo-05.
patch.gz | patch -p1

# gzip -cd $path/$către/
$download-uri/logo-06.
patch.gz | patch -p1
```

Pentru utilizatorii Fedora Core 1, dacă nu le merg patch-urile, nu este nici o problemă, pentru că am creat eu un rând de patch-uri care merg. Acestea sunt accesibile la adresa http://alterego.linux360.ro/~rvilt /resources/kernel/patches /linux-logo-2.4.22-1.2166. nptl.patch.gz, sub forma unui singur patch. S-ar putea să meargă şi pentru alte surse

de kernel mai noi din seria 2.4, dacă nu merg patch-urile de mai sus

Acum trebuie să copiem logo-ul făcut mai devreme (pnm RAW nu-i aşa?) în /usr/src/linux-$ver/arch/i386 /linux_logo.ppm, sau dacă sunteți mai norocoşi şi aveți alte arhitecturi de genul PPC/MIPS/PPC64/Alpha în directorul aferent arhitecturii.

OK. Acum ce? Păi... Mai trebuie doar să modificăm opțiunile noi din kernel şi să-l recompilăm.

Opțiunile care trebuiesc schimbate sunt:

```
CONFIG_FB_DEFAULT_CURSOR
_BLINK_RATE
```

care se referă la rata la care clipeşte cursorul şi este de obicei la 20. Pentru a avea un logo care să şi arate bine este recomandat să-l setăm momentan la -1.

```
CONFIG_FB_PROC_CURSOR
_BLINK_RATE
```

este o opțiune care ne este foarte utilă pentru a nu pierde cursorul de tot. La pornirea sistemului putem sa facem cursorul din nou vizibil când avem nevoie de el.

O soluție este:

```
echo "echo 20 > /proc/sys
/dev/fb/cursor_blink_rate"
>> /etc/rc.d/rc.sysinit
```

Pentru recompilat în principal ajung:

```
# make -s dep bzImage
modules modules_install
install
```

Dacă folosiți Fedora Core (posibil sa fie la fel si pentru RedHat Linux 9), datorită numărului uriaş de modificări aduse kernel-ului, trebuie ceva mai mulți paşi:

```
# make -s oldconfig
# make -s oldconfig //pentru că
```

există câteva dependințe ciclice care trebuie rezolvate.

```
# make menuconfig //trebuie
modificată aici Cursor_Blick_Rate şi
SysCtl_Modifyable_Blink_rate

# make -s dep

# make -s
include/linux/version.h

# make -s CC=gcc32
CFLAGS_KERNEL="-Wno-unused -
g" bzImage

# make -s CC=gcc32
CFLAGS_KERNEL="-Wno-unused"
modules

# make -s modules_install

# make -s install
```

După ce am terminat aici, ar mai fii câteva opțiuni de kernel care ar trebui adăugate la boot-are. Ele sunt console=/dev/tty2 si vga=###. Am inclus desigur si un mic tabel cu rezoluțiile/adâncimile de culoare pentru linux

culori rezoluție	640 x480	800 x600	1024 x768	1280 x1024
256	769	771	773	775
32k	784	787	790	793
64k	785	788	791	794
16M	786	789	792	795

Kernel 2.6

In kernel-ul 2.6 nu trebuie nici un patch. Totul este inclus in surse. E de asemenea adevărat că nu mai există suport pentru modificarea ratei de clipire şi ascunderea cursorului.

Acum trebuie să copiem logo-ul făcut mai devreme (pnm ASCII nu-i aşa?) în /usr/src/linux-$ver/drivers/video /logo/linux_logo.ppm, sau dacă sunteți mai norocoşi şi aveți alte

Logo Kernel

arhitecturi de genul PPC/MIPS/PPC64/Alpha în directorul aferent arhitecturii.

Pentru recompilat la versiunea 2.6 a nucleului linux nu trebuie decât:

```
# make

# make modules

# make modules_install

# make install
```

Şi aici după ce am terminat, ar mai fii câteva opțiuni de kernel care ar trebui adăugate la boot-are. Ele sunt "quiet" şi "vga=###" (conform tabelului de mai sus).

În speranța că a mers totul cum trebuie, am inclus mai sus un screen-shot cu rezultatul.

Autor:

razvan.vilt@linux360.ro

Lucrul pe ciornă - Linux From Scratch

Andrei Ciubotică

Mai toți dintre noi, utilizatorii de Linux, folosim o distribuție binară, adică o colecție de programe / utilitare open-source adunate şi integrate cu grijă de un anumit producător. Este un proces îndelungat, pe care numai o echipă bine pregătită reuşeşte să-l facă. Teoretic, prețul care ar trebui plătit pentru respectiva distribuție acoperă suportul oferit de tehnicieni, faptul că au scris pe un suport amovibil (CD-uri şi/sau DVD-uri) distribuția, manualele tipărite incluse în pachet şi eventual, prețul pentru programele proprietare dezvoltate de către aceştia. Cam acestea sunt costurile ce trebuie acoperite prin vânzarea către utilizatorul final a pachetului ce conține distribuția. Dacă acest preț este justificat sau nu, rămâne la latitudinea utilizatorilor.

Aici intră în scenă proiectul Linux From Scratch, care îşi propune să învețe utilizatorii cum să îşi construiască o distribuție a lor, adaptată propriilor nevoi. Da, frumos spus, dar, pe măsură ce veți avansa în crearea distribuției vă veți întreba din ce în ce mai des dacă merită sau nu timpul alocat. Am să punctez plusurile şi minusurile din punctul meu de vedere, adică a unui utilizator care a încercat această distribuție.

Proiectul LFS poate fi găsit la adresa www.linuxfromscratch.org, unde este afişată o listă de mirror-uri. Abia după ce ați ales o anumită locație (preferabil cât mai aproape de locația dumneavoastră geografică) puteți să descărcați manualul şi sursele ce compun distribuția. Dacă nu aveți lățime de bandă acasă, încercați să mergeți la cunoscuți / InternetCafe-uri pentru că e vorba de vreo 124 MB. Vă recomand să luați manualul în format PDF şi arhiva (`lfs-packages-5.0.tar`) ce conține toate sursele şi patch-urile aferente, pentru că altfel vă va fi puțin mai greu să descărcați fiecare arhivă individual (nu de alta, dar sunt cam multe: 83 fişiere). Dacă ați descărcat arhiva `lfs-packages-5.0.tar`, vă sugerez ca după despachetare să verificați integritatea fiecărei arhive în parte:

```
$tar zxf lfs-packages-5.0.tar
$cd lfs-packages-5.0
$bzip2 -tv *.tar.bz2
```

Odată ce ați intrat în posesia surselor şi manualului, putem trece mai departe. În primul rând citiți manualul ca să vă faceți o imagine de ansamblu, să vă acomodați cu terminologia şi cu modul de lucru.

Procesul de compilare a LFS este împărțit în două etape şi necesită în prealabil existența unei alte distribuții Linux (în principiu, orice distribuție). Motivul: pentru a compila/instala un compilator aveți nevoie de un compilator (am citat din manual). În prima parte (la capitolul 5 din manual) se vor compila utilitarele (cu bibliotecile incluse în executabile), iar în a doua parte (capitolul 6) se va pregăti efectiv instalarea.

Pentru a evita unele neplăceri pe parcursul compilării, aveți nevoie de o partiție separată. Se poate şi pe aceeaşi partiție pe care este distribuția dumneavoastră, dar trebuie să faceți câteva modificări. Dacă sunteți începător sau este pentru prima dată când compilați LFS, cel mai indicat este să folosiți o partiție separată. Partea proastă este că vă trebuie una destul de mare, cu o capacitate mai mare de 2,5 GB. Dacă nu aveți acest spațiu, lucrurile se complică puțin, după cum urmează:

- fiecare pachet, după ce a fost compilat şi instalat, va fi şters (unde este posibil);
- informațiile de depanare (debug) vor fi eliminate imediat după ce a fost instalat respectivul pachet, şi nu la sfârşitul capitolului 5, cum scrie în manual. Se obține astfel spațiu prețios din timp.

Pentru compilarea nucleului, încercați să folosiți compilatorul gcc-2.95.3 (inclus în distribuția LFS), iar pentru încărcarea efectivă a sistemului, folosiți bootloader-ul distribuției gazdă (mai multe detalii în articolul Boot-load din numărul precedent al linux360).

După ce am finalizat instalarea, să vedem ce am obținut: un sistem Linux, care nu face aproape nimic. Nu putem uita însă faptul că, odată parcurşi paşii prezentați în manual, ne vom clarifica foarte multe aspecte ce țin de acest sistem de operare, cum depind pachetele undele de altele astfel ca în final să formeze un tot unitar. Priviți-l ca pe un curs de lungă durată, care nu se ştie, s-ar putea să va prindă foarte bine.

Abia după ce s-au înțeles aceste aspecte, putem trece la un nivel superior, şi anume cum să ne construim într-adevăr sistemul pe care ni-l dorim.

Optimizări

Având în vedere că noi construim sistemul numai din cod sursă, vă recomand să compilați cu optimizările pentru procesorul dumneavoastră. Unele pachete incluse în LFS auto-detectează cele mai bune setări în timpul auto-configurării, în timp ce la altele ar fi indicat să le specificăm noi. În cazul de față, folosind comenzile de mai jos, vom seta câteva opțiuni pentru compilator:

```
$export CFLAGS="-O3 \
-march=<'tip_procesor> "
$CXXFLAGS=$CFLAGS
```

unde <tip_procesor> poate lua valorile: i386, i486, i586, i686, pentium, pentiumpro, pentium4, k6, athlon (Athlon, Duron).

Dacă aveți un procesor, să spunem 486, şi ați specificat la tipul procesorului unul dintr-o clasă superioară, să nu vă mirați că nu rulează respectivele aplicații (deşi compilarea a decurs perfect). Invers se poate.

Atenție la aceste optimizări. Nu întotdeauna sunt binevenite. De exemplu, pentru compilarea GCC-ului (indiferent de versiune) trebuie să lăsăm pachetul în pace, adică să nu specificăm nici un fel de optimizare.

Alte optimizări ce se mai pot aplica:

- `funroll-loops` (se determină numărul de iterații la momentul compilării - evident, unde se poate)
- pentru un număr maxim de optimizări ce se pot obține: `$export CFLAGS="-s -O3 -fomit-frame-pointer \`
- `-march=<tip_procesor> -malign-functions=4 \`
- `funroll-loops -fexpensive-optimizations -malign-double \`
- `fschedule-insns2 -mwide-multiply"`
  $CXXFLAGS=$CFLAGS

Optimizările de mai sus nu vor funcționa întotdeauna. Unele pachete nu vor putea fi compilate, sau vor rula incorect (se pot bloca la un moment dat). Cel mai bine folosiți optimizările minimale, ca să fiți siguri că nu vor apărea surprize.

Argumente pro şi contra LFS

Am să enumăr câteva argumente pro şi contra folosirii unei distribuții bazată pe LFS.

Din punctul de vedere al unui utilizator începător de Linux:

- va învăța şi înțelege cum se compilează aplicațiile cele mai importante;
- ce aplicații conține un anume pachet;
- care sunt, unde şi ce conțin cele mai importante fişiere de configurare;
- va înțelege procesul de boot-are al unui sistem Linux.

Pentru un utilizator avansat, care foloseşte Linux-ul în cadrul unei firme/companii:

- timp mult prea mare alocat procesului de compilare;
- upgrade-ul greoi;
- lipsa suportului tehnic.

Dacă tragem linie şi adunăm, vedem că cei mai avantajați sunt utilizatorii de nivel începător/mediu. Pentru un utilizator avansat, LFS-ul ar fi avantajos la modul următor:

- are deja salvată prima parte din LFS (cu utilitarele compilate static);
- rulează un script creat în prealabil ce compilează (cu optimizările aferente) a doua parte din LFS;
- instalează "la mână" încă vreo 4-6 pachete (dacă depăşeşte acest interval, deja se gândeşte serios la o distribuție binară).

Stația respectivă va rula cel mai probabil 2-3 servicii, şi cam atât. Dacă se va pune problema de upgrade... se va face într-adevăr upgrade, dar cu o distribuție binară.

Eu vă recomand să încercați măcar o dată LFS, indiferent dacă sunteți un utilizator începător sau unul avansat.

Resurse:

- www.lfs.org

Autor:

andrei.ciubotica@linux360.ro

Am voie sau nu am voie? - Licențe software

Ciprian Negrilă

Ca utilizatori ai unui computer, sigur ați întâlnit un program care, în timpul procesului de instalare, v-a prezentat un text în care era scrisă licența. Acest articol va explica într-un mod cât mai detaliat licențele des întâlnite într-un mediu Linux.

Despre ce vorbim?

Licența este un text în care sunt scrise drepturile şi obligațiile utilizatorilor. Există totuşi o dispută: pe de-o parte producătorii programului au dreptate pentru că este programul lor şi noi ca şi utilizatori suntem obligați să folosim programul aşa cum specifică ei şi cum este specificat în contractul de cumpărare al programului (însa de multe ori acest contract nu există). Pe de altă parte, utilizatorul nu este interesat de doleanțele creatorilor, ci doar de modul în care programul îi uşureaza munca/existența. Având în vedere aceste două perspective, cineva, undeva, în lumea aceasta imensă a reuşit sa facă un compromis. Richard Stallman a inițiat proiectul Free Software Foundation. Majoritatea ați auzit de acest proiect, fie prin asocierea cu proiectul GNU, fie vis-a-vis de termenul Open Source. Multă lume confundă termenul Open-Source cu GPL (General Public Licence), deoarece programele Open-Source sunt publicate sub licență GPL sau LGPL. La rândul lor, aceste două tipuri de licențe sunt confundate, deoarece diferă abia vizibil din punctul de vedere al unui utilizator care nu-şi iroseşte timpul citind licențe de pagini întregi.

Pas cu pas

Primul aspect ce trebuie explicat este modul în care sunt eliberate programele Open-Source şi sub ce reguli sunt ele distribuite. Programele Open-Source diferă de restul programelor prin faptul că alături de binare (fişierele necesare rulării programului) sunt distribuite şi sursele care au dus la formarea binarelor. Asta înseamnă Open-Source şi nimic mai mult.

Acum, că am clarificat acest lucru, putem elabora un pic conținutul licenței publice generale (GPL). Licența GPL se aplică oricărui program care este distribuit cu specificarea de orice natură că se supune ei. Astfel eu pot să dau un program (care nu are nici un fel de licență anterioară) cuiva şi să-i spun că se află sub licență GPL. În mod oficial acel program devine Open-Source. Dacă, însă, programul nu are sursele alături de binare, el nu poate fi pus sub licența GPL. Confuzia provine din faptul că orice program distribuit sub GPL este Open-Source, însă nu orice program Open-Source este distribuit sub licență GPL. Deoarece licența GPL se referă numai la modificarea, copierea sau distribuirea programelor împreună cu sursele lor, ea nu se aplică decât acestor acțiuni. Unul dintre cele mai importante este regula distribuției. Confuzia este creată de prezența termenului "free" în componența textului licenței. În primele variante ale licenței, textul nu includea şi o explicație a acestui termen. El putea fi interpretat ca fiind "liber" sau "gratis". Această dispută a fost încheiată o dată cu specificarea aparută ulterior, conform căreia termenul se referă la libertatea de a folosi programul. Drept consecință, există posibilitatea ca pentru serviciul efectuat să fie percepute taxe.

O a doua problemă întâlnită este cu copierea unei bucăți de cod. În acest caz, utilizatorul trebuie să păstreze licența liniilor de cod introduse în program, fiind obligat să le ataşeze sursa alături de program. Acest lucru este foarte rar respectat, deoarece lumea se gândeşte că numai programul întreg se supune GPL. Deci vă sugerez să aveți grijă mare la detalii, pentru că cei de la FSF (Free Software Foundation) sigur au.

Surioara vitregă

Versiunea curentă a GNU GPL este 2.0. Ea a rămas neschimbată de ani de zile, însă în anul 1999 a apărut o subramură a acestei licențe, care se ocupă numai de biblioteci şi componente funcționale ale unui program. Aceasta poate fi considerata (şi este - în accepțiunea generală a Free Software Foundation) ultima versiune a GPL (2.1), deşi se referă numai la un tip restrâns de binare. Regulile sunt schimbate foarte puțin, majoritatea adăugirilor sau schimbărilor fiind făcute tot din motive de neclaritate. Această licență (LGPL) nu este cu nimic inferioară licenței GPL, deoarece înglobează aceleaşi concepte de bază. De asemenea, aceste concepte sunt aplicate un pic diferit. Cea mai mare schimbare este specificarea de noi reguli pentru integrarea codului într-un program eliberat şi distribuit sub licență proprie. Totuşi să nu pierdem din vedere un lucru: LGPL se aplică NUMAI pentru bibliotecile şi componentele funcționale ale programelor. Pentru programele întregi se foloseşte în continuare GPL-ul.

Sărbătoarea Open-Source

După această mică analiză, putem face o adunare şi observăm câteva aspecte: diferența dintre un program Open-Source şi un program eliberat sub licență GPL, diferența între cele două licențe GNU şi modul în care se aplică ele. Adoptarea sistemului Open-Source cu licență GPL a dat naştere sistemului de creare interactivă a programelor. Acest sistem trebuie respectat ca atare, ca pe o revoluție, pentru că asta şi este.

Autor:

ciprian.negrila@linux360.ro

Despre felul în care lucrurile vechi pot deveni noi - Mandrake Linux

Adrian Berindei

Motivație

Vă voi spune, prieteni, care este ideea din spatele acestei distribuții si despre ce a fost şi va fi Mandrake Linux. Nu voi descrie proceduri de instalare şi nu voi răspândi sfaturi înțelepte garnisite de comenzi ascunse menite a vă face fericiți în odiseea cunoaşterii Linux-ului.

Să fie... Mandrake

Ca origine, Mandrake Linux este o distribuție dezvoltată in Franța. Părintele acesteia este Gael Duval şi el a avut ca bază de plecare distribuția Red Hat 5.1! Da, ați citit bine, Mandrake este la origini un... Red Hat modificat. Mandrake a apărut prima oară în 1998, mai exact pe 23 iulie, prima sa versiune fiind 5.1 cu numele Venice. Venice venea cu kernel 2.0.34 şi KDE 1.0. În acelaşi an a fost fondată şi compania numită Mandrake Soft. Scopurile Mandrake prezentate la vremea respectivă în mesajul care anunța apariția Venice, au rămas valabile până astăzi, ele conturând o definiție foarte bună pentru distribuție:

- să ofere un mediu funcțional şi uşor de instalat pentru utilizatorii care nu vor să piardă mult timp instalând si configurând sisteme Linux
- să ofere un sistem Linux atractiv şi uşor de folosit pentru începătorii veniți de pe alte sisteme de operare
- să ofere o distribuție nouă de Linux

Primele versiuni s-au succedat rapid şi au cunoscut un succes incredibil. Inovația a început să îşi facă loc definind distribuția. Era ceva nou, un Linux uşor de folosit, accesibil tuturor şi a cărui interfață grafică pornea automat.

De ce Mandrake

Motivația numelui distribuției este încă incertă, "mandrake" însemnând mătrăgună (mandragora), o plantă cu rădăcină în formă de om (aparent) şi care este mai mult întâlnită în lucrările legate de magie decât de Linux. În evul mediu era considerată o plantă cu proprietăți curative. A fost folosită mult timp ca anestezic încă de pe vremea lui Pliniu şi mai târziu în Evul Mediu. Este cunoscută de mult, poate de când lumea, primele mențiuni fiind întâlnite în Biblie:

Geneza 30:14 - Ruben a ieşit odată afară, pe vremea seceratului grâului, şi a găsit mandragore pe câmp. Le-a adus mamei sale, Lea. Atunci Rahela a zis Leei: "Dă-mi, te rog, din mandragorele fiului tău."

Geneza 30:15 - Ea i-a răspuns: "Nu-ți ajunge că mi-ai luat bărbatul, de vrei să iei si mandragorele fiului meu?" Şi Rahela a zis: "Ei bine! poate să se culce cu tine în noaptea aceasta, pentru mandragorele fiului tău."

Geneza 30:16 - Seara, pe când se întorcea Iacov de la câmp, Lea i-a ieşit înainte şi a zis: "La mine ai să vii, căci te-am cumpărat cu mandragorele fiului meu." Şi în noaptea aceea s-a culcat cu ea.

Mandrake Linux

La momentul scrierii acestui articol, Mandrake a ajuns la versiunea 9.2. Este considerat încă, la mai mult de 5 ani de la apariție o distribuție destinată începătorilor, şi şi-a menținut reputația de sistem Linux uşor de folosit. Se pare că scopurile propuse în 1998 au fost respectate. Dincolo de asta, este una dintre distribuțiile cele mai iubite, având după spusele celor de la Mandrake Soft mai mult de 4 milioane de utilizatori. Lucrul acesta se vede şi în numărul site-urilor dedicate. Versiunea download vine pe trei discuri şi veți găsi în ea foarte multe pachete. Ce se remarcă la aceste pachete este nu atât faptul ca sunt în general versiuni foarte noi ale diverselor programe, dar pot fi chiar şi versiuni în dezvoltare. Aceasta tendință către foarte nou a făcut în timp diferența între Mandrake şi Red Hat, cel din urmă fiind recunoscut pentru conservatorismul celor care îl realizează. Pe de altă parte, privite atent, pachetele vor arăta o alta tendința specifică Mandrake, şi anume prezența unor dependențe necesare instalării unor pachete care nu se află pe aceste discuri. Acest lucru se datorează în principal sistemului prin care Mandrake oferă utilizatorilor softul pregătit de companie. Orice utilizator se poate înregistra contra sumei de 5$ pe luna în ceea ce se numeşte Mandrake Club, de unde poate descărca mii de pachete suplimentare oficiale. Pachetele dispun şi de o semnătura electronică, calitatea lor fiind garantată de companie. În paralel, utilizatorii pot descărca aceleaşi pachete compilate de alte persoane, fără a fi însă garantată calitatea acestora.

Când vine vorba de soft, aveți distribuția care oferă cam tot ce îşi poate dori un user. La câteva zile după release-uri, pe FTP-urile consacrate apar mii de pachete compilate pentru versiunea respectivă. De asemenea, există o serie de site-uri cum ar fi Penguin Liberation Front care se ocupă de compilarea pachetelor cu licențe incerte şi care nu pot intra în distribuție din motive legale. Sistemul prin care se face selecția pachetelor este acela de vot, o dată intrat in Mandrake Club poți acorda voturi aplicațiilor pe care le doreşti incluse pe discuri. Aplicațiile votate, dar care nu mai încap in distribuție pot fi descărcate de pe Internet. Pachetele trec prin mai multe faze, de votare, de test şi de validare. Utilizatorii pot şi argumenta de ce merită sau nu merită să fie preparat rpm-ul respectiv, şi mai ales de ce ar trebui inclus

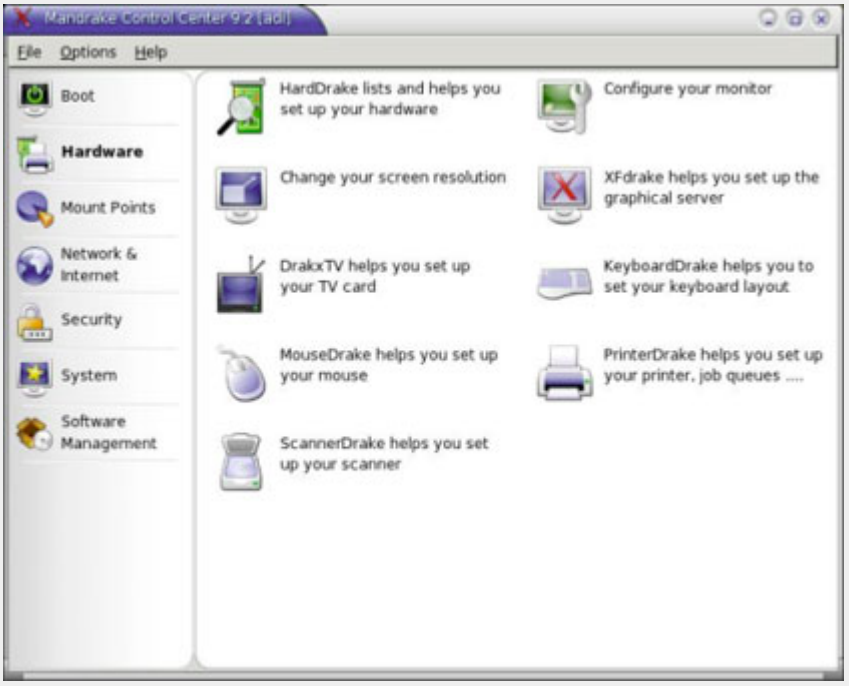
în distribuție (aceasta este şi maniera în care Mplayer a apărut pe discuri, fiind cea mai votată aplicație).

Ce îl face diferit

Principalele componente ale distribuției se găsesc în ceea ce se numeşte Mandrake Control Center sau pe scurt MCC. Ele se constituie într-o serie de instrumente grafice destinate configurării sistemului, asemănător oarecum cu Yast de pe SuSE. De remarcat însă faptul că toate componentele MCC au şi versiuni de consolă, aşa încât nu depindeți de X în folosirea lui. Se lansează apelând din meniu secțiunea `Configuration-> Configure your computer` sau din consolă rulând comanda `drakconf`. Cei mai avansați în Linux vor spune că acest Control Center opreşte userul de la înțelegerea adevăratelor fenomene din spatele Linux. Nu sunt întru totul de acord, deoarece până la urma folosirea lui este opțională şi mai devreme sau mai târziu, dacă un vrăjitor de aici nu merge, ajungi să configurezi manual. Am folosit de multe ori această facilitate şi per ansamblu o consider ca fiind un mediu care poate aduce useri pe Linux, deoarece este mai "prietenoasă" decât consola şi mult mai intuitivă. De la o versiune la alta sunt incluse opțiuni noi sau sunt îmbunătățite cele vechi. Nu voi trece în revistă toate utilitarele, dar voi menționa câteva, mai importante, şi care fac din utilizarea Mandrake o joacă de copil.

La secțiunea Boot: puteți realiza automat o dischetă de boot folosind DrakFloppy sau puteți configura felul în care sistemul bootează folosind DrakBoot. De aici puteți alege dacă doriți ca interfața grafică să pornească automat sau nu, dacă doriți autologin şi de asemenea mediul de lucru dorit. De asemenea puteți configura destul de comod şi intuitiv managerul de boot.

La secțiunea Hardware: primul este Harddrake. Acesta îşi face datoria cu brio, ori de câte ori aduceți hardware nou în calculator fie că este vorba de un simplu HDD sau de altă componenta. La primul



MCC, secțiunea Hardware

boot, va semnala hardware-ul nou, şi vă va oferi posibilitatea să îl configurați, fie că este vorba de un HDD, o unitate CD-ROM sau o placă de rețea. Puține sunt cazurile în care după setup componentele nu sunt detectate si setate corect. Personal, am avut ocazia sa văd harddrake la lucru pe mai multe sisteme si m-a impresionat de fiecare dată. Încă un aspect interesant este acela că HDD-urile primesc automat setările corecte cu ajutorul `hdparm`.

Tot aici veți găsi utilitare destinate diverselor configurări de hardware, fie că este vorba de serverul grafic, mouse sau scanner-ul pe care îl deținеți. De cele mai multe ori aceste instrumente nu vă scutesc de cunoaşterea parametrilor componentelor, setările de finețe trebuind să fie introduse manual. Un instrument binevenit este DrakxTV care oferă posibilitatea configurării plăcilor PC-TV chiar şi de către un începător, scutindu-i de multe dureri de cap pe cei mai puțin familiarizați cu lumea Linux.

La secțiunea Mount Points: veți găsi DiskDrake, un utilitar de partiționare extrem de uşor de folosit cu ajutorul căruia se poate realiza structura de partiții dorită pe un HDD. DiskDrake este şi o interfață grafică pentru `/etc/fstab`, în sensul că puteți defini aici comod puncte de montare sau diverşi parametrii. La boot toate partițiile, inclusiv cele cu sistem de fişiere FAT, vor fi accesibile, astfel încât începătorii se vor simți ca acasă. Vor regăsi în `/mnt` toate partițiile FAT, montate în `win_c, win_d etc.` în funcție de litera corespunzătoare partiției în Windows. La instalarea sistemului puteți alege tipul de sistem de fişiere dorit iar cu `diskdrake` puteți partiționa HDD-ul extrem de uşor, existând şi posibilitatea folosirii spațiului liber pe disk daca nu doriți să pierdeți datele pe care le aveți de pe alt sistem de operare.

O altă componentă importantă este Samba mount points. Ea vine şi dă un răspuns şi o soluție începătorilor care vor să comunice cu maşini pe care rulează Windows. Este un utilitar excelent şi în principiu poate înlocui cu succes LinNeighborhood. Echivalentul lui pentru maşini care rulează Linux este NFS `mount points`. El vă afişează toate share-urile din rețea, vă permite să definiți puncte de montare pentru ele şi apoi sa le montați efectiv cu opțiunea de a salva setările în `/etc/fstab`. Supermountul este o altă facilitate Mandrake care constă în montarea demontarea automată a unităților CD-ROM. O facilitate destul de controversată, mai ales datorită unor

versiuni (de ex. 9.0) unde nu a funcționat corect, este acum perfect funcțională şi utilizatorii începători o vor aprecia mult. O surpriză plăcută în 9.2 a fost faptul că la introducerea unui Card Reader de medii Compact Flash în sistem, acesta a folosit supermount pentru el. Daca nu vă place această facilitate, vi se oferă posibilitatea de a renunța la ea din setările avansate pentru CD-ROM sau din consolă folosind comanda `supermount -i disable` (pentru a-l reporni `supermount -i enable`).

Categoria System: oferă o serie de instrumente mai "mărunte" dar binevenite în orice sistem Linux.

De aici, folosind,drakxservices puteți să alegeți serviciile care să pornească la boot, şi de asemenea puteți porni/opri servicii. Userdrake şi Menudrake vă permit să administrați userii şi respectiv să administrați aplicațiile din meniul sistemului. DrakFont este un utilitar pentru instalare de fonturi. Pe lângă faptul că vă oferă posibilitatea să instalați fonturi de pe diverse medii, interesantă este opțiunea "Get Windows fonts", prin care sistemul preia automat fonturile din MS Windows dacă aveți şi un astfel de sistem de operare instalat. În plus vă permite să selectați anumite aplicații pentru care doriți ca fonturile să fie disponibile cum ar fi Star Office sau Abi Word.

La categoria Software Management veți găsi Rpmdrake. Acesta este managerul de pachete rpm şi este după părerea mea o aplicație excelentă. El vă va uşura munca atunci când vine vorba de instalat / dezinstalat pachete rpm. Puteți aduce foarte uşor în sistem pachete suplimentare, el rezolvând automat dependențele în cazul în care acestea se află printre pachetele disponibile. Ca să vă faceți o idee despre felul în care lucrează vă dau un exemplu interesant. Dacă descărcați de pe Internet un pachet rpm şi încercați să îl instalați în sistem, rpmdrake rezolvă singur dependențele dacă acestea se află printre pachetele pe care le deține. De asemenea, dacă aveți nişte pachete rpm pe un CD sau într-un director pe HDD le puteți aduce extrem de uşor alături de pachetele din distribuție folosind opțiunea Software Media Manager. Rpmdrake ne-a permis să scoatem pentru 9.2 un disc suplimentar cu aplicații şi pe care oricine îl poate folosi ca şi cum ar fi parte a distribuției. Începând cu distribuția 9.0, Rpmdrake a fost despărțit în două aplicații diferite, de instalare şi dezinstalare de programe. Personal nu consider această variantă ca fiind cea mai flexibilă şi preferam să fie o singură aplicație. Părerile sunt împărțite, iar astfel de modificări sunt în general frecvente de la o versiune la alta. La categoria Network & Internet şi Security vă puteți seta rețeaua şi conexiunea la Internet. Destul de intuitive, aplicațiile respective sunt binevenite pentru utilizatorii de acasă, care se conectează pe rețele locale şi nu doresc nişte setări avansate ale sistemului.

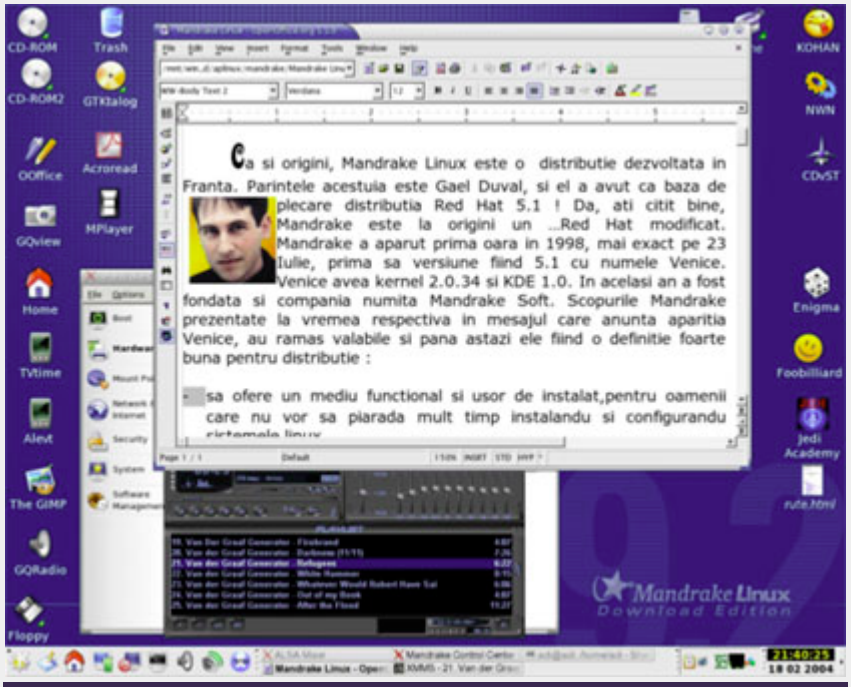
MCC, categoria System

Un Linux pentru desktop?

De foarte multe ori veți auzi că Mandrake este o distribuție pentru desktop. Într-adevăr, pachetele pe care le veți găsi pe CD-urile distribuției pot mulțumi orice utilizator. Veți găsi editoare de text, aplicații multimedia, soft de ars CD-uri, pachete office, pachete CAD, jocuri, soft de monitorizare. La loc de cinste veți găsi Mplayer, scutind astfel de griji o mulțime de începători. De asemenea, este o distribuție care oferă mai toate managerele de ferestre şi mediile de lucru pe care le poate dori cineva, oferind astfel flexibilitate în configurarea şi setarea aspectului grafic al distribuției. Pentru Mandrake, mediul KDE reprezintă ceea ce Gnome este pentru Red Hat. Nu de puține ori au existat discuții pe Internet despre viteza excelentă a KDE pe această distribuție, ea fiind rezultatul unor patch-uri aplicate de Mandrake peste KDE-ul oficial. Tema Galaxy şi-a făcut apariția o dată cu MDK 9.1 şi este tema oficială MDK în binecunoscuta culoare albastră care defineşte distribuția. Se vrea a fi un echivalent al temei BlueCurve de pe Red Hat şi aduce un aspect comun pentru aplicațiile destinate diverselor medii de lucru.

Structura meniului este foarte intuitivă şi orice utilizator, chiar şi începător, poate găsi uşor aplicațiile dorite. Dacă nu vă place felul în care Mandrake structurează aplicațiile în meniu, vi se pune la dispoziție menudrake, program cu care puteți să editați meniul din câteva click-uri.

Pasionații de sunet vor avea de asemenea de ce să se bucure deoarece vor găsi pe Internet mai toate programele audio de Linux compilate pentru el, inclusiv celebrul Ardour. De asemenea am găsit soft de prelucrare video, playere, mixere, într-un cuvânt cam tot ce există pe Linux şi ține de multimedia. Tot ceea ce înseamnă



Un desktop Mandrake într-o sesiune de lucru

teme şi iconițe sunt de asemenea pregătite pentru el şi le puteți folosi extrem de uşor, schimbând felul în care arată sistemul funcție de preferințele voastre. Mecanismul de votare al pachetelor şi tendința utilizatorilor Mandrake de a folosi programe din domenii oarecum puțin "acceptate" în Linux, îl fac mereu să aducă ceva nou de la o versiune la alta. Un aspect inedit a fost acela al includerii driverelor accelerate de la NVIDIA şi ATI pentru versiunile comerciale ale distribuției. Rolul Mandrake este acela de poartă de intrare pentru utilizatorii care migrează pe acest sistem de operare. Prin felul în care este gândită distribuția, ea îşi îndeplineşte rolul cu succes. În general, cei care avansează în cunoaşterea Linux trec prin el spre alte distribuții, şi nu o dată am întâlnit persoane care au început cu el şi care îi recunosc meritele şi îl sprijină chiar dacă nu îl mai folosesc. Până la urmă toate distribuțiile slujesc aceluiaşi scop şi aceloraşi idealuri, chiar dacă structura lor diferă mai mult sau mai puțin.

Un MDK 9.2 cu probleme

Dincolo de lucrurile bune, distribuția 9.2 are şi câteva probleme neplăcute care trebuie subliniate. Primul bug important este acela că unele unități LG se "defectează" în timpul instalării distribuției. Defectul nu este iremediabil şi se datorează in principiu unei erori de soft din firmware-ul anumitor unități. Nu este nimic grav, deoarece o unitate afectată în timpul instalării poate fi refăcută prin rescrierea firmware-ului. LG a pus la dispoziție un firmware destinat MDK 9.2 precum şi procedura de actualizare a acestuia, iar pe site-ul Mandrake veți găsi mai multe detalii precum si lista unităților afectate. Am avut ocazia sa trec prin această experiență şi simptomele se manifestă prin aceea că în timpul instalării sistemul nu mai citeşte discurile, iar după repornire unitatea nu mai este detectată de sistem. Refacerea firmware-ului durează câteva secunde şi nu "doare" deloc. De asemenea există şi un kernel destinat 9.2 care evită problema, este vorba de kernel-2.4.22-21mdk

Al doilea bug celebru este acela legat de dispariția aplicațiilor din meniul KDE. Se poate întâmpla ca după instalarea anumitor pachete, aplicațiile din meniul principal să dispară. Soluția provizorie este să rulați din consolă comanda update-menus. Rezolvarea definitivă a problemei vine tot de la Mandrake si constă în instalarea a cinci pachete rpm responsabile de dispariția meniului.

Despre viitor şi speranță

La momentul scrierii acestui articol este în pregătire versiunea 10.0 a distribuției. Toți aşteaptă o distribuție "istorică", plină de inovații, aşa cum a fost 9.0. După rezolvarea problemelor financiare pe care le-a avut anul trecut, după o distribuție 9.2 care a avut unele probleme neplăcute, Mandrake trebuie să scoată un produs deosebit. Se spune că 10.0 este o distribuție în care Mandrake nu are voie să mai dea greş, fiind un "to be or not to be" al companiei. Acesta va include o nouă temă numită Galaxy 2 iar supermountul a dispărut fiind înlocuit cu MagicDev. De asemenea va avea KDE 3.2 şi un kernel 2.6.3, iar MCC va fi refăcut şi reorganizat. Viitorul lor depinde de această reuşită. Într-o piața în continuă transformare în care Linux devine comercial, ei încearcă să ofere un produs "gratuit" competitiv. Este un nou an şi o noua procedură de lansare îşi face simțită prezența. Pentru a evita problemele de genul celor apărute în 9.2, noul Mandrake va fi lansat în mai multe etape. Într-o primă fază, după trecerea de Beta şi RC, va apărea un Mandrake Linux Community Edition. Abia la o lună sau două după ce această versiune va fi folosită intens şi toate vulnerabilitățile vor fi cunoscute, va apărea versiunea finală. Acest procedeu va conduce categoric la apariția unor versiuni net superioare. Va reuşi Mandrake să supraviețuiască în competiția cu Red Hat/Fedora şi SuSE? Toți fanii speră ca da şi eu sunt alături de ei. Să ne revedem iarăşi la un test Mandrake 10.0 şi să sperăm că veți avea ce citi. Numai de bine tuturor pasionaților de Linux din România şi de pretutindeni.

Resurse:

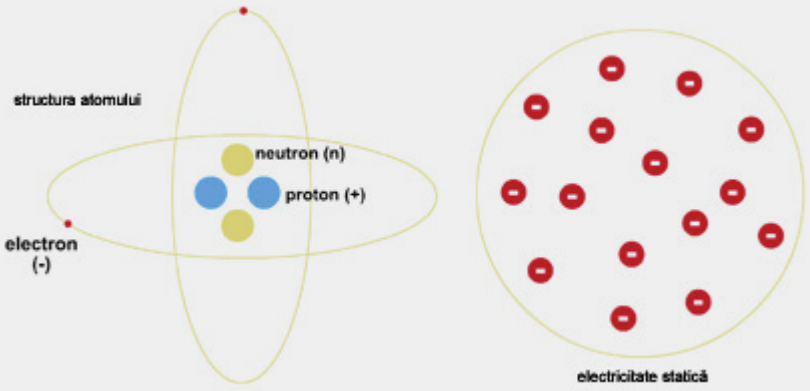
- www.mandrakenation.ro
- www.mandrake.com

Autor:
adrian.berindei@linux360.ro

O necesitate numită rețea

Daniel Secăreanu

Am vorbit în numerele trecute despre cele şapte nivele ale modelului OSI sau cele patru nivele ale protocolului TCP/IP, precum şi despre diferite tipuri de topologii şi echipamente de rețea. În continuare, vom aborda mai amănunțit fiecare nivel al modelului OSI, pentru a încerca să înțelegem mecanismele ce funcționează la fiecare dintre aceste nivele.



Atomul: protoni, neutroni, electroni şi electricitate statică

Vom începe de la baza modelului OSI şi anume de la nivelul 1 sau nivelul fizic. Funcțiile acestui nivel al modelului OSI sunt de a transmite informații prin stabilirea unor specificații electrice între sursă şi destinație. Informațiile percepute de către noi sub formă de imagini, text, audio sau video călătoresc prin mediul de transmisie sub formă de impulsuri electrice, în cazul firelor de cupru, sub formă de impulsuri luminoase, în cazul fibrei optice, sau sub formă de impulsuri electromagnetice în cazul comunicațiilor wireless.

Pentru a înțelege modalitatea de transport a impulsurilor electrice, luminoase sau electromagnetice, trebuie să definim câteva noțiuni de bază de chimie şi fizică. Deşi, poate pentru majoritatea dintre voi acestea sunt noțiuni de bază, am ales totuşi să le prezentăm pe scurt, pentru orice eventualitate.

Cum probabil toată lumea ştie, un atom este alcătuit din nucleu, protoni, neutroni şi electroni. Protonii sunt particule încărcate cu sarcină pozitivă, neutronii au o sarcină neutră iar electronii sunt particule încărcate cu o sarcină negativă.

Protonii şi neutronii sunt legați în jurul nucleului de o forță nucleară puternică ce nu le permite să se respingă, deşi particulele cu acelaşi tip de încărcătură se resping. Deşi electronii ar trebui, teoretic să fie atraşi de către protoni, ei au destulă viteză pentru a orbita în jurul nucleului, fără a fi atraşi de acesta. În acelaşi timp însă, forța care îi ține pe aceştia aproape de nucleu este o forță destul de slabă ce poate fi învinsă destul de uşor.

Acest lucru se poate întâmpla în cazul unor anumiți atomi, generând ceea ce numim electricitate. Electricitatea este de fapt această circulație liberă a electronilor. Electronii se pot afla însă şi într-o stare de repaus, numită şi electricitate statică.

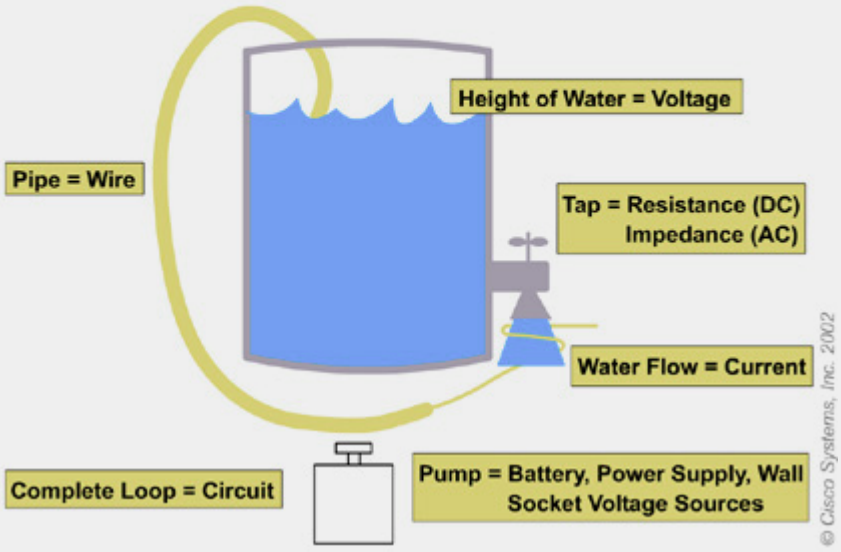
Electricitatea statică poate apărea, de exemplu, atunci când traversăm un covor sintetic într-o cameră uscată şi răcoroasă şi atingem un obiect metalic cu mâna. În acel moment simțim un mic şoc electric. Dacă pentru noi această descărcare electrostatică este infoensivă, ea poate cauza efecte dezastruoase în cazul cipurilor electronice sau a informațiilor din acestea.

În funcție de uşurința cu care electronii circulă prin materiale, acestea pot fi clasificate în trei categorii: conductori, semiconductori şi izolatori. Conductorii sunt materiale ce permit circulația facilă a electronilor. Exemple de conductori sunt cuprul, aurul şi argintul. Semiconductorii sunt materiale ce permit controlarea precisă a numărului de electroni ce circulă prin ele. Exemple de semiconductori includ carbonul, germaniul şi cel mai important semiconductor - siliconul. Izolatorii sunt materiale ce permit foarte greu sau chiar deloc circulația electronilor. Exemple de izolatori sunt plasticul, sticla, aerul sau lemnul.

Pe baza proprietăților electronilor şi a materialelor prin care aceştia circulă, putem defini mai multe noțiuni cum ar fi tensiunea, intensitatea, rezistența, impedanța sau împământarea.

Tensiunea (sau forța electromotoare) este forța electrică (sau presiunea) exercitată atunci când protonii şi electronii se separă. Forța creată împinge spre particulele încărcate cu sarcină opusă şi dinspre particulele încărcate cu aceeaşi sarcină. Tensiunea apare, de exemplu, într-o baterie unde activitatea chimică a acesteia face ca electronii să circule de la polul negativ spre cel pozitiv printr-un circuit extern. Tensiunea mai poate fi creată de frecare (electricitate statică), de magnetism (generatoare electrice) sau de energia solară. Tensiunea se notează în general cu litera U, are ca unitate de măsură voltul şi reprezintă lucrul

Analogia circulației electronilor cu un sistem de alimentare cu apă



mecanic depus pentru a separa protonii de electroni.

Curentul (intensitatea acestuia) reprezintă circulația electronilor printr-un circuit. Atunci când există tensiune şi un circuit, electronii circulă de la polul negativ (care îi respinge) înspre polul pozitiv (care îi atrage). Intensitatea se notează în general cu litera I, are ca unitate de măsură amperul şi reprezintă numărul de electroni ce trec printr-un anumit punct din circuit la un moment dat.

Există două tipuri principale de curent: curent alternativ şi curent continuu. În cazul curentului alternativ, polaritatea acestuia variază în timp, schimbând astfel şi direcția în care circulă electronii. Exemple de curent alternativ avem în prizele de perete. În cazul curentului continuu, polaritatea acestuia rămâne aceeaşi, electronii circulând întotdeauna în aceeaşi direcție. Curentul ce circulă într-o baterie este un exemplu de curent continuu.

Rezistența reprezintă forța ce se opune circulației electronilor prin diferite materiale. Rezistența este proprietatea materialelor ce le diferențiază în cele trei categorii descrise mai devreme: conductori, semiconductori şi izolatori. Rezistența se notează în general cu litera R şi are ca unitate de măsură ohm-ul (simbolul omega). Rezistența se măsoară în general în cazul curentului continuu.

Impedanța măsoară opoziția combinată atât la curentul alternativ cât şi la cel continuu. Impendața este un termen general ce măsoară forța de rezistență opusă mişcării libere a electronilor. Impedanța se notează cu litera Z iar unitatea de măsură este, ca şi în cazul rezistenței, ohm-ul.

Legea lui Ohm defineşte relația dintre tensiune, intensitate şi rezistență în felul următor: I = U / R - intensitatea curentului ce trece printr-un conductor este egală cu tensiunea curentului împărțit la rezistența conductorului.

Împământarea poate reprezenta mai multe lucruri. În general, punctul de împământare este punctul în care o clădire atinge pământul, de preferat şi printr-o conexiune directă cu sistemul electric al clădirii. În cazul unei prize şi a unui conector electric, împământarea este reprezentată de al treilea picior al conectorului, oferind astfel electronilor o cale alternativă de a circula, alta decât corpul uman. Împământarea mai poate reprezenta şi punctul zero de tensiune, în cazul măsurătorilor electrice, sau punctul de referință zero în cazul unui multimetru, de exemplu.

Electronii circulă doar în circuite închise, sau circulare. Un exemplu simplu de circuit este o lanternă. Procesele chimice din baterie generează curent, permițând electronilor să circule dinspre polul negativ înspre cel pozitiv. Întrerupătorul este un punct în circuit în care acesta poate fi închis sau deschis. Becul din lanternă oferă rezistența necesară electronilor pentru eliberarea energiei sub formă de lumină. În cazul unui circuit închis, electronii călătoresc de la baterie spre bec, eliberând energia necesară aprinderii acestuia.

Vă întrebați, poate, de ce toate aceste noțiuni de electricitate într-un material despre rețele de calculatoare. Ei bine, principiile de bază ale funcționării rețelelor de calculatoare sunt foarte asemănătoare cu principiile de bază ale electricității, sau pe scurt, cu acest simplu circuit electric.

Imaginați-vă că, în sistemul client - server discutat într-unul din numerele anterioare, clientul este bateria ce generează fluxul de informații către server, acesta având capacitatea necesară analizării informațiilor şi trimiterii răspunsurilor înapoi spre client, într-un circuit închis. În cazul în care acest circuit ar fi deschis sau întrerupt într-un punct, circulația informațiilor nu ar mai avea loc.

De fapt, la primul nivel al modelului OSI avem de-a face cu transmisia biților prin intermediul mediului de rețea. Imaginați-vă faptul că biții nu sunt altceva decât impulsuri electrice, sau electroni călătorind printr-un material conductor. Echivalentul unui bit cu valoarea 1 ar fi existența impulsului electric, în timp ce echivalentul unui bit cu valoarea 0 ar fi lipsa impulsului electric.

Autor:

daniel.secareanu@linux360.ro

Tunele IP - origini, concept şi utilitate

Radu - Eosif Mihăilescu

Cu toții am văzut cel puțin o dată în viață un tunel. Poate când am mers cu trenul undeva am trecut printr-un tunel... când mergem cu metroul (în Bucureşti) parcurgem distanța dintre stații prin tunele şi exemplele ar putea continua.

Din punct de vedere conceptual, un tunel este un mijloc de a transporta o cantitate de "ceva" între două puncte în spațiu atunci când între cele două puncte mediul este "ostil" acelui "ceva".

Pentru a realiza aceasta, acea cantitate de "ceva" este mai întâi încapsulată într-un container compatibil cu mediul dintre cele două puncte. Apoi, containerul este trimis de la primul punct la cel de al doilea folosind infrastructura de transport standard a mediului dintre cele două puncte.

În final, cel de-al doilea punct primeşte containerul şi îl decapsulează obținând astfel "ceva"-ul inițial.

După cum probabil bănuiți, dacă cele două puncte reprezintă noduri de comunicație pentru alte puncte atunci pentru acestea din urmă existența tunelului este transparentă.

Revenind acum la cazul practic ce face subiectul prezentului articol, tunelele IP au rolul de a încapsula datagrame IP în alte datagrame IP (sub o formă oarecare) şi de a transmite datagramele-container astfel obținute de la un router la altul folosind informațiile de adresare de pe container (şi deci respectând "rigorile" mediului extern, adica al Internet-ului). Router-ul destinație decapsulează datagramele ce constituie tunelul şi obține un flux secundar de datagrame ce reprezintă informația tranzitată prin tunel - informație ce nu se deosebeşte cu nimic ca formă cu cea tranzitată prin oricare altă interfață de rețea.

Tipuri de încapsulări "automate"

Să vorbim acum de trei tipuri mai cunoscute de încapsulări aşa-zis "automate". Prin automate înțelegem faptul că de procesul de încapsulare/decapsulare se ocupă cod din nucleu (kernel) şi nu un proces extern acestuia.

- **IPIP** (IP-in-IP): această încapsulare foloseşte ca container datagrame clasice `IPv4`. Poate părea ciudat, dar pur şi simplu ia câte o datagramă destinată tunelului, o pune în zona de date a unei noi datagrame `IPv4`, completează câmpurile de adresă ale acesteia din urmă cu cele necesare pentru mediul extern şi o emite. Nimic mai mult. Aceasta este poate cea mai simplă încapsulare (şi de înțeles, şi de realizat, şi de configurat). Ea este folosită pe scară largă în rețelele de dispozitive mobile sau fără fir.
- **GRE** (Generic Routing Encapsulation): această încapsulare este identică cu IPIP cu unica diferență că în loc de a folosi datagrame IPv4 (protocolul 4), ea foloseşte datagrame GRE (protocolul 47) - în rest funcționarea este identică.
- **SIT** (Simple Internet Transition): această încapsulare este de o factură mai specială. Ea încapsulează datagrame `IPv6` (protocoalele 41, 43, 44, 50, 51, 58, 59 şi 60 adică `ipv6`, `ipv6-route`, `ipv6-frag`, `ipv6-crypt`, `ipv6-auth`, `ipv6-icmp`, `ipv6-nonxt` şi respectiv `ipv6-opts`) în datagrame IPv4. Ea este utilizată în procesul de tranziție, aflat în curs, de la IPv4 la IPv6.

Tipuri de încapsulări "speciale"

Prin "speciale" înțelegem încapsulări care nu intră în categoria anterioară fie din cauza formei speciale pe care o iau, fie pe motiv că sunt deservite de programe separate şi nu de codul nucleului sistemului de operare. Acestea vor fi tratate pe larg în numărul viitor în cadrul subiectului rețele virtuale private.

Avem, în principal, două astfel de încapsulări:

- **ESP** (Encapsulating Security Protocol): această încapsulare este folosită împreună cu AH (Authentication Header) în cadrul tehnologiei IPSec pentru a crea tunele criptate între două noduri din Internet.
- **PPP** (Point to Point Protocol): aceasta din urmă este folosită pe scară largă, în formă pură sau în combinație cu alte protocoale pentru o varietate largă de tunele.

Tunele IP "automate" în Linux

Pentru administrarea acestora se foloseşte programul `/sbin/ip` (din pachetul iproute versiunea 2), mai precis facilitatea "`tunnel`" a acestuia după cum urmează: "`/sbin/ip tunnel add tunl1 mode ipip remote <adresa distanta> local <adresa locala> dev interfata`" pentru a crea un tunel IPIP. Invocația este identică pentru GRE cu observația că "`ipip`" devine "`gre`" iar "`tunl1`"-"`gre1`".

În acest moment am obținut o interfață de rețea ce reprezintă capătul local al tunelului. Această interfață de rețea (tunl1 respectiv gre1) poate fi configurată ca şi oricare alta având toate proprietățile unei interfețe de rețea convenționale

Autor:

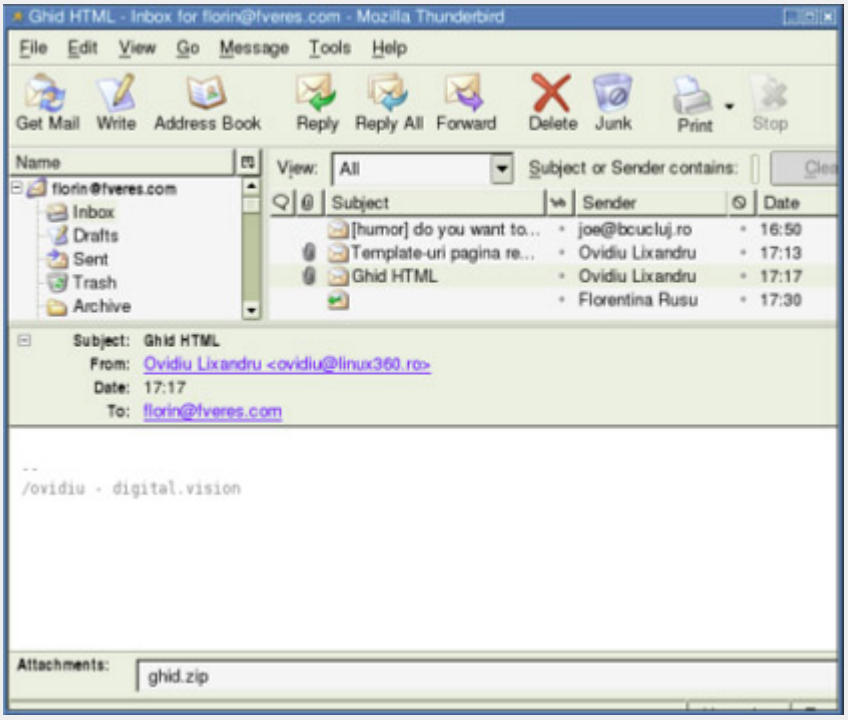
radu.mihailescu@linux360.ro

Poştaşul de încredere - test comparativ clienți de e-mail

Florin Vereş

În acest moment există foarte multe programe care îndeplinesc funcția de client de e-mail pentru Linux. În rândurile următoare vă vom prezenta principalele aplicații de gen, urmărind în paralel facilitățile acestora.

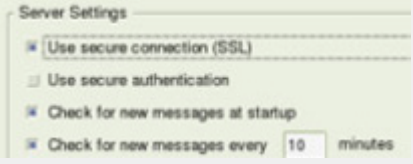
Cei mai cunoscuți şi, în acelaşi timp, utilizați clienți de mail sunt (în ordine alfabetică): Balsa, Ximian Evolution, Kmail, Mozilla Mail şi Mozilla Thunderbird. Unii din aceşti clienți rulează şi sub alte platforme (MacOS X, Windows, BeOS etc.)



Mozilla Thunderbird - unul din clienții creați de Mozilla Software Foundation

Tipuri de căsuțe mail suportate

Toți clienții de mail enumerați mai sus pot accesa directoare de poştă locale (mbox, maildir şi mh), IMAP, POP3, dar numai clienții de la Mozilla şi Ximian suportă IMAPS şi POP3S. Balsa are mici probleme cu protocolul IMAP (nu suportă listarea tuturor directoarelor de pe server, şi nici crearea de directoare noi).



Trimitere mail

Toate aplicațiile din test suportă SMTP, dar la capitolul comunicare cu serverul prin SSL/TLS, un singur client nu permite trimiterea securizată a e-mailurilor: Balsa.

Conturi multiple

Mozilla Mail, Thunderbird, Evolution şi KMail stau bine, putându-se ocupa de mai multe conturi simultan. Din păcate, nu se poate spune acelaşi lucru şi despre Balsa, căruia îi lipseşte această facilitate.

Corector ortografic

Toți "concurenții" stau bine la acest capitol, posibilitatea de corectare ortografică (în engleză) fiind prezentă.

Filtre

Toate aplicațiile suportă definirea personalizată a filtrelor pentru e-mailurile primite, produsele Mozilla având încorporat şi un filtru pentru spam foarte puternic.

Suport HTML

Deşi mesajele HTML nu mai sunt o noutate, unele programe de mail nu suportă acest format. Din fericire, toate aplicațiile din test pot crea şi afişa mesaje HTML.

Tipărire

Toate aplicațiile pot tipări e-mailurile primite, singura condiție fiind ca imprimanta dvs. să fie instalată corect.

Afişare şi compunere e-mailuri în diferite coduri de caracter

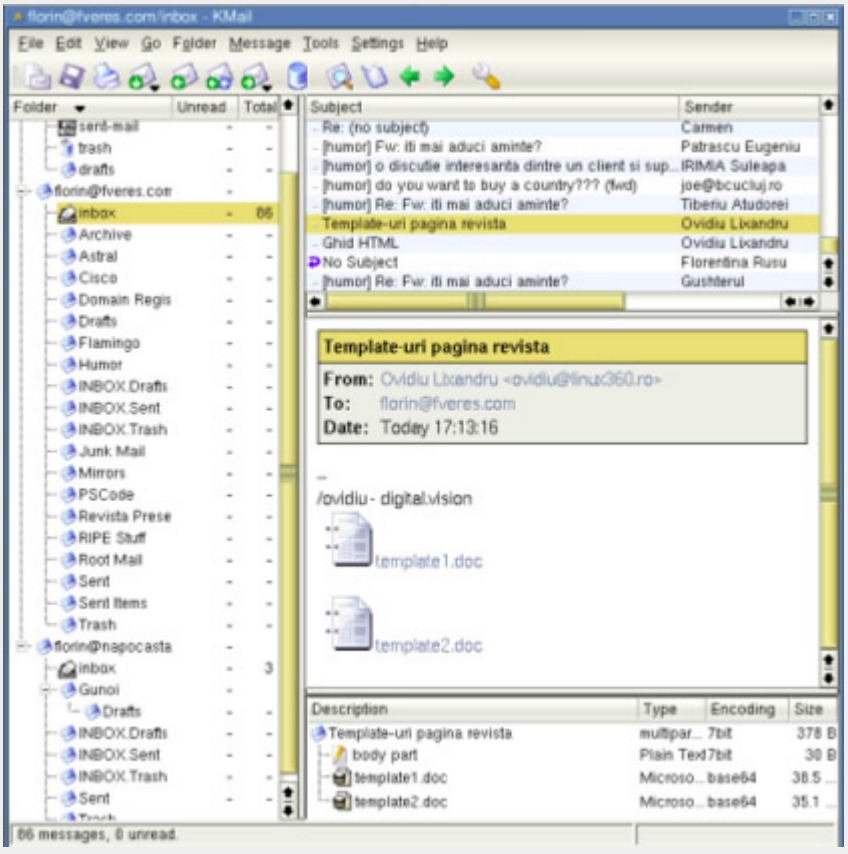
Toate programele din test au capabilitatea de a lucra cu diferite seturi de caractere, de la ISO-8859-2 până la UTF-8.

Agendă

Toate programele au încorporată o agendă, dar numai Mozilla Mail, Thunderbird, Ximian Evolution şi Balsa suportă LDAP.

Asistenți de configurare

Imediat după pornirea Mozilla Mail, Mozilla Thunderbird, Balsa sau Evolution, sunteți întâmpinat de un asistent care vă va ghida în configurarea inițială a clientului de mail. Din păcate, această facilitate nu este disponibilă la KMail.



KMail - clientul de poștă înglobat în KDE

Ştiri

Numai Ximian Evolution, Mozilla Mail şi Mozilla Thunderbird includ un News Reader (NNTP), prin care puteți asista şi participa la discuții pe diverse newsgroup-uri.

Teme şi extensii

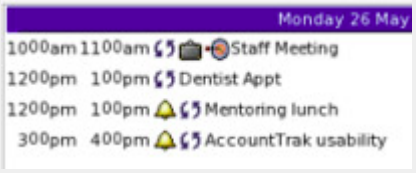
Produsele Mozilla (Mail şi Thunderbird) au capabilitatea de a fi personalizate prin teme şi extensii, care îmbunătățesc felul în care arată aplicația, dar şi adaugă diverse funcționalități. Pentru Evolution, cei de la Ximian au creat o extensie (comercială), Ximian Connector, care face posibil lucrul cu Evolution pe servere Microsoft Exchange.

Semnătură digitală şi criptare

Când vine vorba de securitate, toate aplicațiile din test suportă semnarea digitală a mesajelor, dar şi criptarea lor, folosind o cheie publică şi o cheie privată. Pentru a putea cripta sau a semna digital mesajele, va trebui să aveți instalat GnuPG sau OpenPGP, pachete care vor aduce în sistem bibliotecile necesare de criptografie.

Planificare calendaristică

Deşi nu e obligatoriu ca un client de e-mail sa aibă un modul de planificare calendaristică, cei de la Ximian au decis ca Evolution să includă şi un astfel de modul. Astfel, Evolution este singurul program din testul nostru care vă poate ajuta în planificarea timpului activităților.



Resurse

KMail şi Balsa folosesc puține resurse de sistem, spre deosebire de cele două Mozilla şi Evolution, care au nevoie de un calculator destul de puternic pentru a rula în condiții optime.

Dacă aveți nevoie de securitate a e-mailului (prin folosirea POP3, IMAP şi SMTP prin SSL/TLS), vă recomandăm să vă asigurați că aveți un sistem decent, deoarece criptografia necesită o putere mare de calcul.

Concluzie

Recomandarea noastră merge spre Mozilla Mail. Thunderbird, deşi cu facilități asemănătoare, suferă încă de bolile copilăriei. Pe locul doi vine Evolution, un clinet aproape la fel de puternic. Kmail şi Balsa sunt ultimele opțiuni de luat în considerare dacă sunteți în căutarea unui client de e-mail puternic.

Autor:

florin.veres@linux360.ro

Zidul de foc dintre noi şi ceilalți - Firewall

Răzvan Popa

Termenul de firewall provine de la numele structurii cu rol de prevenire a întinderii focului prezentă la automobile (între motor si pasager) şi clădiri.

Firewall-urile în domeniul comunicațiilor pe Internet au rolul de a proteja utilizatorii din spatele lor. Un firewall restricționează accesul la resursele unei rețele cât şi a rețelei la resursele altora.

Pentru a proiecta un firewall, este necesară deciderea asupra câtorva aspecte:

- la ce servicii va avea acces un utilizator din interiorul rețelei şi în ce măsură
- la ce servicii va avea acces un utilizator din exterior în interiorul rețelei şi în ce măsură

Un firewall care filtrează pachete funcționează la nivel de rețea. Informația are voie să circule doar dacă regulile firewall-ului îi permite. Un pachet poate fi filtrat dupa tipul său, adresă sursă/destinație, după portul accesat şi nu numai.

Un firewall, prin modul său de proiectare, foloseşte resurse puține, mare parte din analiza pachetelor facându-se doar asupra header-ului pachetelor. Un dezavantaj în folosirea acestui tip de firewall este lipsa posibilității autentificării cu parolă, singura identitate pe care o are un user fiind IP-ul si MAC-ul plăcii sale de rețea.

Firewall-ul nu necesită schimbări la nivel de client, filtrarea propriu-zisă făcându-se transparent.

Pentru început vom analiza metoda de implementare a unui firewall de tip filtru cu ajutorul utilitarului `/sbin/ipchains`.

Ipchains este predecesorul lui Iptables şi, deşi este considerabil limitat comparativ cu fratele său mai mare, este o soluție încă foarte des întâlnită.

Cum funcționează?

În realitate, nu ipchains este cel care implementează firewall-ul ci kernelul prin intermediul infrastructurii de firewall "ipchains", utilitarul fiind doar o interfață de interacțiune între infrastructură şi utilizator. A fost inclus în sursa kernel în seria 2.2.x şi este compatibil atât cu seria 2.4.x cât şi cu seria 2.6.x.

Majoritatea distribuțiilor includ aceasta infrastructură ca modul. Pentru a activa se va folosi comanda:

```
# modprobe ipchains
```

Din punctul de vedere al infrastructurii ipchains, traficul ce tranzitează nodul în cauză poate fi împărțit(dupa directia de deplasare si destinatia finala) în trei mari categorii(lanțuri) carora le sunt asociate (prin corespondență directă) trei structuri liniare formate din reguli de filtrare şi denumite "lanțuri".Acestea sunt

- `INPUT` - cu acest lanț de reguli sunt confruntate pachetele care intră în nod şi care îl au ca destinație
- `OUTPUT` - cu acest lanț de reguli sunt confruntate pachetele care ies din nod avându-l ca sursă
- `FORWARD` - cu acest lanț de reguli sunt confruntate pachetele care ies sau intră din/în nod şi care nu îl au nici ca sursa şi nici ca destinație (adică pachetele rutate de acest nod - de aici şi denumirea de lanț "de înaintare")

Parametrul prin care se specifica lanțul pe care se va opera este "`-A <nume_lant>`".

Pentru fiecare lanț, traficul se poate categorisi după:

- protocolul folosit: `-p {tcp/udp/icmp}`
- sursa pachetului: `-s <ip>`
- destinația pachetului: `-d <ip>`
- interfața: `-i <interfață>`
- tipul serviciului căruia îi este destinat pachetul.

Ultimul parametru necesar este decizia (acțiunea) care va fi pusă în aplicare la întâlnirea condiției specificate prin parametrii precedenți. Această acțiune se poate specifica cu parametrul "`-j <decizie>`".

Câteva dintre aceste acțiuni pot fi:

- `ACCEPT` - permite trecerea pachetului
- `REJECT` - respinge pachetul în mod activ înştiințând expeditorul în legătură cu decizia
- `DENY` - respinge pachetul în mod pasiv
- `REDIRECT --to <adresa>` - trimite pachetul la `<adresa>`
- `MASQ` - aplică o transformare de tip NAT pachetului, mai precis transformarea MASQUERADE, adică îi rescrie acestuia adresa de plecare cu cea a nodului în cauză, precum şi portul (serviciul de plecare)

Exemple:

```
# ipchains -A INPUT -s 0/0 -p tcp -d 192.168.0.1 smtp -j REJECT
```

Se urmăreşte traficul care vine pentru stație (`-A INPUT`). În acest trafic este urmărit traficul care provine de la orice ip (0/0 este o prescurtare înțeleasă de `/sbin/ipchains` şi interpretată ca 0.0.0.0/0.0.0.0 în notație IP/NetMask sau ca 0.0.0.0/0 în notație CIDR) şi care îi este destinat portului 25 (smtp) a IP-ului 192.168.0.1. Odată detectate aceste

pachete se aplică decizia REJECT.

```
# ipchains -A forward -s 192.168.0.0/24 -d ! 192.168.0.0/24 -i eth0 -j MASQ
```

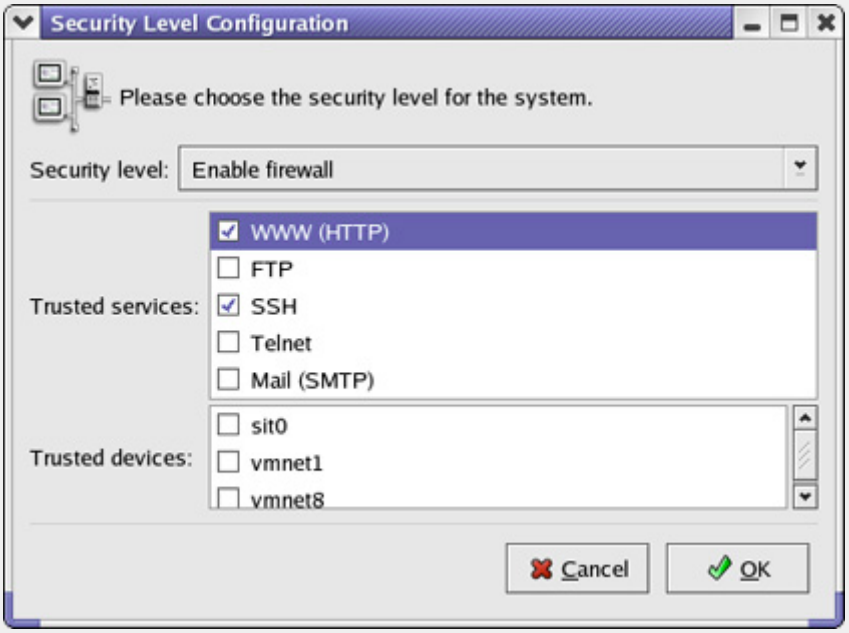
În această comandă se lucrează pe lanțul forward. 192.168.0.0/24 este notația CIDR pentru rețeaua de clasa C 192.168.0.0/255.255.255.0 (formată din adresele de la 192.168.0.1 pâna la 192.168.0.254). Odată întâlnit un astfel de pachet se aplică decizia de masq. Mai multe detalii despre mascaradare se pot găsi pe `http://en.tldp.org/HOWTO/IP-Masquerade-HOWTO/`.

Altă acțiune importantă pe care o poate efectua ipchains este `log`. În general nu veți dori să folosiți `log` pentru traficul obişnuit ci pentru excepții. Aceasta nu e acțiune propriu-zisă, activarea sa făcându-se cu adăugarea parametrului `-l` la o regulă care va fi impusă. O linie de log se va prezenta în modul următor:

```
Packet log: input DENY eth0 PROTO=17 192.168.0.1:53 192.168.0.2:1025
L=34 S=0x00 I=18 F=0x0000 T=254
```

În general aceste loguri (jurnale) sunt destinate specialiştilor, dar pot fi utile şi celor abia inițiați. Explicația unui astfel de log ar fi următoarea:

- *input* este lanțul pe care s-a definit acțiunea.
- *DENY* este decizia luată.
- *eth0* este interfața pe care s-a definit acțiunea
- *PROTO=17* înseamna că protocolul pe care se foloseşte pachetul este 17. O listă cu numărul protocoalelor poate fi gasită în `/etc/protocols`. 17 este protocolul UDP.
- *192.168.0.1;53* este IP-ul sursă al pachetului şi portul de pe care s-a transmis, 53.
- *192.168.0.2;1025* este IP-ul destinație şi portul pe care se transmite, 1025.
- *L=34* este mărimea pachetului, 34 bytes.
- *S=0x00* reprezintă tipul serviciului.



Front-end grafic în Fedora Core pentru configurarea firewall-ului

- *I=18* este ID-ul IP-ului.
- F=0x0000 reprezintă regula de fragmentare impusă
- T=254 reprezintă TTL-ul pachetului.

Aceste informații se pot găsi de obicei in `/var/log/syslog` sau în `/var/log/kern.log`. Locul în care se vor face aceste loguri se poate modifica în fişierul de configurație a syslog(`/etc/syslog.conf`).

Cum construim o regulă

Sintaxa de utilizare a ipchains este:

- `/sbin/ipchains -[ADC] <lant> reguli de identificare a pachetelor [optiuni]`
- `/sbin/ipchains -[RI] <numar regula vizata> reguli de identificare a pachetelor [optiuni]`
- `/sbin/ipchains -D lant <numar regula vizata> [optiuni]`
- `/sbin/ipchains -[LFZNX] [lant] [optiuni]`
- `/sbin/ipchains -P lant actiune [optiuni]`
- `/sbin/ipchains -M [ -L | -S ] [optiuni]`

Comenzi:

- `-A/--append`: adaugă regula la sfârşitul lanțului selectat
- `-D/--delete`: şterge o regulă după numărul sau după un set de reguli de identificare
- `-R/--replace`: înlocuieşte o regulă
- `-I/--insert`: inserează regula la poziția specificată
- `-L/--list`: listează regulile existente
- `-F/--flush`: şterge regulile
- `-Z/--zero`: reinițializează contoarele

Parametri:

- `-P/--policy` : specifică regula implicită care se aplică unui lanț
- `-p/--protocol` : protocolul pachetului
- `-s/--source` : sursa pachetului
- `--source-port` : portul sursă
- `--destination-port` : portul destinație
- `-j/--jump` : specifică regula care va fi aplicată
- `-i/--interface` : specifică interfața pe care se va urmări regula

Pentru un începător, ipchains este cea mai bună alegere, fiind destul de complex pentru a-i oferi ocazia să simtă gustul puterii unui firewall şi totodată fiind destul de simplu de implementat.

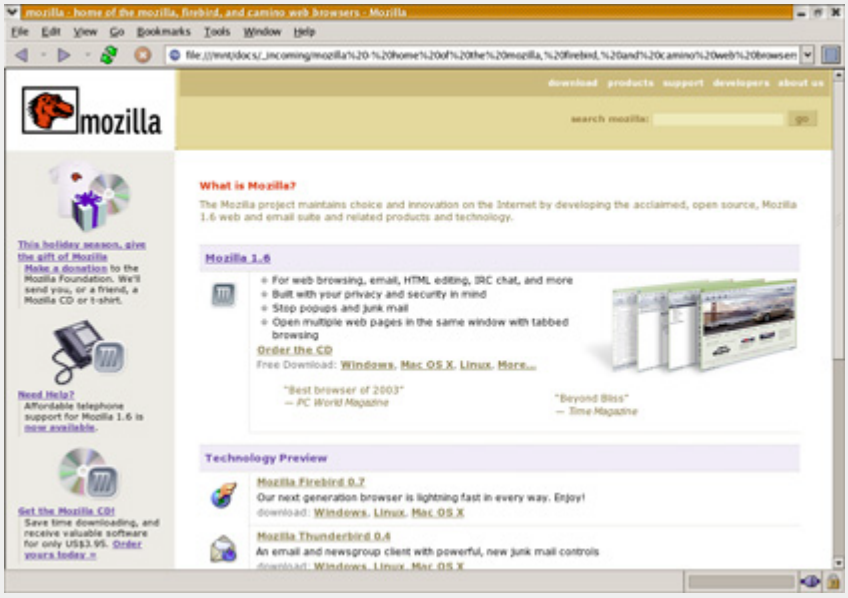
Autor:

razvan.popa@linux360.ro

Pe coama valului - Mozilla 1.6

Ovidiu Lixandru

Mi-aduc aminte de încântarea simțită când am navigat pe net pentru prima dată. Era cândva prin liceu iar accesul se făcea integral prin dial-up. Primele site-uri vizitate, Ferrari şi ProFM, mi-au lăsat o impresie plăcută şi o poftă de a le vizita iarăşi. Şi, pe măsură ce netul a devenit o facilitate tot mai accesată, am devenit mai exigent cu uneltele pe care le foloseam.



Şi Bălcescu grăi

Mozilla îşi are rădăcinile în Mosaic, primul browser grafic dezvoltat pentru World Wide Web. Programatorii săi de la National Center for Supercomputing Applications (Universitatea Illinois) au abandonat proiectul după 4 ani, intuind continuarea sa de către diverşi utilizatori entuziaşti şi facilitându-le acestora sarcina prin punerea la dispoziție a surselor. Netscape Communications au preluat multe din facilitățile Mosaic şi au lansat propriul browser, Navigator. Mai târziu, când acestuia i s-au alăturat un client de mail şi ştiri, ca şi un mic editor HTML, pachetul a fost lansat sub numele de Communicator. Până la generația 4.0, Netscape Navigator a fost liderul incontestabil al pieței browserelor Internet.

Apariția sistemului de operare Windows 98 ce avea integrat (abuziv, după părerea tribunalelor) browser-ul Internet Explorer a înclinat balanța popularității spre produsul Microsoft. Acesta aducea multe noutăți, suportul fiind asigurat pentru HTML 4.0, DHTML şi CSS 1.0, ca şi posibilitatea de a salva integral o pagină (foarte utilă mie). Suportul cu care se putea lăuda IE 4.0 a fost o surpriză pentru toată lumea, predecesorul său din generația 3 fiind foarte sărac din acest punct de vedere. Firma din Redmond a acționat foarte isteț, colaborând îndeaproape pentru implementarea standardelor suportate cu W3C.

Netscape a trecut printr-o perioadă extrem de dificilă, concurenței acerbe a Microsoft alăturându-i-se "gluma" Netscape Communicator 6 (vă mai aduceți aminte în câte minute pornea?) şi achiziționarea de către AOL. Sub conducerea (şi puterea financiară) a acestora, Netscape a avut acea sclipire de geniu ce hotărăşte uneori cursul istoriei - înființarea unei comunități open-source de dezvoltare pentru crearea unui browser, nume de cod Mozilla, pornind de la Netscape Navigator, ei urmând să preia îmbunătățirile în produsul software propriu.

Mozilla a crescut încet şi sigur, ajungând să aibă astăzi cel mai extins suport al standardelor W3C, un motor de randare foarte rapid, ca şi o stabilitate şi securitate de invidiat. El este disponibil pentru un număr impresionant de platforme, printre cele mai populare numărându-se Linux, Windows şi MacOS. Resursele necesare pentru rularea sa pe arhitecturi x86 sunt medii, un procesor tactat la 500MHz şi 128MB RAM făcând față cu brio. Ca o paranteză, aceste cerințe sunt mai mici decât la versiunile sale incipiente derivate din nelansatul Netscape Navigator 5.0.

Instalăm cu responsabilitate

Kit-ul de instalare este disponibil pentru descărcare de pe site-ul Mozilla şi mirror-urile sale. Veți avea de adus în jur de 10MB, în funcție de tipul de pachet ales pentru instalare. Puteți alege dintre binarele precompilate de către Mozilla.org, pachete rpm, deb, şi, pentru cei cu spirit de aventură şi timp de pierdut, surse.

Instalarea pachetelor precompilate (aromă rpm pentru Fedora Core 1-ul meu) a decurs fără sughițuri. S-a instalat browser-ul, clientul de mail, editorul HTML şi clientul de IRC Chatzilla.

La soare te puteai uita

La prima pornire, aspectul browser-ului este destul de încărcat. Spațiul util destinat navigării este diminuat de prezența Sidebar-ului şi a Personal Toolbar-ului,

ambele fiind dezactivate imediat de mine. Sidebar-ul este un asistent ce conține, în mod predefinit, căutarea, bookmark-urile, history-ul şi What's Related, cu posibilitatea de a descărca multe alte plugin-uri. Nimic util pentru mine. În Personal Toolbar, utilizatorul îşi poate salva link-urile cele mai des accesate şi să le aibă la un click distanță. Iarăşi inutil.

Cum Mozilla suportă teme pentru inferfață (şi nu doar pentru browser, ci pentru toate aplicațiile sale), am mers la o mai veche cunoştință de a mea (salutări Gavin) de unde am descărcat şi instalat tema MozCurveBlue. Cu aceasta, am obținut un browser care se integrează perfect cu desktop-ul meu BlueCurve. Am mai dezactivat câteva butoane din toolbar şi am obținut browser-ul ideal: patru butoane pentru navigație şi bara de adrese.

Semne de carte

Bookmark-urile de la versiunea anterioară s-au păstrat fără probleme. Mozilla are inclus o unealtă dedicată managementului bookmark-urilor, pe numele său Bookmark Manager (vă aşteptați la altul?). Cu acesta le puteți aşeza în fel şi chip, puteți căuta după unul în caz că ați uitat unde l-ați pus, le puteți grupa în foldere şi, chestiefoarte utilă, se poate salva un grup de tab-uri sub forma unui singur bookmark.

'jde cai putere

Cum un utilizator nu se poate numi utilizator până nu-şi bagă nasul sub capotă, să vedem ce se află în *Edit - Preferences*.

Primul meniu, *Appearance*, vă permite să setați parametrii de start ai Mozilla, fonturile predefinite folosite pentru afişarea paginilor şi mărimea acestora, culorile, temele şi pachetele de limbă. Cu ajutorul acestora din urmă, puteți schimba limba aplicației, însă eu n-am reuşit să găsesc unul pentru limba română. Am rămas corigent tot la engleză, limba IT-ului.

Mai departe, meniul *Navigator* se ocupă de setările browser-ului. De aici, puteți seta Mozilla ca browser predefinit pentru paginile web, alege butoanele care apar pe toolbar, completare automată a adreselor web, motorul preferat de căutare (Google la loc de cinste), setările pentru tab-uri şi multe altele.

La *Composer* puteți afecta setările editorului HTML din pachetul Mozilla.

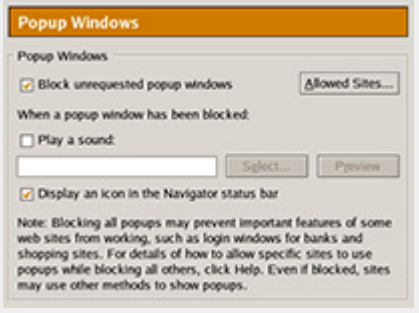
Următorul punct pe lista de atac, *Mail & Newsgroups*, are ca subiect clientul de mail. Acesta conține câteva setări echivalente cu cele ale browser-ului, cum ar fi cele pentru a defini Mozilla Mail ca aplicație predefinită pentru e-mail. Restul se ocupă însă de modalitățile de vizualizare, compunere şi trimitere a mesajelor.

Privacy & Security vă oferă controlul asupra aspectului cel mai delicat al navigării pe Internet, intimitatea.

La primul submeniu, *Cookies*, opțiunile recomandate sunt de acceptare a cookie-urilor doar de la site-ul pe care îl vizitați, dezactivarea folosirii cookir-urilor în clientul de mail şi fără limită de viață a cookie-urilor. Tot de aici puteți accesa *Cookie Manager*-ul, o unealtă ce vă permite vizualizarea şi ştergerea, dacă e cazul, a cookie-urilor existente în mod curent pe sistem, ca şi stabilirea regulilor de acceptare şi respingere.

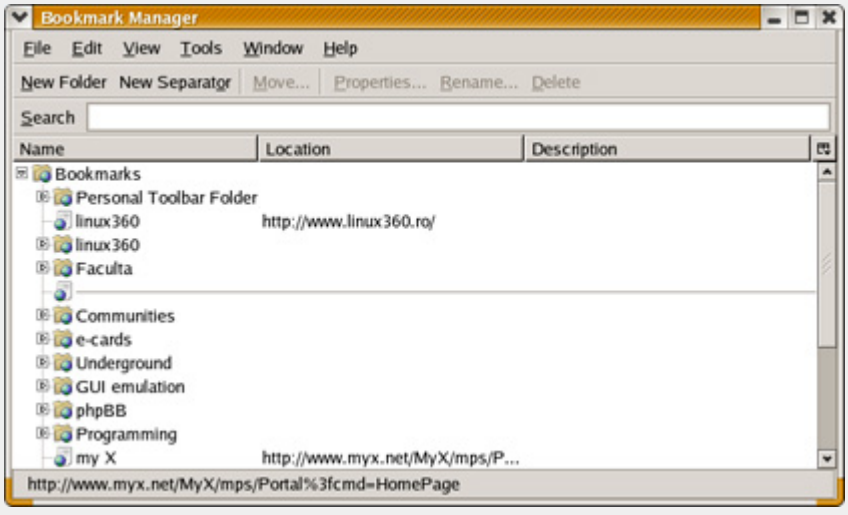
Opriți tracking-ul online

Mozilla vă ajută să scăpați de asemenea şi de ferestrele publicitare pop-up atât de supărătoare chiar pe site-uri din ograda proprie (home.ro, trafic.ro, chip.ro etc.). Când Mozilla blochează o fereastră pop-up, vă va anunța printr-o mică iconiță în bara de stare.



Vis îndeplinit: site-uri fără popup-uri

Trebuie menționat că Mozilla blochează numai ferestre deschise automat prin



Ghiciți care e primul site pe listă?

scripting, şi nu pe cele care se deschid ca urmare a interacțiunii utilizatorului cu elementele de control din pagină (butoane, link-uri etc.). Acest algoritm este atât de eficient, încât nu am avut decât 2-3 rateuri în peste 3 ani de utilizare. Chiar şi în acest caz mi-a fost necesar doar un click pe iconița din bara de stare ce mă anunța că a fost blocată o fereastră, un altul pe *Add to allowed sites* şi reîncărcarea paginii.

Din secțiunea *Forms* se poate seta memorarea de către Mozilla a diverselor câmpuri din forumularele online pentru o completare ulterioară mai rapidă.

Passwords are un rol asemănător, subiectele fiind de această dată câmpurile de login de pe paginile de Internet, memorând nume de utilizator şi parole. Dacă folosiți combinații diferite pentru aceeaşi pagină (două conturi de e-mail pe acelaşi server, de exemplu), nu vă îngrijorați, Mozilla le va memora pe toate şi vă va întreba la încărcarea paginii cu care din ele doriți să completeze formularul.

Master Passwords vă protejează printr-o singură parolă toate datele memorate de browser: câmpuri de formulare, nume de utilizator, parole, certificate digitale personale şi chei private. Aceasta este o setare foarte bună în cazul în care mai foloseşte cineva acelaşi cont de sistem şi nu doriți să aibă acces la datele personale stocate de către Mozilla.

Următoarele submeniuri, *SSL*, *Certificates* şi *Validation*, se înrudesc şi se referă toate la securitatea comunicației dintre browser şi server.

Ultimul meniu de preferințe se numeşte *Advanced* şi, conține, după cum îi spune şi numele, câteva setări de subtilitate. Veți putea activa sau dezactiva folosirea Java şi, implicit, a încărcării aplicațiilor ce folosesc acest limbaj (instalarea Sun Java Runtime Envirnment se face separat, acesta nefiind inclus în pachet). Vă recomand, pentru o securitate şi navigare mai facilă, să umblați şi la setările de *Scripts & Plugins*, lăsând activate doar *Hide the status bar* şi *Change images*, şi să-l dezactivați complet pentru clientul de mail.

Javascript supravegheat

Keyboard Navigation este un alt punct la care trebuie să umblați. Mozilla conține o facilitate foarte utilă, Find As You Type. Cu ajutorul acesteia puteți găsi extrem de rapid cuvinte sau pasaje de text în pagină scriindu-le pur şi simplu de la tastatură, fără a mai utiliza dialogul de căutare. Dar pentru aceasta trebuie să mutați opțiunea predefinită de căutare doar în link-uri pe cea de căutare în toată pagina (Any text in the page).

Mai jos vă veți putea exprima preferințele pentru mărimea cache-ului browser-ului, proxy-uri, conexiunile HTTP, rotița mouse-ului etc.

Printre munții de site-uri

Cum deja am scris despre facilități de resimt acidul lactic în degete, e timpul să-l vedem la treabă pe Internet.

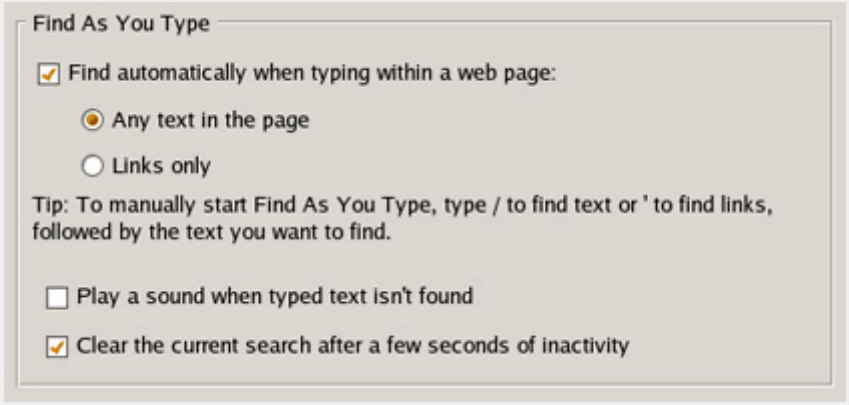
Utilizatorul poate opta pentru afişarea clasică, cu fiecare pagină având fereastra ei, sau pentru varianta mai modernă, în care avem o singură fereastră a browser-ului, iar paginile sunt încărcate în tab-uri ale acesteia. Eu vă recomand să mergeți pe mâna tab-urilor, manevrarea acestora fiind vizibil mai uşoară decât a ferestrelor individuale.

Suportul complet pentru standardele W3C actuale îi asigură browser-ului Mozilla o comptabilitate de invidiat cu 99% din site-urile de pe net. Care sunt restul de 1%? Sunt site-uri făcute de webdeveloperii de 2 lei şi 5 bani ce afişează cu mândrie pe pagină textul "Microsoft Internet Explorer 14.1 SP2 or better required". Când vedeți aşa ceva, fugiți fără să vă uitați înapoi.

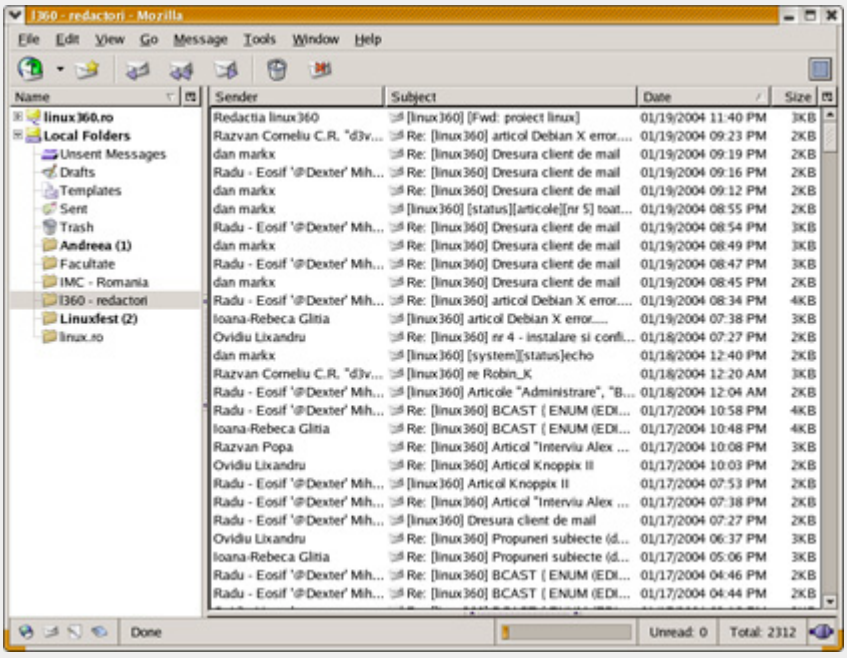
Cu alte cuvinte, avem suport pentru HTML, DHTML, DOM, CSS, XML, XSL, XLink, XPath, RDF, MathML, XUL, XBL, XPCOM, SOAP şi Javascript (sau ECMA script, cum doriți). V-a lăsat cu gura căscată, nu?

Motorul de randare Gecko, care echipează şi alte browsere cum sunt Firefox şi Epiphany, este foarte performant, reuşind efectiv să randeze pagina pe măsură ce aceasta se încarcă. Mai are scăpări, dar, spre deosebire de mshtml sau khtml, este cel mai aproape de ceea ce spun programatorii.

Nu am rezistat să nu-l supun unor teste mai deocheate. L-a trecut pe cel ale încărcării a 20 de pagini diferite atât în tab-uri, cât şi în ferestre individuale, în ambele cazuri performanța aplicației neavând prea mult de suferit. La cel al încărcării unui fişier HTML ce conținea 1MB



Find As You Type pentru căutări rapide



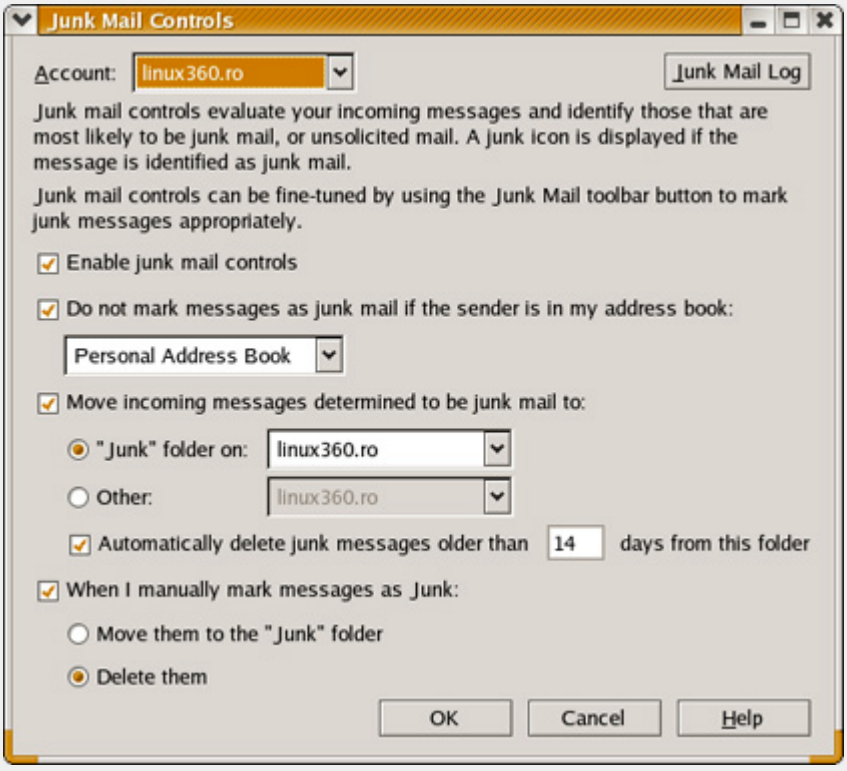
Fără plicuri, fără tavă de argint, dar mult mai rapid

de text, deşi s-a comportat mai bine decât versiunile anterioare, nemaiblocându-se, scroll-ul prin document a fost foarte încet.

Poştaşul electronic

A doua aplicație inclusă în pachet este Mozilla Mail, un client de poştă electronică şi ştiri.

Acestuia i-am descoprit puterea mult mai târziu decât a browser-ului, având în minte minusurile pe care le avea cel din generația Netscape 4.



Adio şi n-am cuvinte, spam!

Vremurile s-au schimbat însă, iar Mozilla Mail are suport pentru POP3, POP3S, IMAP, IMAPS şi NNTP. Se pot seta conturi multiple, putându-se seta pentru fiecare în parte toți parametrii la care vă puteți gândi.

Un ou, doi oi

Cele mai puternice funcții ale clientului de mail sunt filtrele. Acestea sunt de două tipuri: filtrele de sortare, pe care le defineşte utilizatorul (puteți instrui programul să pună toate mesajele de la o persoană anume într-un anumit folder) şi filtrele de junk mail. Acestea nu necesită decât activarea pentru contul respectiv, algoritmii de detectare a mesajelor nedorite acționând automat după anumte cuvinte cheie sau după adresele de la care se primesc mesajele.

Facilitatea este binevenită, adresa de e-mail a redacției fiind foarte expusă acestui tip nedorit de mesaje, Mozilla Mail ştergându-le însă fără excepție.

Posibilitățile de conlucrare cu alte aplicații sunt mediocre. Puteți importa agenda de contacte, mesajele şi setările din Netscape Communicator, dar cam atât. Ştie de câteva formate generice (tab, csv), dar nu vă bazați pe ele. Aceste opțiuni spartane nu sunt foarte bune, dar nici rele. Cu acest client, e bine să porniți de la zero şi să-l configurați din temelii. O dată acest lucru îndeplinit, vă asigur că nu vă va mai provoca nici o problema de compatibilitate.

Afişarea mesajelor este deosebit de flexibilă, aplicația dispunând de numeroase crieterii predefinite, putând să vă setați şi moduri proprii (de exemplu să fie afişate numai mesajele primite în ultimele 24 de ore, în ordine descendentă). Am întâlnit la această aplicație şi un criteriu nou, acela al sortării mesajelor după ordinea primirii. Astfel, chiar dacă ora sistemului de pe care s-a trimis mesajul este eronată, el va afişat în ordinea corectă pe calculatorul destinație.

O altă opțiune interesantă e de a folosi coduri de culori pentru a identifica vizual importanța mesajelor. Regulile le stabiliți

tot în dialogul de configurare al filtrelor.

Securitatea este absolută, putând dezactiva pentru această aplicație atât folosirea plugin-urilor (Flash, Quicktime etc.), a cookie-urilor cât şi a Javascript-ului. Astfel, indiferent că lucrați cu mesaje HTML sau plain-text, veți sta liniştit că nu se întâmplă nimic rău (fără să fi apăsat dumneavostră tasta Del în mod imprudent, adică).

Mozilla Mail este exact ceea ce îi spune şi numele - un client de poştă, şi îşi îndeplineşte sarcina exemplar. Pot afirma că este de departe cel mai puternic client pe care l-am folosit până acum, incluzând Evolution şi KMail.

Agenda

Mozilla Address Book este o mică aplicație de management al contactelor. Este foarte bine integrată clinetul de mail, îndeplinind funcție de autocompletare atunci când scrieți destinatarul unui e-mail, de exemplu.

Detaliile pe care le puteți introduce despre o persoană sunt destul de multe, singurele pe care eu aş fi avut nevoie să le folosesc şi pe care nu le-am regăsit printre câmpuri fiind ziua de naştere şi a ID-urilor pentru protocoale de mesagerie insant.

Şi fiindcă tot veni vorba de mesagerie instant, agenda are o asemenea funcție. Eu, unul, nu am reuşit s-o pun în practică. Într-adevăr, când selectam un contact şi apăsam butonul din toolbar asociat acestei funcții, pornea GAIM, dar nu se întâmpla nimic. Am încercat să completez câmpul Nickname al respectivului contact cu unul din ID-urile sale de mesagerie, dar fără nici un rezultat.

Internet Relay Chat

ChatZilla este un client IRC simplu, ideal dacă nu aveți nevoie de scripting şi alte minunății specifice acestui tip de clienți. Eu folosesc oricum instant messenger-ul pentru acest tip de comunicație.

Şi noi, dezvoltatorii?

Şi noi. Dacă vă veți plimba prin meniul Tools al browser-ului, veți descoperi câteva bunătăți: Javascript Console, Java Console, DOM Inspector şi Javascript Debugger. Acestea furnizează un mediu complet de testare şi debugging al aplicațiilor client-side. Foarte puțini realizează puterea ascunsă în spatele lor, eu neavând încă ocazia să văd pe cineva care le foloseşte (excluzând propria persoană).

Tot țintit spre dezvoltatori este şi micul Composer, un editor HTML WYSIWYG. Dar numai atât - HTML. Suportul pentru editarea acestuia este foarte bun iar interfața este foarte intuitivă. Este practic un editor de text în genul Writer, numai că acesta produce fişiere destinate Web-ului.

Misc.

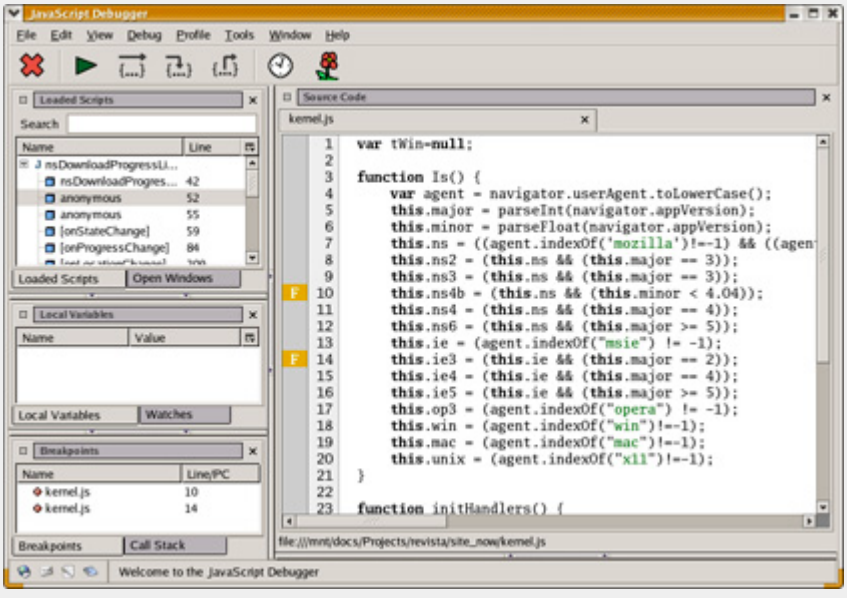
Pachetul mai conține un download manager care nu străluceşte însă, putând relua download-urile întrerupte numai în cadrul sesiunii curente, şi aceasta mai dând greş din când în când.

Mozilla foloseşte profile, putând crea identități cu propriile configurații şi date stocate. Un fel de conturi de utilizatori restrânse la nivelul aplicației.

Utilizatorii cu adevărat curioşi vor găsi în fişierul $HOME/.mozilla/prefs.js o nouă jucărie. Cu ajutorul acestui fişier se pot modifica absolut toți parametrii aplicației, mulți neapărând în dialogul de editare a preferințelor. Foarte folositoare mi s-a părut posibilitatea de a schimba complet string-ul user-agent (prin setarea parametrului general.useragent.override). Astfel, puteți simula pentru aplicații sau paginile web vizitate orice browser şi orice platformă doriți.

...este egal cu

Recomand pachetul Mozilla oricui navighează pe Internet şi are resursele hardware necesare. Este un model de funcționalitate, stabilitate şi securitate. Aplicațiile principale, browser-ul şi clientul de mail, sunt indiscutabil cele mai bune de pe piață şi, din felul în care se dezvoltă, vor rămâne pentru mult timp în fruntea tuturor.



RAID-ul gândacilor digitali

Resurse:

- www.mozilla.org
- www.mozdev.org

Autor:

ovidiu.lixandru@linux360.ro

Date rătăcite? Recuperați-le cu Knoppix LiveCD

Ion Mudreac

Ni se mai întâmplă să mai prindem câte o pană de curent cu PC-ul pornit, după care să nu mai putem boot-a sistemul de operare şi accesa datele care de multe ori pot avea valoare mare şi importanță deosebită pentru utilizatori .

Aceste situații pe care le putem numi şi situații de criză sunt extrem de neplăcute, putând avea consecințe negative atât asupra sistemului cât şi productivității utilizatorului ce depinde de aceste date.

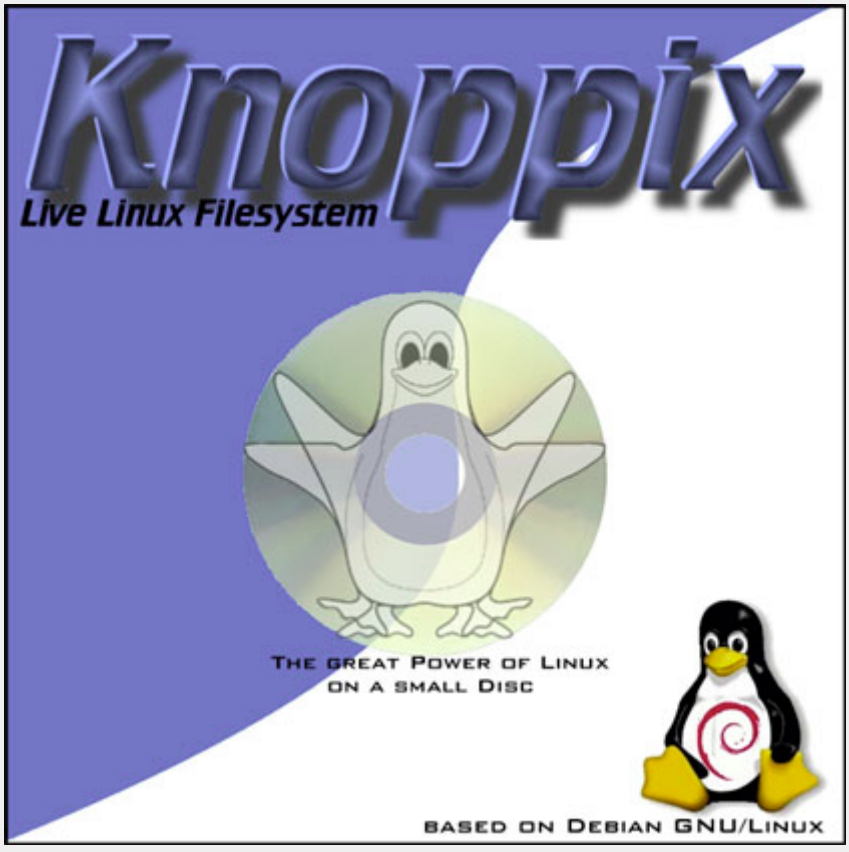
Situațiile de criză în deteriorarea datelor pot apărea din mai multe motive. Atât factorul uman poate fi implicat direct în crearea acestor situații neplăcute cât şi factorul tehnic poate contribui în aceeaşi măsură.

Există soluții de recuperare a datelor şi una dintre aceste soluții o vom prezenta în acest articol.

Vom vorbi în rândurile de mai jos despre cum se poate accesa un sistem Linux ce nu ne mai permite să boot-ăm de pe el.

Am prezentat în numerele trecute Knoppix LiveCD, de care ne vom folosi în încercarea de a recupera datele importante.

În trecut soluția universală de boot şi recuperare a datelor era floppy disk-ul. O dată cu răspândirea tot mai largă a CD-ROM-urilor ce au devenit un standard de facto în industria IT, au apărut tot mai multe CD-uri boot-abile (aşa-numitele SuperRescue CD) bazate în special pe Red Hat 7.2 dar care aveau un mare dezavantaj în imposibilitatea detectării hardware-ului mai exotic cum ar fi echipamentele USB sau wireless.



La toate aceste probleme s-a descoperit o soluție denumită Knoppix, ce oferă facilități nemaiîntâlnite până acum:

- un sistem independent bazat pe Debian şi KDE
- bootare completă în jur de două minute
- gratuit sau se poate achiziționa la prețul unui CD blank

Toate facilitățile pe cere ni le aduce Knoppix le-am discutat în numerele anterioare ale revistei.

Resuscitare

Dacă ni s-a întâmplat să nu mai putem boota sistemul de operare Linux, există o ieşire elegantă. Boot-ăm LiveCD-ul Knoppix, în timpul procesului de boot Knoppix va identifica toate discurile şi partițiile din sistem pe care le va prezenta pe Desktop în KDE. De aici putem destul de simplu accesa fişierele din PC. Odată accesate aceste partiții pe care Knoppix le montează în mod read-only, fişierele deschise nu pot fi modificate numai vizualizate. Nu este nici o problemă, această situație se poate remedia foarte simplu. Clic dreapta pe pictograma partiție ce se doreşte a fi accesată, va apărea un meniu cu opțiunea "`Change read/write mode`" iar aceasta ne va permite accesarea sistemului de fişiere cu posibilitatea atât de citire, cât şi scriere.

Odată boot-at, LiveCD-ul vă loghează cu

utilizatorul ce are aceeaşi denumire ca şi distribuția, şi care nu are drepturile depline asupra sistemului. Pentru a facilita accesul cu drepturi depline asupra sistemului Knoppix, aveți nevoie să vă logați cu utilizatorul *root* pentru care trebuie creată o parolă.

```
knoppix@ttyp0[knoppix]# su
root@ttyp0[knoppix]# passwd
```

Pentru montarea partițiilor de pe HDD în regim read/write din consolă:

```
root@ttyp0[knoppix]# mount -t reiserfs -o rw /dev/hda5 /mnt/hda5
```

Pentru demontare:

```
root@ttyp0[knoppix]# umount /mnt/hda5
```

În cazul în care vi se afişează un mesaj de eroare de genul "`Could not unmount device, device is busy`", înseamnă că sistemul de fişiere este accesat de către un alt program şi în acest caz se recomandă să închideți programele ce rulează în acel moment. Apoi este deajuns să vizualizăm fişierul `/etc/fstab` pentru o observa modificările:

```
root@ttyp0[knoppix]# cat /etc/fstab

...

# Added by Knoppix
/dev/hda5 /mnt/hda5 reiserfs noauto,users,exec 0 0
```

Detectarea hardware

Knoppix este o distribuție LiveCD ce excelează în detecția hardware. La toate acestea se mai adaugă un set complet de utilitare ce ne pot furniza informații suplimentare cu privire la componentele din sistem: `fdisk`, `lspci`, `iwconfig`, `ifconfig`, `dmesg`, `/proc` şi multe altele. Aceste utilitare ne pot permite testarea hardware pentru verificarea compatibilității cu Linux a diverselor componente cum ar fi plăcile de sunet, modemurile software, plăcile de rețea wireless etc.

- `fdisk -l` va afişa toate partițiile de pe toate harddisk-urile.
- `lspci -v` va furniza informații despre toate componentele ce sunt conectate la magistrala PCI
- `cat /proc/cpuinfo` vă oferă informații detaliate despre procesorul instalat.
- `ifconfig -a` ne va afişa informații atât despre plăcile de rețea detectate cât şi dispozitivele ppp - interfețele pentru modem
- `iwconfig` ne va afişa informațiile ca şi în cazul lui ifconfig, numai că acestea vor conține informații pentru NIC-urile wireless.
- `dmesg` este o comandă foarte utilă în special pentru persoanele ce se ocupă de kernel hacking. Pentru mai multe detalii operați `#man dmesg`. Sintaxa acestei comenzi este `#dmesg | grep <nume dispozitiv>`

Şi în final în KDE avem `System > Info Center`. Acesta ne permite o vizualizare în formă grafică a componentelor hardware instalate şi detectate de către Knoppix LiveCD.

Recuperarea fişierelor

Prima opțiune în salvarea datelor este copierea fişierelor importante pe un alt mediu: floppy, HDD, CD-R etc. Cea mai simplă şi rapidă opțiune este copierea datelor pe un alt HDD. Aceasta se poate efectua după bootarea LiveCD-ului Knoppix, montarea partițiilor sursă/destinație de pe ambele HDD-uri şi copierea efectivă a fişierelor ce ne interesează. Înainte de a copia datele trebuie pregătit HDD-ul pe care îl vom folosi pentru back-up.

Partiționarea şi reformatarea

Înainte de toate trebuie să avem un HDD disponibil pe care să îl instalăm în sistem. După aceasta, bootăm LiveCD-ul Knoppix şi deschidem un root shell.

Dacă pe discul pe care va fi recuperată informația deja există o partiție, putem trece direct la formatarea acesteia cu sistemul de fişiere ales din multitudinea ce ne este pusă la dispoziție de suportul oferit în Linux. Atenție la denumirile HDD-urilor folosite de sistemul Linux. Pentru discurile SCSI vom avea `/dev/sd`, iar pentru cele IDE `/dev/hd`.

Comanda pentru vizualizarea partițiilor existente pe un HDD IDE este:

```
root@ttyp0[knoppix]# fdisk -l /dev/hdb
```

Formatăm prima partiția existentă de pe HDD-ul hdb:

```
root@ttyp0[knoppix]#
mkfs.ext2 -c /dev/hdb1
```

În acest caz am utilizat sistemul de fişiere ext2 la care am adăugat opțiunea `-c` care va verifica HDD-ul şi pentru eventuale sectoare defecte.

Desigur puteți opta şi pentru alt sistem de fişiere cum sunt ext3, ReiserFS, XFS, JFS etc. care sunt suportate de kernelul Linux.

```
root@ttyp0[knoppix]#   mke2fs
-j -c /dev/hdb1
root@ttyp0[knoppix]#
mkreiserfs /dev/hdb1
```

În cazul în care hdd-ul pregătit pentru back-up este nou sau nu are nici o partiție în prealabil creată, mai întâi vom crea partiția după care o vom formata.

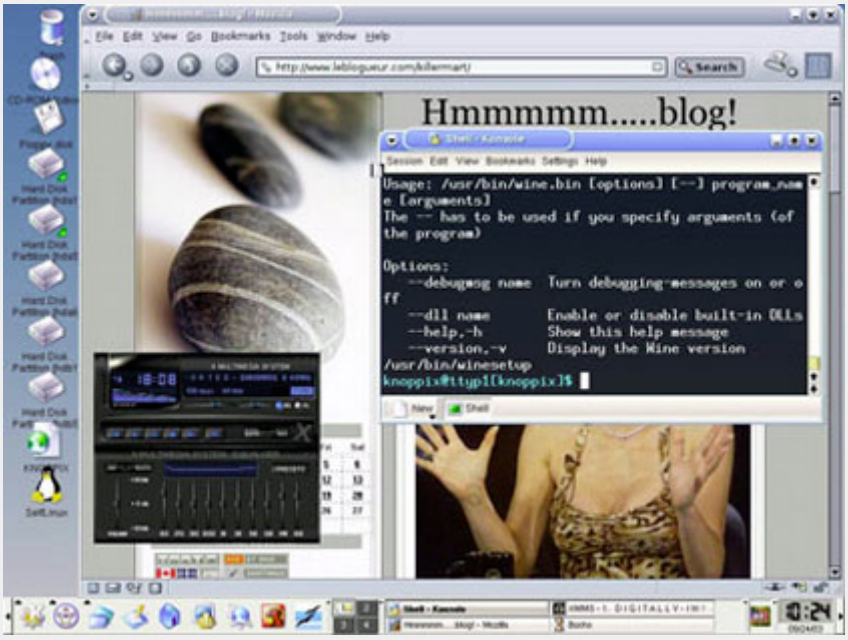
Pentru crearea partițiilor putem utiliza **fdisk**, un program foarte bine cunoscut în lumea Linux datorită robusteții sale dovedite de-a lungul timpului. Să nu vă fie frica să utilizați această unealtă pentru că vă permite un grad ridicat de flexibilitate în crearea şi manipularea partițiilor. fdisk nu va scrie nimic pe disc până nu veți introduce comanda pentru executare.

Vom prezenta secvența pentru crearea unei partiții

```
root@ttyp0[knoppix]#
fdisk /dev/hdb
```

În timpul rulării utilitarului fdisk în orice moment puteți apăsa tasta "`m`" unde vă va fi prezentată o listă de comenzi ce sunt acceptate de către fdisk. Pentru crearea unei noi partiții utilizăm tasta "`n`", iar pentru crearea unei partiții primare utilizăm tasta "`p`". Enter de două ori pentru acceptarea modificărilor efectuate. Dacă nu doriți utilizarea întregului disc, apăsați tasta Enter o singură dată şi introduceți apoi dimensiunea dorită pentru noua partiție:

`+1000M` (în cazul nostru, 1GiB)



În orice moment puteți vizualiza partițiile ce au fost create utilizând tasta "`p`". După ce suntem siguri de modificările exercitate putem să le salvăm pe disc utilizând tasta "`w`".

Utilitarul fdisk va crea implicit partiția de tipul 83, ceea ce înseamnă partiție de tip Linux. Pentru vizualizarea întregii liste a tipurilor de partiții apăsăm tasta "`l`". Pentru modificarea tipului de partiție tastați "`t`". Pentru ştergerea partiției tastați "`d`".

Pentru utilizatorii ce nu au experiență în ale consolei le recomandăm unealta grafică pentru partiționarea discurilor QTParted.

Acest utilitar este foarte simplu şi intuitiv în utilizare. Pentru a-l executa, accesați meniul `System > QTParted`. QTParted ştie să creeze, să şteargă, să mute şi să redimensioneze partiții în mod nedistructiv. El se descurcă foarte bine şi cu sistemul de fişiere NTFS.

Copierea fişierelor în mod GUI

Utilizând interfața grafică este mult mai simplu în copierea fişierelor de pe un disc pe altul decât utilizând consola. Putem utiliza opțiunea drag and drop pentru copierea fişierelor între ferestre deschise, fiecare fereastră reprezentând câte un hard disk, sursa şi destinația. Numai să aveți grijă la HDD-ul pe care se vor scrie fişierele să fie montat în mod citire/scriere, nu doar citire.

Copierea fişierelor în mod consolă

Primul pas este crerea directorului în care vom copia fişierele ce ne interesează, iar apoi copierea lor efectivă:

```
# mkdir /mnt/hdb1
/home/utilizator/backup
# cp -r /mnt/hda5/home
/utilizator/mnt/hdb1
/home/utilizator/backup
```

Clonarea întregului disc

În cazul în care vrem să transferăm sistemul pe un alt disc este destul de simplu dar aveți nevoie fie ca disc-urile să fie de cel puțin aceeaşi capacitate, fie ca discul destinație să fie mai mare. Înainte de a începe orice proces de clonare a discului asigurați-vă că nu este montată nici o partiție de pe ambele HDD-uri. În exemplul nostru avem `/dev/hda` ce va fi sursa şi `/dev/hdb` destinația. Comanda `dd` copiază byte-for-byte inclusiv MBR (master boot record):

```
# dd if=/dev/hda of=/dev/hdb
```

Momente de confuzie

După mai multe montări şi demontări succesive, puteți să ajungeți în situația de a nu vă mai orienta şi să nu mai ştiți ce partiție ați montat şi unde. Dar să nu vă faceți griji, avem o rezolvare la aceasta problemă utilizând `/proc`:

```
# cat /proc/mount
```

Comanda ne va afişa toate sistemele de fişiere montate, tipul acestora şi opțiunile cu care au fost montate, read/write sau numai read.

Pentru vizualizarea detaliilor referitoare la hard disk-urile din sistem şi tipul acestora (SCSI sau IDE) vom utiliza:

```
# fdisk -l
```

sau, pentru discurile IDE,

```
# dmesg | grep hd
```

respectiv, pentru discurile SCSI,

```
# dmesg | grep sd
```

Copierea pe CD-R

Odată ce ați bootat în Knoppix, mediul de lucru implicit este KDE. Acesta conține utilitarul **K3b** pentru scrierea de CD-uri care ne poate fi de un mare folos în recuperarea datelor dacă nu avem un alt HDD sau nu suntem în rețea. BIneînțeles, trebuie să avem o unitate CD-RW şi un blank CD. Şi în acest caz putem utiliza opțiune drag and drop în fereastra deschisă de K3b.

Recuperarea datelor pe orice alt suport de înmagazinare date

Unitățile Zip, floppy disk-urile şi echipamente de stocare USB sau Firewire, toate vor fi recunoscute automat de către Knoppix şi le veți putea accesa direct de pe desktop-ul KDE. De aici puteți copia fişierele importante dar să aveți grijă să nu depăşească capacitatea de stocare a mediului utilizat.

Recuperarea datelor utilizând rețeaua

Distribuția Knoppix permite recuperarea datelor şi în rețea. Primul pas este configurarea accesului în rețea. În mod GUI găsim în `Knoppix > Network/Internet` un asistent cu ajutorul căruia vom introduce datele de identificare din rețea cum ar fi adresa IP, broadcast, netmask şi, pentru acces Internet, vom mai avea nevoie să introducem datele pentru gateway şi DNS.

Configurarea rețelei se poate efectua şi în consola tastând ca root

```
# netcardconfig
```

ce va executa un asistent asemănător cu cel din GUI.

Odată ce rețeaua a fost configurată, avem la dispoziție câteva opțiuni pentru recuperarea datelor prin copiere utilizând comanda `cp` dacă sistemul de fişiere a fost montat local. La fel, putem utiliza comanda `scp` (secure copy), pentru o mai mare securitate în transferul datelor. Comanda `scp` utilizează serviciul ssh pentru criptarea datelor în rețea şi ne va permite transferul fără configurarea serviciilor NFS sau Samba. În acest caz aveți nevoie ca PC-ul pe care-l vom utiliza ca back-up să ruleze serviciul **ssh**.

Dacă acesta nu e pornit pe calculatorul destinație, va trebui să o facem noi:

```
# /etc/init.d/sshd start
```

Comanda pentru copierea fişierelor alese pentru back-up este:

```
# scp -rp /mnt/hda5
/home/utilizator 192.168.1.5
:/home/utilizator/tmp
```

Utilizarea chroot

În multe cazuri există posibilitatea ca un utilizator să fi modificat ceva în sistem şi sistemul Linux sa nu mai poată boota tocmai din cauza acestei intervenții a utilizatorului la fişierele de configurare. Pentru a îndrepta lucrurile există posibilitatea de acces a sistemului ca si cum ați fi bootat din el utilizând comanda chroot.

Pentru utilizarea comenzii chroot trebuie să fiți logat ca root:

```
root@ttyp0[knoppix]#
mount /dev/hda1 /mnt/hda1
root@ttyp0[knoppix]#
chroot /mnt/hda1
root@Knoppix:/
```

Knoppix LiveCD poate fi în multe cazuri unealta ideală în recuperarea datelor de pe sistemele Linux ce nu mai pot fi bootate şi, de ce nu, de pe sisteme Windows. Odată cu apariția versiunii Knoppix 3.4 care va include un kernel Linux 2.6 ce permite scrierea partițiilor NTFS, va fi mult mai simplu de recuperat şi/sau modificat fişierele de pe partiții non-Linux cum este NTFS-ul.

Este bine sa consultați manualul care este livrat cu fiecare comandă în parte.

- `man fdisk`
- `man mkfs`
- `man fstab`
- `man ifconfig`
- `man mount`
- `man proc`
- `man dmesg`
- `man iwconfig`
- `man chroot`
- `man scp`
- `man sshd`
- `man dd`
- `man lspci`

Autor:

ion.mudreac@linux360.ro

Instalați corect Unreal Tournament 2003

Ciprian Negrilă

UT2003 pe Linux?

Da, se pare că succesul înregistrat de id Software (cu al lor unic şi irepetabil Quake) în comunitatea open-source a deschis ochii celor de la Epic Games. Astfel se face că pe cel de-al treilea CD se găseşte ascuns departe de privirile indiscrete un fişier numit simplu: `linux_installer.sh`.

Ce mai aşteptăm?!

Nu vă grăbiți să-l rulați direct de pe CD pentru că în timpul instalării va trebui să schimbați CD-ul şi atunci nu veți putea să rulați `umount` pe CD-ROM, scriptul rulând de pe CD. Cu alte cuvinte Epic ne minte când ne anunță că scriptul se dezarhivează într-o locație temporară. Ceea ce trebuie să faceți este să copiați scriptul într-un director temporar (cum ar fi "/tmp") şi să-l rulați de acolo. Un mic avertisment pentru utilizatorii de plăci grafice NVIDIA Riva TNT2 sau procesoare AMD K6-2: jocul nu rulează din păcate pe configurațiile acestea. Însă dacă aveți un sistem mai "tare" vă recomand cu căldură să îl încercați. Înainte de a continua cu instalarea asigurați-vă că sunteți în modul grafic şi că aveți OpenGL instalat şi configurat cum trebuie. Dacă săriți peste pasul acesta, o să înregistrați un frame-rate monstruos (în jur de 0.3 fps). Dacă este totul în regulă puteți să treceți la instalarea propriu-zisă.

Şi calvarul începe

Rulați scriptul şi aşteptați ca mica aplicație să se compileze. În scurt timp pe ecran va apărea installer-ul urmat de licența jocului. Faceți click pe butonul "I Agree" după ce ați citit în prealabil cu ce sunteți de acord. Apoi trebuie să selectați directorul în care să instalați jocul. Vă recomand destinația implicită a installer-

ului, dar puteți folosi ca alternativă `/usr/games/ut2003`. Dacă preferați să-l instalați în altă parte, riscați să împrăştiați programele peste tot, ceea ce nu vă recomand. Apoi selectați calea link-ului către executabil. Vă recomand şi aici opțiunea implicită sau orice alt director "bin" al sistemului (`/usr/bin`, `/bin` etc.). Când sunteți multumit de selecțiile făcute, dați click pe butonul "Begin Install". Dacă nu este deja montat, installer-ul o să vă ceară "Install Disk". Aveți grijă ca CD-ul să fie montat într-un director uşor de ghicit (`/mnt/cdrom`). S-ar putea să vă confruntați cu o frecventă citire a unității floppy. Dacă se întâmplă încercați să nu ignorați acest lucru deoarece prelungeşte instalarea, care este şi aşa extrem de anevoioasă. Puteți rezolva acest mic bug prin introducerea unei dischete în unitate şi rularea comenzii de montare (`mount /dev/fd0`). Astfel discheta va fi citită o singură dată după care installer-ul va renunța să o mai controleze. Acum tot ce trebuie este să aşteptați până termină cu primul CD. Între timp vă puteți arunca o privire prin readme ca să nu vă plictisiți. Când a terminat primul CD o să aveți o surpriză. Installer-ul cere "Disk number one", care aparent este în unitate. Aceasta este pur şi simplu o greseală (oare!?) deşi vom vedea că cei care au făcut installer-ul s-au "întrecut pe ei înşişi". Ignorați această cerere a primului CD şi introduceți-l pe cel de-al doilea. Nu uitați să-l montati tot în `/mnt/cdrom`. Selectați "Yes" şi totul va continua normal.

Dacă installer-ul încearcă iar să citească de pe dischetă nu vă speriați, acest lucru se va întampla doar o dată, iar la ultimul CD încă o dată. După aceasta, instalarea va continua fără probleme până la CD-ul al treilea. După ce a terminat de copiat de pe cel de-al doilea CD installer-ul va cere cu neruşinare "Disk number two". Se pare că cei de la Epic nu ştiu să numere sau pur şi simplu vor să ne arate interesul lor... în orice caz noi trebuie sa montăm CD-ul trei în acelaşi "/mnt/cdrom" şi să selectăm "Yes". După ce s-a terminat totul va apărea un terminal care va cere serial number-ul. Mare grijă la acest pas, deoarece dacă introduceți codul greşit toată instalarea a fost făcută degeaba. Aşa că nu încercați cu Shift+Insert sau Ctrl+V că veți avea rezultate nedorite (dar cum puteți da paste, dacă codul se află pe carcasa CD-ului? :-D). Încercați să nu ratați acest pas deoarece nu există altă cale de a introduce codul, trebuie să reinstalați. După ce ați scăpat şi de acest hop încheiați instalarea. Dacă totul a decurs cum trebuie comanda cu care lansați jocul este "ut2003". Pe ecran va apărea splash-screen-ul jocului, după care puteți începe să jucați. Şi cu asta s-a încheiat cu succes şi "fără evenimente" instalarea unui joc superb.

Concluzii

Deşi are unele probleme, multe dintre ele haioase, acest installer este un mare pas înainte. Saăsperăm ca acțiunea celor de la Epic Games va deschide orizonturi noi producției de jocuri pe platforma Linux. Până atunci vă las să vă nimiciți rivalii în acest Unreal Tournament de excepție.

Autor:

ciprian.negrila@linux360.ro

Refresh, rezoluție, modeline - configurarea ansamblului video

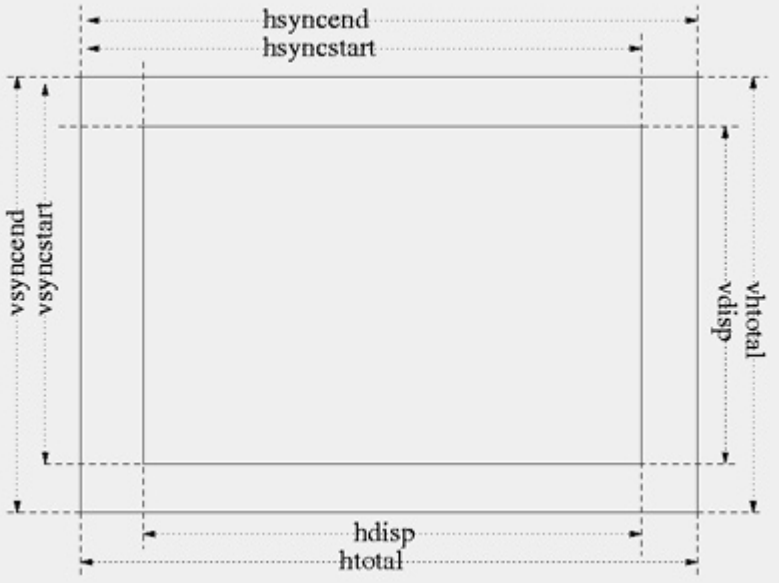
Ciprian Negrilă

A fost odată un monitor

Cu toții ştim că monitorul este cel mai bun prieten al nostru în ceea ce priveşte interpretarea şi analizarea datelor furnizate de "cutia noastră neagră". Şi pentru a putea optimiza această comunicare trebuie să ştim unele lucruri elementare. Astfel, voi începe cu câteva sfaturi elementare. Primul este ca, înainte să vă apucați de restructurări majore ale oricărui fişier de configurare, să citiți manualul monitorului. Este un pas de bază şi chiar cei mai experimentați utilizatori se pot întâlni cu valori sau variabile necunoscute. Ceea ce veți citi în acest articol se poate dovedi FOARTE PERICULOS pentru integritatea monitorului sau a sănătății voastre. Nu, nu este o glumă. Un alt element pe care vi-l recomand cu căldură este să nu umblați la setările implicite DECÂT în cazul în care ceva nu merge bine sau ceva nu vă mulțumeşte.

Aventură cu /dev/stdout

A sosit timpul să ne aventurăm în ungherele intime ale monitorului şi să privim câteva aspecte de bază ce alcătuiesc imaginea afişată pe ecran. Primul şi cel mai important este *pixelul*. Un pixel este un punct foarte mic şi colorat ce apare pe monitor. Rezoluția este exprimată în funcție de pixeli, deci putem spune că un pixel este cea mai mică unitate de măsură a imaginii. Astfel traducerea unei rezoluții de 1024x768, înseamnă 1024 de pixeli pe axa Ox (orizontală) si 768 de pixeli pe axa Oy (verticală) ceea ce totalizează 786432 pixeli pe tot ecranul. Cu cât numărul de pixeli este mai mare, cu atât rezoluția este mai bună adică pixelii sunt mai mici. Pe lângă rezoluție, mai există o variabilă de care depinde calitatea percepută a imaginii afişate de un monitor cu tub catodic, şi anume frecvența de cadre (sau "vertical refresh rate", cum sună termenul englezesc). Aceasta este frecvența cu care este reîmprospătat rastrul (imaginea) afişată. Acum voi discuta amănuntele şi cum puteți să vă calculați singuri aceste date, în funcție de monitor.



Variabilele ce compun display-ul monitorului

Şi am ajuns la miez

Dacă ce ați citit până acum vi s-a părut ceva "banal", atunci vă propun să aflăm împreună cum se calculează frecvența de cadre (*RR*). Dacă notăm frecvența de tact a RAMDAC-ului plăcii video (dot clock - câți pixeli pe secundă poate să genereze placa) cu *DC*, iar numărul de tacturi de RAMDAC necesare pentru ca monitorul să umple o linie pe orizontală, respectiv un cadru pe verticală, cu *BO* şi *BV*, obținem următoarea formulă de calcul: *RR = DC / (BO * BV)*. Aceasta se întâmplă deoarece *BO * BV* înseamnă de fapt numărul total de tacturi de *RAMDAC* pentru afişarea unui ecran întreg de puncte. Dacă împărțim viteza RAMDAC-ului la acest număr, obținem numărul de rastre (imagini) ce pot fi afişate într-o secundă. De aici rezultă dependența funcțională dintre *dot clock* (frecvența RAMDAC-ului), rezoluția curentă şi frecvența maximă de cadre: cu cât rezoluția este mai mare (în contextul aceluiaşi *RAMDAC*), cu atât frecvența maximă de cadre este mai mică.

Un popas în /etc/X11/XF86Config

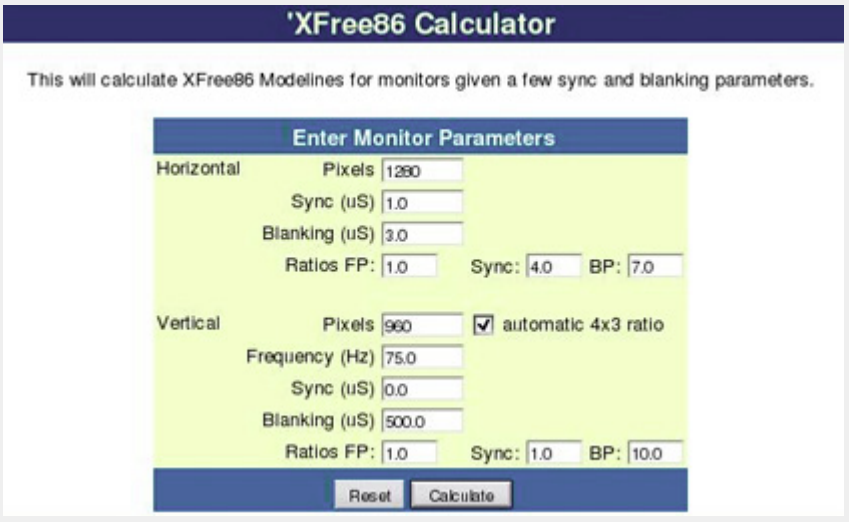
Până în clipa asta cu toții ar trebui să ştiți că în `/etc/X11/XF86Config` se găsesc liniile ce dictează server-ului X cum să pornească. Acesta este organizat în secțiuni. Pe noi ne interesează numai două secțiuni: "*Monitor*" şi "*Screen*". La secțiunea "*Monitor*" prezintă interes doar "*HorizSync*" şi "*VertRefresh*" unde se trec intervalele specificate în cartea monitorului pentru cele doua frecvețe de sincronizare, iar la "*Screen*" opțiunile "*DefaultDepth*" şi subsecțiunea "*Display*".

Opțiunea "*DefaultDepth*" lasă posibilitatea de a regla numărul implicit de culori folosite de server pentru a afişa imaginile (în biți). În subsecțiunea "*Display*" regăsim o opțiune similară acesteia, numită "*Depth*", dar care este folosită în mod curent şi nu în cazuri de urgență. Lângă aceasta opțiune găsim toate rezoluțiile permise serverului X. Acestea sunt scrise între ghilimele, în dreptul opțiunii "*Modes*". Pe lângă aceste setări mai există şi *modeline*-urile.

Ce este un *modeline*? Explicat simplu, un modeline este doar un nume pentru un set de parametri ce definesc un mod grafic suportat de placa video în uz. Un modeline se aşează ÎNTOTDEAUNA în interiorul secțiunii "*Monitor*" sub formă de subsecțiune. În interiorul acestui bloc se regăsesc trei opțiuni: "*DotClock*", "*HTimings*" şi "*VTimings*". De primul am mai vorbit, iar ultimele două se calculează în funcție de "*HorizSync*" şi "*VertRefresh*". Acestea se regăsesc (teoretic) în manualul monitorului. Pentru a vă da un exemplu de modeline al unui monitor ce dorim să afişeze o imagine cu o rezoluție de 1152x864 pixeli cu o frecvență de cadre de 75Hz, putem considera următoarele:

```
# 1152x864 @ 75Hz, 67.42 kHz hsync
Mode "1152x864"
DotClock 98.71
HTimings 1152 1184 1288 1464
VTimings 864 867 870 899
EndMode
```

Pentru a putea folosi acest model, este necesar ca RAMDAC-ul plăcii video să



Exemplu de configurație pentru calcularea unui modeline

fie capabil de a susține un "dot clock" de cel putin 98.71MHz (aplicabil pe exemplul dat). Prezentarea modeline-urilor şi modul serverului de a le manipula s-au schimbat de la apariția versiunii 4.0. Astfel, de la acea versiune, orice modeline incompatibil cu una din cele trei opțiuni (caracteristice monitorului în uz) este ignorat. Astfel se asigură (parțial) funcționarea în siguranță a monitorului. Acum, că am terminat cu teoria cea întunecată, vă voi oferi soluția practică de a calcula un modeline. În secțiunea *Resurse* de la finalul articolului veți găsi un link către o pagină unde un script vă calculează şi vă generează automat codul pentru un modeline în funcție de specificațiile monitorului vostru.

Ce-i în mână

Concluzia pe care o putem trage după toate aceste calcule şi căutări disperate de specificații pare complicată. Însă majoritatea producătorilor de distribuții au încorporat scripturi şi programe ce autodetectează setările în funcție de modelul monitorului. Aşa parcă ne mai liniştim. Indiferent care ar fi realitatea, curiozitatea şi dorința de cunoaştere împinge comunitatea să întrebe: "dar în spatele programului ce se află?"

```
File: XF86Config        Col 0                3177 bytes
# also use USB mice at the same time.
        Identifier  "DevInputMice"
        Driver      "mouse"
        Option      "Protocol" "IMPS/2"
        Option      "Device" "/dev/input/mice"
        Option      "ZAxisMapping" "4 5"
        Option      "Emulate3Buttons" "no"
EndSection

Section "Monitor"
        Identifier   "Monitor0"
        VendorName   "Monitor Vendor"
        ModelName    "AOC SPECTRUM 4V,4VA,4Vlr & 4VlrA, 4Vn, 4VnA"
        HorizSync    30.0 - 50.0
        VertRefresh  50.0 - 100.0
        Option      "dpms"
EndSection
```

Extract dintr-un XF86Config inainte de inserarea unui modeline

Resurse:

- http://www.zaph.com/Modeline/

Autor:

ciprian.negrila@linux360.ro

Shell Scripting - Puterea ascunsă a consolei

Silviu Foca

Făcusem în numărul 3 o introducere în Shell scripting pe care ne face plăcere să o continuăm din acest număr.

Reamintim că un shell este un interpretor de comenzi. Majoritatea shell-urilor ne oferă, suplimentar, câteva facilități cum ar fi: reutilizarea comenzilor executate anterior (*history*), completarea automată a numelor comenzilor introduse doar parțial (*autocompletion*) precum şi un limbaj propriu (*built-in*) ce ne permite realizarea de scripturi.

Mai spuneam că scripturile de interpretor (*shell script*) sunt o colecție de comenzi adunate în fişiere şi realizasem un prim script pentru shell-ul BASH.

Să detaliem puțin shell-ul BASH, shell-ul cel mai cunoscut şi folosit în Linux.

BASH îşi arată disponibilitatea de a primi comenzi prin afişarea unui prompt, un caracter sau cuvânt convențional ce poate varia în funcție de dorința noastră.

Inițial promptul este de tipul `[root@localhost root]#` pentru administratorul sistemului (root) şi `[user@localhost user]$` pentru restul utilizatorilor (în exemplu utilizatorul <user>). Deci este format din numele utilizatorului, numele maşinii şi numele relativ (scurt) al directorului curent. În exemplificările de mai jos vom folosi doar $ sau # pentru simbolizarea promptului.

O comandă shell în forma sa cea mai simplă este de tipul:

```
numeprogram argumentul1
argumentul2 ... argumentuln
```

Ca exemplu este prezentat codul următor:

```
$ cat /etc/shells
/bin/bash2
/bin/bash
/bin/sh
/bin/ash
/bin/bsh
/bin/tcsh
/bin/csh
```

Numele programului este `cat` iar primul şi unicul argument este `/etc/shells`. Conținutul acestui fişier este lista tuturor shell-urilor disponibile pe acel sistem.

stdin, stdout, stderror

Un script lucrează cu 3 canale de comunicație fundamentale:

- **stdin** - de unde se primesc datele de intrare. Reprezintă intrarea standard şi, în mod normal, corespunde tastaturii.
- **stdout** - unde vor fi afişate rezultatele furnizate de script. Ieşirea standard corespunde ecranului calculatorului.
- **stderror** - unde vor fi afişate mesajele de eroare. Standard error corespunde în mod implicit ecranului.

Shell-ul permite redirecționarea acestor intrări/ieşiri standard, lucru mai mult decât util atunci când dorim să scriem scripturi. Cel mai simplu exemplu de redirecționare este acela de a trimite ieşirea unui program (implicit rezultatul fiind afişat pe ecran) într-un fişier cu ajutorul caracterului ">".

Exemplu:

```
$ cat /etc/shells > rezultat
```

Fişierul `rezultat` va fi o copie a fişierului `/etc/shells` al cărui conținut în loc să fie trimis către ecranul calculatorului a fost pus în fişierul `rezultat`. Dacă fişierul `rezultat` nu există, acesta va fi creat chiar dacă output-ul programului nu va conține nimic, iar în cazul în care fişierul există deja va fi suprascris, astfel încât conținutul precedent va fi pierdut.

Atunci când comenzii `cat` nu i se specifică numele fişierului de citit se vor folosi ca date de intrare informațiile introduse de la tastatură.

Exemplu de creare a unui fişier `numere` prin introducere directă.

```
$ cat > numere
10 22
35
19
^D
```

Simbolul "^D" reprezintă tastarea secvenței de taste `CTRL-D` aceasta producând caracterul cu codul 4 zecimal ce are semnificatia `EOF` (sfârşit de fişier). Acum avem fişierul `numere` ce conține trei linii cu numere.

În această situație, `cat` ştie să preia datele atât de la tastatură cît şi dintr-un fişier (deci nu trebuie să redirecționăm noi input-ul). Însă nu toate aplicațiile sunt implementate în acest mod, caz în care vom specifica noi de unde să fie preluate datele de intrare.

Pentru indicarea unui fişier ca intrare standard se foloseşte semnul "<".

Exemplu: Presupunem că avem programul "pare" care are rolul de a citi un şir de numere de la tastatură şi a specifica dacă numerele introduse în el sunt pare sau impare. Dacă dorim ca numerele să fie citite dintr-un fişier vom redirecționa intrarea standard.

```
$ pare < numere
10 par
22 par
35 impar
19 impar
```

De asemenea, redirecționările pot fi înlănțuite, astfel încât atât intrarea standard cât şi ieşirea standard să fie reprezentate de fişiere.

Exemplu:

```
$ pare < numere > rezulate
```

Aceste redirecționări pe care le-am exemplificat până acum nu interacționează cu `standard error`. De exemplu să presupunem că tastăm greşit o comandă:

```
$ cat numre > copienumere
cat: numre: No such file or directory
```

După cum se vede, chiar dacă am făcut redirecționarea către fişierul `copienumere`, eroarea a fost afişată pe ecranul calculatorului. Este posibil să facem şi redirecționarea erorilor către un fişier prin precedarea semnului ">" de numărul 2 care este numărul descriptorului de fişier asociat (ca standard universal UNIX) cu `standard error`. În aceeaşi idee, `stdin` are descriptorul 0 iar `stdout` are descriptorul 1.

Exemplu:

```
$ cat numre > copienumere 2>erori_cat
```

În urma acestei comenzi fişierul `copienumere` va fi gol iar în fişierul `erori_cat` vom regăsi mesajul "`cat: numre: No such file or directory`".

Redirecționarea se poate face în aşa fel încât output-ul curent să fie scris la sfârşitul fişierului deja existent (adică nu prin ştergerea conținutului acestuia) utilizând secvența ">>" în locul semnului ">". Şi în acest caz, fişierul va fi creat în eventualitatea în care nu există.

Exemplu:

Dacă dorim adăugarea unei alte linii la fişierul `numere`:

```
$ cat >> numere
81 92 107
^D
$ cat numere
10 22
35
19
81 92 107
```

PIPE

Foarte des este avantajos să legăm output-ul unei comenzi cu input-ul alteia. Semnul care face această conectare între un output şi input este "`|`" iar mecanismul se numeşte "**pipe**" (conductă). Să presupunem că vrem să selectăm din output-ul programului `pare` (din exemplul de mai sus) doar liniile care conțin numere pare. Pentru aceasta vom folosi comanda `grep` care are rolul de a face filtrarea rezultatelor şi afişarea doar a acelora ce conțin un anumit termen, în cazul nostru "par".

Exemplu:

```
$ pare < numere | grep par
10-par
22-par
92-par
```

Funcționarea acestei comenzi este următoarea:

1. Programul `pare` primeşte input-ul din fişierul `numere` aşa cum este indicat de semnul "<".

2. Acesta prelucrează fişierul şi trimite către ieşirea standard rezultatul (lista cu numere în dreptul cărora scrie par sau impar) care este conectat cu input-ul comenzii `grep` după cum indică semnul "|".

3. Comanda `grep`, negăsind nici un nume de fişier între argumentele proprii după cuvântul cheie ("par"), prelucrează input-ul primit de la `pare` şi selectează doar liniile ce corespund termenului de căutare (în acest caz, "par") şi le trimite către ieşirea standard (output), monitorul calculatorului.

Caractere JOLLY

Se întâmplă uneori ca argumentele unei comenzi să fie o listă de fişiere ale căror nume diferă foarte puțin sau au o parte comună. Pentru a evita specificarea în linia de comandă a acestor nume, shell-ul ne pune la dispoziție macanismul numit "**caractere jolly**" (Shell wildcards. De asemenea, mecanismul mai este cunoscut şi sub denumirea de "**shell globbing**").

Cele două principale caractere jolly sunt "*****" şi "**?**" şi vor fi utilizate de către shell după cum urmează:

- ***** - Fiecare apariție a acestui caracter poate fi înlocuită cu orice număr de caractere (chiar şi zero) diferite de "/"
- **?** - Acest caracter poate fi înlocuit doar de un singur caracter diferit de "/"

Să presupunem, de exemplu, că avem într-un director cinci fişiere: `prog.c`, `prog.h`, `makefile`, `manual.txt` şi `prog.exe`.

```
$ ls
makefile  manual.txt  prog.c
prog.exe prog.h
$ ls *
makefile  manual.txt  prog.c
prog.exe prog.h
$ ls prog.*
prog.c prog.exe prog.h
$ ls prog.?
prog.c prog.h
$ ls ma*
makefile manual.txt
```

O utlilizare mai avansată şi mai utilă a acestor caractere jolly poate fi căutarea fişierelor ce se termină în '.c' în directorul curent şi în toate subdirectorele:

```
$ find . -name "*.c" -print
./prog.c
./src/src1.c
./src/src2.c
```

Dacă nu am fi folosit ghilimele în

comanda de mai sus, rezultatul ar fi fost altul şi asta deoarece shell-ul face întâi expansiunea caracterelor jolly apoi execută comenzile aferente.

```
$ find . -name *.c -print
./prog.c
```

În cazul în care trebuie să selectăm anumite fişiere din liste ce cuprind sute sau mii de fişiere putem utiliza şi metoda intervalelor.

Exemplu

```
$ ls [n-z]*
prog.c prog.exe prog.h

src:
src1.c src2.c
```

Au fost selectate doar fişierele ce încep cu o literă din intervalul n-z.

VARIABILE de MEDIU

Variabilele de mediu memorează unele definiţii (făcute de obicei la deschiderea sistemului) cu scopul de a face disponibile unele valori fundamentale necesare în funcționarea diferitelor programe. Fiecare variabilă are un nume şi o valoare. Utilizând **BASH**, comanda pentru definirea unei variabile este:

```
$ NUMEVARIABILA=valoare
```

Această definiţie va putea fi folosită doar de către shell-ul în care s-a dat comanda; dacă se doreşte ca aceasta să fie validă şi pentru alte programe trebuie să folosim comanda:

```
$ export NUMEVARIABILA
```

Şi, pentru a simplifica totul printr-o singură linie, putem scrie

```
$ export
NUMEVARIABILA=valoare
```

De asemenea, BASH mai are şi o comandă complementară care are rolul de a se asigura că variabila definită va fi vizibilă doar în funcția sau scriptul curent şi anume "`local`".

Cele mai importante variabile de mediu sunt următoarele:

- `USER` - numele utilizatorului
- `HOME` - directorul home al utilizatorului
- `PATH` - lista cu directoare în care shell-ul va căuta comenzile pentru execuție. A se observa că în această variabilă poate fi inclus şi '.' adică directorul curent - există însă o polemică pe marginea acestui fapt: există multe păreri pro şi contra includerii lui "./" în PATH, în marea lor majoritate legate de securitate.
- `PWD` - directorul curent
- `MANPATH` - lista de directoare unde shell-ul va căuta manualele
- `TERM` - tipul de terminal pe care se lucrează, această variabilă fiind utilă de exemplu unui editor de texte
- `PS1` - promptul. BASH permite definirea promptului într-o manieră foarte rafinată.

Pentru a vizualiza valoarea unei variabile se foloseşte comanda:

```
$ echo $NUMEVARIABILA
```

Prin semnul '$' am indicat că ne referim la o variabilă şi nu la şirul de caractere 'NUMEVARIABILA'.

Lista tuturor variabilelor definite la un moment dat o putem avea cu ajutorul comenzii '`env`' sau '`set`'.

După cum spuneam, **BASH** ne permite să definim prompt-ul după plăcerea şi gustul nostru. Pentru a infirma puțin concepția prin care consola este recunoscută ca fiind alb/negru şi urâtă, vă voi prezenta un mic truc prin care veți putea da puțină culoare şi diversitate consolei la care lucrați.

În mod normal, după instalarea sistemului veți regăsi într-unul dintre fişierele de configurare (`/etc/profile`) variabila de mediu `PS1` de forma `[\u@\h \w]` care va spune sistemului să vă afişeze un prompt de forma

```
[user@hostname director]
```

Pentru a putea schimba culoarea promptului, vom folosi codurile de culori din BASH care sunt reprezentate prin următoarele numere.

Codurile atributelor:

- **00**=nici o proprietate
- **01**=bold
- **04**=subliniat
- **05**=intermitent
- **07**=inversat
- **08**=ascuns

Codurile pentru text:

- **30**=negru
- **31**=roşu
- **32**=verde
- **33**=galben
- **34**=albastru
- **35**=magenta
- **36**=cyan
- **37**=alb

Codurile pentru fundal:

- **40**=negru
- **41**=rosu
- **42**=verde
- **43**=galben
- **44**=albastru
- **45**=magenta
- **46**=cyan
- **47**=alb

Folosind aceste coduri de culori vom putea defini promptul dupa dorința noastră, de exemplu:

```
PS1="\[\e[01;32m\][\u@\h \W]
#"
```

va produce un prompt verde cu caractere bold. Însă şi tot textul pe care îl vom tasta în consolă va fi de asemenea verde. Pentru a preveni acest lucru vom adăuga la sfârşitul şirului codul de culoare pentru text alb şi de mărime normală astfel încât variabila noastră va avea forma:

```
PS1="\[\e[01;32m\][\u@\h \W]
#\[\e[00;37m\]"
```

Plecând de la aceste exemple se pot

combina în cele mai variate moduri culorile şi stilurile promptului ce va da un pic de viaţă bătrânei console.

JOB

Deoarece unele programe au nevoie de un timp mai mare pentru a fi executate, shell-ul ne pune la dispoziţie unele mijloace de a putea continua folosirea sistemului în timp ce aceste programe sunt executate. Această operaţie se numeste rularea în background (fundal) a programelor iar rularea normală a programelor se numeşte foreground (prim plan).

Un program lansat în background se numeste '**job**'. Fiecare shell activ are o lista de job-uri căroara le asociază câte un număr.

Să presupunem că avem un program ce efectuează calcule ce au nevoie de mult timp pentru a fi terminate şi dorim lansarea acestuia în background:

```
$ calcule &
[1] 2292
```

[1] - semnifică numărul de job
2292 - semnifică numărul procesului (PID)

Aceste programe lansate în background folosesc standard output şi error pentru a afişa rezultatele lor. Pentru a nu interfera cu alte aplicaţii pe care le rulăm este indicat să se facă redirecţionarea rezultatelor.

Exemplu:

```
$ calcule > calculeoutput &
```

În schimb, un program lansat în background nu dispune de tastatură ca standard input aceasta fiind lăsată shell-ului pentru a avea posibilitatea de a introduce noi comenzi. Totuşi, pentru a putea interacţiona cu programele lansate în background, shell-ul ne permite comutarea acestora între background şi foreground.

Comanda **jobs** ne permite să vedem lista proceselor lansate în background precum şi numerele de identificare ale acestora.

Să presupunem că dorim lansarea programului `calcule` cu trei input-uri diferite din fişiere:

```
$ calcule < calculeinput1 > calculeoutput1 &
[1] 101
$ calcule < calculeinput2 > calculeoutput2 &
[2] 103
$ calcule < calculeinput3 > calculeoutput3 2> calculeerori &
[3] 105
$ jobs
[1]-  Running  calcule  < calculeinput1  > calculeoutput1 &
[2]-  Running  calcule  < calculeinput2  > calculeoutput2 &
[3]-  Running  calcule  < calculeinput3  > calculeoutput3 2> calculeerori &
```

În cea de-a treia comandă, standard error este redirecţionat în fişierul `calculeerori`.

Pentru a termina un proces din cele trei pe care le avem în background vom folosi comanda kill având ca argument numarul de job precedat de semnul "%"

```
$ kill %1
[1]-  Terminated  calcule  < calculeinput1 >calculeoutput1 &
```

În cazul în care lansăm un proces în foreground şi dorim mutarea acestuia în background ne vom folosi de secvenţa de taste `CTRL-Z (^Z)` care stopează temporar procesul, redând controlul consolei cu care putem trimite în background procesul prin comanda '`bg`' având ca argument numărul de job al procesului.

Pentru trecerea unui proces în foreground vom folosi comanda '`fg`' cu argument numărul job-ului precedat de semnul '%'.

Aceste două comenzi, `bg` şi `fg` nu sunt programe de sine stătătoare ci sunt direct implementate în `shell`.

Exemplu:

```
$ calcule
^Z
[4]+ Stopped calcule
$ bg %4
[4]+ calcule &
$ fg %2
$ calcule < calculeinput1 > calculeoutput1 &
```

În exemplul de mai sus, am lansat în foreground programul `calcule`, apoi l-am stopat cu secvenţa `CTRL-Z` ca să îl putem trece în background şi să readucem în foreground procesul cu numărul de job 2.

Bineînţeles că putem reutiliza această tehnică de câte ori este nevoie până la terminarea procesului.

SCRIPTURILE DE SHELL

Un script de shell este un fişier de text ce conţine o secvenţă de comenzi ce vor fi executate de către shell în scopul automatizării unor procese.

În mod normal scripturile sunt executate de către shell-ul pe care îl rulăm în acel moment pe sistem. Pentru a specifica explicit numele programului ce dorim să interpreteze comenzile din script trebuie să menţionăm pe prima linie a scriptului numele acestuia cu toată calea către el precedat de caracterele "#!"

Exemplu:

```
#!/bin/bash
```

Este o foarte bună obişnuinţă să specificăm programul ce va executa comenzile chiar dacă acesta este shell-ul implicit pe acel sistem.

Un simplu script care execută o arhivare a unor fişiere este următorul:

```
#!/bin/bash
```

```
cp C/project.c /tmp
cp H/project.h /tmp
cd /tmp
tar cvf project.tar project.c project.h
cp project.tar $HOME
rm -f project.*
cd $HOME
```

A se nota folosirea în script a variabilei de mediu $HOME pentru a indica propriul director home. În cazul BASH, aceasta poate fi înlocuită şi cu caracterul "~" cu aceeaşi semnificație.

Scriptul trebuie făcut executabil prin intermediul comenzii '`chmod +x numescript`' şi poate fi rulat tastând '`./numescript.sh`' sau, dacă nu este executabil, `bash numescript.sh`

Scripturile sunt executate de un shell diferit de cel în care se face apelarea. Acest nou shell se lansează în momentul începerii execuției scriptului, execută toate comenzile din script şi se sfârşeşte odată cu scriptul redând controlul shell-ului inițial.

De asemenea scriptul poate fi executat şi de către shell-ul actual prin apelarea sa de către comanda '`source numescript.sh`'.

După cum ați observat, este indicat ca scripturile să le terminați cu "extensia" `.sh` pentru a se deosebi de celelalte fişiere din director, deşi în principiu pot avea orice denumire.

Pentru a realiza scripturi cât mai complexe, **BASH** dispune de un limbaj intern (built-in) care se aseamănă foarte mult cu limbajele moderne de programare.

Acest limbaj conține printre altele atât binecunoscutele structuri `if`, `while`, `for` precum şi posibilitatea declarării de variabile.

TEST

Orice program în Unix returnează programului care l-a lansat un număr ca rezultat al propriei execuții. Shell-ul fructifică această opțiune pentru a efectua unele teste. Valoarea zero va fi interpretată drept "True" (Adevărat) iar orice altă valoare ca "False" (Fals). Există o serie întreagă de programe concepute special pentru aceasta însa cel mai folosit dintre ele este de departe 'test', care ajută la determinarea diferitelor proprietăți ale fişierelor şi la compararea şirurilor de caractere între ele. În realitate, datorită unor chestiuni de eficiență, multe shell-uri printre care şi **BASH** au o versiune proprie a acestei comenzi în aşa fel încât să evite executarea unui nou program în fiecare situație ce necesită o verificare cu `test`.

Fiecare opțiune cu care este lansat `test` duce la un anumit tip de verificare dintre care cele mai importante sunt:

- `test -d file` - Adevărat dacă argumentul există şi este un director
- `test -f file` - Adevărat dacă argumentul există şi este unul obişnuit (nu director, legătura simbolică, ...)
- `test -e file` - Adevărat dacă argumentul există
- `test -L file` - Adevărat dacă argumentul există şi este o legatură simbolică
- `test -r file` - Adevărat dacă argumentul există şi poate fi citit (în contextul curent de acces)
- `test -x file` - Adevărat dacă argumentul este un fişier executabil (+x)
- `test -z sir` - Adevărat dacă şirul de caractere are lungime mai mare decât zero
- `test sir1 = sir2` - Adevărat dacă şirurile de caractere sunt egale
- `test sir1 != sir2` - Adevărat dacă şirurile sunt diferite

În mod uzual, în scripturi se foloseşte o formă particulară a comenzii `test` pentru a facilita citirea şi înțelegerea mai uşoară a scriptului, şi anume în loc de `test -f file` se foloseşte `[ -f file ]`

VARIABILE

Shell-ul ne permite declararea de variabile şi folosirea acestora în interiorul scripturilor noastre. Două categorii aparte de variabile sunt:

- variabilele de mediu
- variabilele de shell

Dacă variabilele de mediu le-am detaliat la începutul acestui articol, vom prezenta în continuare câte ceva despre variabilele de shell.

Acestea sunt şiruri cu un singur caracter a căror valoare o putem vedea cu ajutorul operatorului "$".

Cele mai importante variabile de shell sunt:

- `$#` - memorează numărul de argumente ce sunt transmise scriptului din linia de comandă
- `$?` - memorează valoarea terminării ultimului program sau comenzi executate
- `$0` - memorează numele scriptului ce a fost lansat
- `$*` - memorează toate argumentele ce au fost introduse în linia de comandă
- `$!` - conține numărul ultimului proces ce a fost trimis în background cu "&"

Vă prezint în continuare un mic script ce are ca scop evidențierea lucrului cu variabile în shell:

```
#!/bin/bash
var=valoarea1
echo var=$var
var=valoare2
echo var=$var
echo "Numarul de parametri ="$#
echo "Numele programului ="$0
echo parametrii introdusi ="$*
```

Salvați acest script cu ce nume doriți, faceți-l executabil şi lansați-l cu câteva argumente pe linia de comanda: `./script arg1 arg2 argN`

Structura IF - THEN - ELSE

Sintaxa acestei structuri este:

```
if [ expresie ]
then
comanda1
```

```
elif
comanda3
else
comanda2
fi
```

Dacă în cazul limbajelor de programare moderne, buclele `if` nu mai sunt încadrate între acolade sau alte semne, în bash pentru a specifica închiderea buclei `if` se foloseste cuvântul cheie '`fi`'.

De asemenea se poate renunța la cea de-a doua parte a structurii, şi anume la partea acoperită de către '`else`'.

Pentru realizarea unor teste mai complexe se pot folosi mai multe `if`-uri în cascadă cu condiția să se respecte sintaxa pentru fiecare dintre ele în parte.

Pentru înțelegerea acțiunii lui '`if`' vă propun un exemplu care probează existența unui fişier şi, în cazul în care acesta este găsit, îi afişează conținutul.

```
#!/bin/bash
if [ -f fisier ]
then
more fisier
else
echo "Fisierul nu exista"
fi
```

Operatorii logici

Bash conține trei operatori logici principali **AND**, **OR**, **NOT** având acelaşi înțeles ca şi în celelate limbaje de programare şi anume: **AND** - verifică dacă ambele condiții sunt satisfăcute, **OR** - rezultă adevărat dacă cel puțin una dintre condiții este satisfăcută iar **NOT** semnifică negația condiției inițiale. În scripturi aceşti operatori sunt folosiți prin intermediul unor caractere speciale dupa cum urmeaza:

```
AND - &&, OR - ||, NOT - !
```

În continuare aveți un script ce vă prezintă cei trei operatori logici la lucru:

```
#!/bin/bash
var1=0
var2=1

#Vizualizam          valorile
initiale
#ale variabilelor
echo var1=$var1, var2=$var2


#utilizarea lui AND
if [ $var1 = 0 ] && [ $var2
= 1 ]
then
echo  $var1=0  AND  $var2=1  :
Adevarat
fi


#utilizarea lui OR
if [ $var1 = 1 ] || [ $var2
= 1 ]
then
echo  $var1=1  OR  $var2=1  :
Adevarat
fi


#utilizarea lui NOT
if ! [ $var = 1 ]
then
echo NOT $var=1 : Adevarat
fi
```

Pe lângă operatorii logici, bash ne oferă şi posibilitatea lucrului cu operatori şi expresii matematice.

Cei mai importanți operatori matematici ar fi:

- - + * / % - scădere, adunare, înmulțire, împarțire, rest
- ! ~ - negație logică bit cu bit
- == != - egalitate, diferență
- & - AND bit cu bit
- ^ - OR exclusiv bit cu bit
- | - OR bit cu bit
- && - AND logic
- || - OR logic

Operațiile matematice sunt efectuate pe numere întregi cuprinse în intervalul 2147483647 şi -2147483647, iar pentru a evalua o expresie se foloseşte sintaxa `$[expresie]`

Ciclul FOR

Cu această structură este posibilă executarea a aceleiaşi serii de comenzi de un anumit număr de ori sau pentru fiecare element dintr-o listă dată.

Sintaxa secvenței `for` este:

```
for variabila in valorea1
valorea2
do
comanda
done
```

Pentru a înțelege mai bine funcționarea ciclului `for`, vom exemplifica cu următorul script:

```
#!/bin/bash
for var in valoare1 valoare2
valoare3
do
echo var=$var
done
```

Ciclul WHILE

Şi aceasta este o structură clasică ce se regăseşte în marea majoritate a limbajelor de prgramare şi are următoarea sintaxă:

```
while [expresie]
do
comanda
done
```

De asemenea vă propun un script mic pentru mai buna întelegere a acestei structuri.

```
#!/bin/bash
var=3
while [ $var != 0 ]
do
echo var = $var
var=$[$var-1]
done
```

Deseori în crearea scripturilor este nevoie de a interacționa cu utilizatorul pentru unele informații suplimentare pentru executarea în bune condiții a scriptului. Pentru a primi date de la utilizator în timpul execuției scriptului vom folosi instrucțiunea `read`.

Aceasta are scopul de a seta o variabilă cu o anumită valoare pe care o

citeşte de la standard input, adică de la tastatură.

Există şi posibilitatea introducerii mai multor variabile odată, însă fiecărei variabile i se va repartiza câte un şir de caractere delimitat de spațiu (adică un cuvânt pentru fiecare variabilă), ultimei variabile fiindu-i repartizat tot restul input-ului.

Exemplu

```
#!/bin/bash
echo Introduceti valoarea variabilelor var1 var2 var3
read var1 var2 var3
echo var1=$var1
echo var2=$var2
echo var3=$var3
```

După execuție rezultatul va fi următorul:

```
[root@localhost]
# ./numescript
Introduceti valoarea variabilelor var1 var2 var3
valoarea1 valoarea2 valoarea3 si restul valorilor
var1=valoarea1
var2=valoarea2
var3=valoarea3 si restul valorilor
```

Instrucțiunea CASE

Această intrucțiune poate fi considerată o versiune mai puternică a structurii `if-then-else` ce permite o discriminare între un număr mai mare de posibilități.

Sintaxa este următoarea:

```
case variabila in
valoarea1) comanda1 ;;
valoarea2) comanda2 ;;
*) comandaN ;;
esac
```

unde '*) comandaN' reprezintă ceea ce se va executa dacă variabila nu va îndeplini nici una dintre condițiile anterioare. Exemplu:

```
#!/bin/bash
read -n 1 -p "Tastati un caracter:" var
echo ""
case $var in
[0-9]) echo "Ati tastat un numar" ;;
[a-z] | [A-Z]) echo "Ati tastat o litera" ;;
*) echo "Ati tastat altceva"
esac
```

Putem desemna unei variabile şi rezultatul execuției unui program ca de exemplu în următorul script care este o versiune extrem de primitivă a comenzii `file`.

```
#!/bin/bash
var=`ls`
echo var=$var
for i in $var
do
if [ -f $i ] ; then
echo $i "este fisier"
elif [ -d $i ] ; then
echo $i "este director"
else
echo $i "este altceva"
fi
done
```

Ca orice alt limbaj de programare ce se respectă, **bash** include şi posibilitatea declarării de funcții, care sunt părți de program pe care le putem apela în interiorul scriptului cu numele cu care au fost declarate.

Exemplu

```
#!/bin/bash
varfunc=NEsetata
function functie()
{
echo "Execut functia"
varfunc=setata
}


echo "Incepem scriptul"
echo varfunc=$varfunc
functie
echo varfunc=$varfunc
echo "Terminat scriptul"
```

După cum se vede în scriptul anterior, comenzile din interiorul funcției sunt ignorate de către shell până în momentul apelării acesteia cu numele declarat.

Acestea sunt elementele de bază ale shell scripting-ului pe care vă invit să le studiați cu atenție şi în cazul în care aveți nevoie de mai multe detalii despre aceste structuri '**man bash**' este un prieten de neînlocuit.

Pentru finalul acestei prezentări vă propun realizarea unui script funcțional şi înțelegerea tuturor elementelor din care este compus.

Scriptul la care mă refer face o salvare periodică a unor fişiere şi directoare pe care le considerăm noi importante, această tehnică se numeşte backup şi ar trebui să fie luată în considerare de către oricine lucrează în acest domeniu deoarece oricând se poate întâmpla o "nenorocire" şi să rămânem fără munca depusă timp îndelungat, zile sau chiar luni.

În câteva cuvinte, scriptul nu face altceva decât să citească o listă de directoare dintr-un fişier text standard (am numit acest fişier `lista.txt`), ca apoi să facă o arhivă cu toate fişierele aflate în aceste directore.

Pentru reducerea timpului de utilizare a resurselor hardware ale calculatorului am implementat şi metoda backup-ului incremental care face arhivarea doar a fişierelor ce au fost modificate în ultima zi, astfel incât calculatorul să nu fie nevoit să arhiveze zilnic sau săptămânal unele fişiere ce nu sunt modificate timp îndelungat.

În mod normal directorul în care se va face stocarea arhivelor este '`/tmp/backup`', însă acesta poate fi modificat prin editarea variabilei `$DIR` de la începutul scriptului. Deasemenea ziua în care se va face un backup complet este setată a fi duminică însă şi aceasta poate fi schimbată cu uşurință prin modificarea valorii variabilei `$ZIFULL`.

Pentru mai buna înțelegere a scriptului am adăugat câteva comentarii pentru aproximativ fiecare parte de cod ce

îndeplineşte o anumită funcție în script.

Vă invit să analizați scriptul, să încercați să înțelegeți ce fac toate comenzile din el, iar când descoperiți lucruri sau opțiuni noi vă îndrum către '**man comanda**' pentru mai bună înțelegere a respectivelor comenzi sau opțiuni.

```
#!/bin/sh
#Declararea variabilelor
ZIUA='date +%w' #Reprezinta ziua din saptamana, Duminica=0, Luni=1, etc...
DIR="/tmp/backup"
#Directorul unde vor fi salvate arhivele de backup
ZIFULL=0 #Ziua din saptamana in care se va face un backup complet, default Duminica ( 0 )
LISTADIR="lista.txt"
#Fisierul din care se vor citi directoarele ce vor fi salvate in arhiva de backup


function folosinta()
{
echo "Folosinta: `basename $0` fisier_lista_backup director_backup zi_backup_full"
echo "Exemplu: `basename $0` lista.txt /tmp/backup 0"
echo "Aceasta comanda va face backup-ul citind directoarele din fisierul lista.txt"
echo "stocand arhiva in directorul /tmp/backup iar daca este Duminica face full backup"
exit
}


case "$#" in
0 ) #se vor folosi setarile default
;;
3 )
#Verificam daca primul argument este un fisier
if [ -f $1 ]
then
LISTADIR="$1"
else
folosinta
fi

#Verificam daca cel de-al doilea argument este un director
if [ -d $2 ]
then
DIR="$2"
else
folosinta
fi


#Verificam daca al treilea argument este un numar si se incadreaza in intervalul 0-6
if [ $3 -gt -1 ] && [ $3 -lt 7 ]
then
ZIFULL=$3
else
folosinta
fi
;;

#Pentru un numar de argumente diferit de 0 sau 3 vom arata modul de folosinta si vom termina scriptul
* ) folosinta
;;
esac

#Controlam existenta directorului in care se vor salva arhivele, iar daca acesta nu exista il vom crea
if [ ! -d $DIR ]
then
mkdir $DIR
echo "Am creat directorul $DIR"
fi

#Verificam ziua din saptamana pentru a decide tipul de backup ce va fi facut
if [ "$ZIUA" = "$ZIFULL" ]
then
#backup complet

#arhivarea directoarelor specificate
tar uf "$DIR/backup_saptamanal.tar" -T $LISTADIR 2> $DIR/errori_backup_saptamanal.txt

#crearea unui fisier text cu numele fisierelor continute in arhiva
tar tf "$DIR/backup_saptamanal.tar" > $DIR/lista_fisiere_backup_saptamanal.txt
else
#backup incremental

#transformarea listei intr-o singura linie
cat $LISTADIR | tr -s '\n' ' ' > tmp.txt

#declararea variabilei $LISTA pentru a ne usura munca ;)
LISTA="$DIR/lista_fisiere_backup_zilnic-$ZIUA.txt"

#crearea listei cu fisierele modificate in ultima zi
find `cat tmp.txt` -depth -type f \( -ctime -1 -o -mtime -1 \) -print > $LISTA

#stergerea backup-ului de saptamana trecuta din aceeasi zi
rm -f $DIR/backup_zilnic-$ZIUA.tar

#arhivarea fisierelor
tar cf "$DIR/backup_zilnic-$ZIUA.tar" -T $LISTA 2> $DIR/errori_backup_zilnic-$ZIUA.txt

#stergerea fisierelor temporare
rm -f tmp.txt
fi
echo "Backup terminat"
```

În cazul în care scriptul este rulat fără nici un parametru, vor fi folosiți parametrii standard pe care i-am declarat în partea inițială. Scriptul este conceput să fie rulat zilnic prin introducerea acestuia în `crontab`.

Vă aşteptăm cu îmbunătățiri la acest script pe forumul linux360. Spor la scripting!

Autor:

silviu.foca@linux360.ro

Expresii regulate

Radu - Eosif Mihăilescu

Iată că ne întâlnim şi în cadrul secţiunii de programare - de data acesta pentru a vorbi despre un concept mai puţin legat de programarea propriu-zisă şi mai mult de ceea ce se numeşte "shell scripting" - este vorba de "expresii regulate".

Ce este aceea o expresie regulată? O expresie regulată este un model, o descriere de caracteristici, un şablon care se aplică (în sensul că defineşte o mulţime) unui set de şiruri de caractere. O expresie regulată este, dacă doriţi, un criteriu de comparaţie sau expresia unei interogări pe un set de şiruri de caractere.

Poate părea complicat şi abstract la început aşa că o să dăm nişte exemple: cea mai simplă expresie regulată este cea nulă (adică ""). Această expresie regulată descrie mulţimea vidă a şirurilor de caractere, adică nici un şir - indiferent de datele de intrare.

Un exemplu complementar este ".*" adică expresia regulată care descrie mulţimea formată din toate (semantica este de "oricare") şirurile de caractere - vom vedea puţin mai târziu de ce a fost scrisă "punct-asterix" şi nu oricum altcumva.

Să dăm un ultim exemplu mai complicat înainte de a trece la tratarea sintaxei expresiilor regulate: "^-l[ ]?.+ $". Această expresie regulată descrie mulţimea şirurilor de caractere care încep ("^") cu "-l", continuă cu un spaţiu facultativ ("[ ]?") şi se încheie ("$") cu unul sau mai multe caractere (oricare ar fi acelea) urmate de un spaţiu (obligatoriu) - ".+ ".

Să trecem acum la:

Sintaxa expresiilor regulate

Unitatea morfologică fundamentală a unei expresii regulate este expresia regulată care descrie un singur caracter. Majoritatea literelor, cifrelor şi a altor caractere reprezintă expresii regulate de un caracter care se descriu pe ele însele. De exemplu "A" este o expresie regulată care descrie mulţimea şirurilor de caractere de forma "A". Caracterele care au o însemnătate aparte în cadrul expresiilor regulate se numesc "metacaractere" - acestea pot fi "retrogradate" la rolul de caractere simple prin precedarea cu "\" (backslash).

Se numeşte expresie-mulţime o listă de caractere încadrată între paranteze pătrate. O asfel de expresie regulată descrie şiruri de caractere formate din oricare singur caracter din acea listă. Dacă primul caracter al listei este "^", atunci definiţia multimii este negată. Exemple: "[0123456789]" descrie şirurile de caractere formate dintr-o cifră oarecare iar "[^abC]" descrie şirurile de caractere care NU sunt "a", "b" sau "C".

Într-o expresie-mulţime, se numeşte expresie-interval o expresie formată din două caractere despărţite de o cratimă ("-" - minus sau hyphen). O asfel de expresie este o "scurtătură" pentru a descrie o mulţime - de exemplu, "[1-3]" este echivalent cu "[123]". Trebuie să mai ştiţi că există câteva denumiri standard pentru câteva expresii-intervale foarte uzuale, cum ar fi "[:alnum:]" pentru "[0-9A-Za-z]". Atenţie: acestea sunt denumiri pentru expresia-interval şi nu pentru cea mulţime - rezultă că va trebui să mai adăugaţi o pereche de paranteze pătrate pentru a obţine expresia-mulţime.

Majoritatea metacaracterelor îşi pierd însemnătatea atunci când apar în cadru unei expresii-mulţime. Rezultă că, pentru a include o paranteză pătrată închisă ("]") în expresie aceasta trebuie scrisă prima în listă; pentru a include un caret (sau accent circumflex - "^") plasaţi-l oriunde în afară de prima poziţie iar pentru a include o cratimă, plasaţi-o ultima.

Mergând mai departe, avem următoarele expresii regulate predefinite:

- ".": această expresie înseamnă "orice caracter". De exemplu, "a" si "(" sunt descrise de ea
- "^" şi "$": aceste expresii regulate descriu şirul vid de caractere aflat între începutul rândului şi primul caracter respectiv între ultimul caracter şi sfârşitul rândului. Acestea se folosesc de obicei în combinaţie cu alte expresii pentru a obţine criterii complexe de selecţie - la fel ca următoarele
- "\<" şi "\>": aceste expresii regulate descriu şirul vid de caractere aflat la începutul, respectiv sfârşitul unui cuvânt.

În fine, o expresie regulată poate fi urmată de următorii operatori de repetiţie (cei mai folosiţi):

- "?": elementul anterior este opţional şi se aplică cel mult o singură dată
- "*": elementul anterior se aplică de zero sau mai multe ori
- "+": elementul anterior se aplică cel puţin o dată.

Aceasta ar fi, în mare, "demitizarea" expresiilor regulate. În numărul viitor vom discuta despre câteva programe (utilitare standard) care folosesc în mod curent expresii regulate - cum ar fi `egrep` sau `sed`.

Autor:

radu.mihailescu@linux360.ro

HTML - Legături

Bidea Cristian

În numerele trecute am tot vorbit despre faptul că HTML cuprinde cam toate opțiunile introduse de un document obişnuit dar are anumite caracteristici care îl diferențiază net de acestea. Exact la o astfel de caracteristică mă voi referi în continuare. *Link-urile*, *hyperlink-urile* sau *Web link-urile* sunt ceea ce au făcut HTML atât de popular.

Prin link vom înțelege o legătură între două obiecte Web (imagini, documente HTML, sunete, secvențe video). Un link are două terminații, numite ancore, şi o direcție. Link-ul începe la ancora "sursă" şi indică spre ancora "destinație" care, aşa cum am spus, poate fi orice obiect Web.

```
...
<a href = "index.html">Acasa</a>
...
```

În exemplul anterior am văzut cum poate fi declarat un link. Acum să vedem cum acesta este interpretat vizual. În browser nu va apărea decât cuvântul "Acasă" iar dacă vă veți duce cu pointer-ul mouse-ului pe acesta, veți observa cum pointerul se schimbă (în cele mai multe cazuri aşa se întâmplă). Aceasta ne indică faptul că "Acasă" este un link. În momentul când apăsăm pe el browser-ul va încerca să încarce obiectul specificat de atributul *href*, în cazul nostru documentul HTML "index.html".

Link-ul mai are o caracteristică foarte interesantă: poate indica spre o anumită parte din documentul care îl conține. Ce înseamnă asta? Dacă spre exemplu într-un document am mai multe capitole şi documentul este relativ mare atunci pot pune la început link-uri cu numele capitolelor care să indice spre acestea. În momentul când link-urile sunt apăsate browserul nu va încărca un obiect nou, ci va aduce în "prim plan" partea din document spre care indică link-ul. Pentru a realiza acest lucru atributul *href* va lua valoarea unuia dintre atributele *id* sau *name* a paragrafului spre care se doreşte să se indice, prefixat de caracterul "#".

```
...
<a href = "#cap1" title = "Despre istoria Linux">Capitolul 1</a>
<a href = "#cap2" title = "Despre chroot jails">Capitolul 2</a>
...
<a name = "cap1">
Continutul capitolului unu vine aici.
</a>
<a name = "cap2">
Continutul capitolului doi vine aici.
</a>
...
```

```
...
<a href = "#cap1" title = "Despre istoria Linux">Capitolul 1</a>
<a href = "#cap2" title = "Despre chroot jails">Capitolul 2</a>
...
<a id = "cap1">
Continutul capitolului unu vine aici.
</a>
<a id = "cap2">
Continutul capitolului doi vine aici.
</a>
...
```

Aşa cum puteți observa în exemplele anterioare, a mai apărut un atribut: *title.* Acesta va fi reprezentat de browser în momentul în care pointer-ul mouse-ului se află pe link-ul în cauză. Reprezentarea poate diferi de la browser la browser. Unele pot afişa valoarea lui *title* în bara de stare, altele vor afişa mici mesaje pop-up, etc.

Mai există o serie de atribute care sunt folosite pentru a informa browserul cu privire la obiectul "destinație" care urmează a fi încărcat. Acestea sunt *hreflang* şi *charset*. Ambele se specifică numai dacă elementul *href* este specificat. *Hreflang* indică browserul-ului care este limbajul de bază a resursei indicate de link, iar *charset* indică ce tip de codare pentru caractere foloseşte resursa indicată de link.

```
...
<a href = "index.html" charset=UTF-8 hreflang=en>
<a href = "index.html" charset=EUC-JP hreflang=en-US>
...
```

Între link-uri pot exista anumite relații de ordine. Acestea pot fi specificate folosind atributele *rel* şi *rev*. De ce sunt importante aceste atribute? Dacă un anumit document HTML ocupă un loc anume în cadrul unei succesiuni de documente (de exemplu conține un capitol dintr-o carte) atunci link-urile spre celelalte documente HTML din succesiunea de documente pot arăta acest lucru prin intermediul atributelor prezentate anterior.

```
...
<a href="index.html" rel="Index">Coperta<a>
<a href="capitol6.html" rel="Prev">Capitolul 6<a>
<a href="capitol8.html" rel="Next">Capitolul 8<a>
...
```

Înafara link-urilor specificate de tagurile <A></A> mai există link-uri mai speciale pe care le-am folosit în numerele trecute, şi anume cele define de tagul <LINK>. Am folosit aceste link-uri în momentul în care am folosit stiluri specificate în documente externe. Veți vedea că aceste link-uri sunt foarte folositoare în foarte multe situații şi rezolvă o serie de probleme mai puțin cunoscute. Link-urile care folosesc tagul <LINK> sunt folosite în secțiunea de

header a documentului HTML.

```
...
<HEAD>
<LINK REL="Index" HREF="index.html">
<LINK REL="Prev" HREF="Capitolul6.html">
<LINK REL="Next" HREF="Capitolul8.html">
</HEAD>
...
```

Secvența anterioară nu este redată vizual de toate browser-ele dar o să vedem că deşi nu e mereu redată vizual poate avea o importanță aparte pentru motoarele de căutare. Motoarele de căutare au un modul, de obicei numit spider, care are o sarcină precisă: să caute şi să indexeze pagini web. Am mai prezentat metode de a transmite motoarelor de căutare, care indexează documentul nostru, informații cu privire la conținutul documentului. Folosind <LINK> putem transmite şi alte informații motoarelor de căutare.

```
...
<LINK title="Index site în română" type="text/html" rel="alternate" hreflang="ro" href="http://www.undeva.ro/roversion.html">
<LINK title="Index site în franceză" type="text/html" rel="alternate" hreflang="ro" href="http://www.undeva.ro/frversion.html">
...
```

Deci, folosind <LINK>, putem spune motoarelor de căutare unde se află versiuni în alte limbi, ale documentului indexat. Vom vedea în continuare că putem spune motoarelor de căutare unde să găsească versiunea pentru imprimantă a documentului sau care este pagina de start a unei colecții de documente care sunt indexate de "păianjenii" motoarelor de căutare.

```
...
<LINK media="print" title="Documentul în PostScript" rel="alternate" type="application/postscript" href="http://www.undeva.ro/prversion.html">
<LINK rel="Start" title="Prima pagina a sitului" type="text/html" href="http://www.undeva.ro/index.html">
...
```

Link-urile au o construcție puternic generalizată, datorită în primul rând obiectelor Web care pot fi ancoră sursă sau destinație, de aceea există destul de multe construcții ilegale pe care trebuie să încercați să le evitați.

Valorea atributelor *name* sau *id,* care apar ca parametri pentru tagul <A>, este "case-insensitive". Acest lucru înseamnă că dacă avem două construcții cu tagul <A> în care *name* ia valoarea "abc" în prima construcție şi valoarea "AbC" în cea de-a doua construcție, atunci valorile parametrului *name* în cele două construcții sunt egale. HTML impune o condiție de unicitate asupra valorilor parametrilor *name* şi *id,* construcțiile acestea devenind astfel, interzise. Unicitatea impusă nu este doar la nivel local (cu efect doar asupra unei clase restrânse de taguri), ci la nivel global. Astfel că, regula nu se schimba cu nimic, dacă considerăm oricare două taguri ale căror atribute *name* au aceeaşi valoare.

Altă construcție interzisă de HTML este aceea în care tagurile <A></A> formatează o altă construcție formată cu ajutorul tagurilor <A></A>, ca în exemplul următor.

```
...
<A href="index.html"><A href = "book.html"></A></A>
...
```

Ne-am obişnuit ca după prezentarea sintaxei fiecarui element să ne ocupăm de modificarea atributelor vizuale. Şi în acest caz vom respecta acest obicei. La link-uri situația este un pic diferită pentru că un link are mai multe stări (vizitat, nevizitat, etc). Voi prezenta un exemplu şi apoi voi comenta pe marginea acestuia.

```
... <HEAD>
<STYLE type="text/css">
a.lk:active {font-family: Arial, Helvetica, sans-serif; font-size: 10px; color: #0099FF; text-decoration: underline; cursor: hand; ; background-color: #CCCCCC}
a.lk:hover {font-family: Arial, Helvetica, sans-serif; font-size: 10px; color: #9900FF; text-decoration: underline; cursor: hand; ; background-color: #CCCCCC}
a.lk:link {font-family: Arial, Helvetica, sans-serif; font-size: 10px; color: #0099FF; text-decoration: underline; cursor: hand; ; background-color: #CCCCCC}
a.lk:visited{background-color:#11AA00; text-decoration:none;}
</STYLE>
</HEAD>
...
<A class="lk" href="index.html">Inapoi</A>
...
```

Am încercat să modificăm atributele tagurilor <A> şi am folosit o construcție mai specială de genul: tag.clasa:selector. Selectorul este un termen specific stilurilor şi este folosit pentru a modifica numai acele taguri care îndeplinesc anumite condiții specificate de selector. De exemplu *link* şi *visited* selectează acele link-uri care n-au fost încă vizitate şi respectiv cele care au fost vizitate. *Hover* şi *active* "intră în acțiune" atunci când utilizatorul execută o anumită acțiune (trece cu mouse-ul peste element, apasă butonul mouse-ului în timp ce se află pe element, etc).

Link-urile pot fi specificate şi prin intermediul hărților de imagini. Hărțile de imagini permit autorilor să definească anumite regiuni dintr-o imagine şi să asocieze acestor regiuni anumite acțiuni. Acțiunile vor fi definite folosind lucrurile învățate la link-uri.

Hărțile de imagini sunt de două tipuri:

- Hărțile de imagini interpretate la client (client-side image maps)
- Hărțile de imagini interpretate la server (server-side image maps)

Pentru a defini o hartă care să fie interpretată de browser (client) trebuie să urmăm etapele următoare:

1. Definim imaginea folosind tagurile <IMG> sau <OBJECT> şi folosim atributul *usemap* pentru a folosi o hartă definită ulterior;
2. Definim harta folosind tagul <MAP>.

```
...
<OBJECT data="meniu.png" type="image/png" usemap="#map_meniu">
<MAP name="map_meniu"> <A href="index.html" shape="rect" coords="0,0,10,10">Index</A>
<A href="produse.html" shape="rect" coords="10,0,20,10">Produse</A>
<A href="concurs.html" shape="poly" coords="0,0,10,10,10,0,0,0">Concurs</A>
<A href="guest.html" shape="circle" coords="40,10,5">Carte de oaspeti</A>
</MAP>
...
```

Putem observa că în tagurile <A> au apărut două atribute noi: *shape* şi *coords*. Următorul tabel arată ce valori poate lua *shape* şi ce coordonate trebuiesc specificate pentru fiecare valoare în parte.

Valoare	Coordonate
default	Toată regiunea
rect	x1,y1,x2,y2
circle	x,y,r
poly	x,y,x1,y1,...,xn,yn,x,y

În concluzie există trei posibilități pentru *shape*. Pentru dreptunghi (rect) se specifică coordonatele colțului stanga-sus şi colțului dreapta-jos, pentru cerc (circle) se specifică coordonatele centrului şi raza, iar pentru poligon (poly) sunt specificate toate vârfurile poligonului în care primul vârf specificat trebuie să coincidă cu ultimul.

Într-o hartă acțiunile pot fi specificate, aşa cum am văzut, folosind tagurile <A></A> dar vom vedea că putem face acelaşi lucru folosind elementul AREA.

```
...
<OBJECT data="meniu.png" type="image/png" usemap="map_meniu">
</OBJECT>
<MAP name="map_meniu">
<AREA href="capitolul1.html"
      alt="Capitolul 1"
      shape="rect"
      coords="0,0,118,28">
<AREA href="capitolul2.html"
      alt="Capitolul 2"
      shape="rect"
      coords="184,0,276,28">
<AREA href="capitolul3.html"
      alt="Capitolul 3"
      shape="circle"
      coords="184,200,60">
<AREA href="capitolul4.html"
      alt="Capitolul 4"
      shape="poly"
      coords="276,0,276,28,100,
      200,50,50,276,0">
</MAP>
...
```

Am văzut cum putem defini o imagine ca fiind hartă, folosind tagul <OBJECT>. Să vedem şi un exemplu care foloseşte tagul <IMG>.

```
...
<img src="meniu.png" usemap="#map_meniu">
...
```

Este mai avantajos în multe situații să folosim tagul <OBJECT> şi să vedem o astfel de situație.

```
...
<OBJECT data="meniu.png" usemap="#map_meniu">
<OBJECT data="meniu.gif" usemap="#map_meniu">
<MAP name="map_meniu">
...
</MAP>
</OBJECT>
</OBJECT>
...
```

Construcția anterioară poate fi folosită pentru a ne asigura că browser-ul va afişa un format de imagine. De exemplu, dacă browser-ul nu recunoaşte formatul PNG va încerca să citească imaginea în format GIF.

Hărțile de imagini introduc o flexibilitate crescută pentru designeri, pentru că aceştia pot creea imaginea şi abia apoi vor defini regiunile care au asociate acțiuni. Există totuşi multe argumente împotriva folosirii hărților de imagini. Cel mai important este acela că imaginile pot ocupa foarte mult şi folosirea unei singure hărți poate creşte considerabil timpul de descărcare al documentului HTML care conține harta în cauză. Apoi este destul de greu să gândim tot site-ul ca o imagine şi să avem pretenția de a schimba conținutul site-ului uşor. De aceea hărțile de imagini trebuiesc folosite cu grijă şi numai acolo unde nu se poate obține, ceea ce se doreşte, pe o altă cale.

Am ajuns şi la sfârşitul expunerii noastre. Trebuie să spun că odată cu acest articol am reuşit să acopăr elementele de bază pentru limbajul HTML. În numerele viitoare vom vorbi despre elemente care ajută extraordinar de mult în realizarea unui design profesional al site-urilor WEB (tabele, layer-e, etc).

Deasemeni, tot din numărul viitor vom inaugura o serie dedicată exclusiv design-ului care se va baza foarte mult pe lucrurile prezentate deja în seria despre HTML. Am considerat necesar introducerea acestei serii pentru a putea păstra aceeaşi linie de prezentare în seria dedicată limbajului de marcare HTML şi pentru a reuşi prezentarea HTML într-un context mai practic, care în opinia mea este un lucru binevenit.

Resurse:

- www.w3.org

Autor:

cristian.bidea@linux360.ro

Interoperabilitate Samba - Windows 2003 Server

Radu Popa

Acest tutorial vă va purta, pas cu pas, prin integrarea unui server de fişiere şi tiparire Samba într-un domeniu controlat de un Windows 2003 Server.

Lansarea lui Windows 2003 în aprilie anul trecut, a făcut administratorii de reţele să se întrebe dacă sunt gata sau nu să treacă de la domeniile controlate de Windows NT şi 2000 la cele conduse de cel mai discutat sistem de operare al companiei Microsoft.

De cealaltă parte, "pinguinul" făcea şi el senzaţii. Câteva organizaţii începuseră deja să implementeze încet, încet Linux-ul, iar altele doreau sa înlocuiască definitiv infrastructura Windows existentă cu una bazată pe Linux.

Întrebarea care se punea era dacă se poate integra Linux-ul, din perspectiva unui server de fişiere şi tiparire, într-un domeniu Windows 2003. Până la apariţia versiunii 3.x de la Samba (server de fişiere si tipărire pentru Linux si Unix), s-a dovedit revoluţionar facând acest lucru în domenii Windows NT si 2000. Dar, aşa cum este bine ştiut, Microsoft nu doreşte ca rivalul sau open-source sa fie atât de bine integrat cu produsele Windows.

Calculatoarele folosite în acest tutorial sunt următoarele:

Windows 2003 PDC - adresă IP: 192.168.0.1, hostname: w2k3
Numele complet al domeniului: TEST.LOCAL
Domeniu: TEST

Suse Linux Professional English 9.0 - adresă IP: 192.168.0.2, hostname: smb3

Pachete software necesare:

OpenLDAP development şi core.
Librăriile Kerberos 5, workstation şi development.
PAM core, development şi Kerberos 5 şi smb.

Pentru instalarea pachetelor in format RPM, folosim comanda:

```
rpm -ivh nume_pachet.rpm
```

Samba, ultima versiune (3.0.2a).

De asemenea avem nevoie de pachetele standard pentru compilare (gcc, make).

Dupa descărcare versiunii 3.0.2a de la Samba (www.samba.org), urmeaza dezarhivarea:

```
tar zxf /tmp/samba-3.0.2a
```

Apoi instalarea propriu-zisă:

```
cd samba-3.0.2a/source
./configure --prefix=/opt/samba --with-winbind --with-pam-winbind --with-smbmount; make; make install

cd /etc
```

În cele ce urmează, vrem să înlocuim în krb5.conf, PREFIX_DOMAIN.SUFFIX_DOMAIN şi prefix_domain.suffix_domain cu valorile corespunzătoare pentru domeniul nostru (TEST.LOCAL respectiv, test.local).

Procedăm la fel şi cu /opt/samba/lib/smb.conf.

Vom crea apoi directoarele necesare pentru file sharing:

```
mkdir /opt/samba/netlogon
mkdir /shared; chmod 777 /shared.
```

Fisierele de configurare ar trebui sa arate cam aşa:

```
/opt/samba/lib/smb.conf

#GLOBAL
realm = TEST.LOCAL
ads server = 192.168.0.1
security = ADS
encrypt passwords = yes
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
workgroup = PREFIX_DOMAIN
winbind uid = 10000-20000
winbind gid = 10000-20000

#FOR WINDOWS 9x
[NETLOGON]
path = /opt/samba/netlogon
read only = yes

#SAMBA SHARE
[SHARED]
path = /shared
read only = no
public = no
only guest = no
writable = yes

/etc/pam.d/samba:

#%PAM-1.0
auth required /lib/security/pam_winbind.so
auth required /lib/security/pam_pwdb.so
auth required pam_nologin.so
auth required pam_stack.so service=system-auth
account required /lib/security/pam_winbind.so
```

```
account required
/lib/security/pam_pwdb.so
account             required
pam_stack.so  service=system-
auth
session             required
pam_stack.so  service=system-
auth
password required
pam_stack.so service=
system-auth
```

```
/etc/pam.d/login:
```

```
auth required
pam_securetty.so
auth sufficient
/lib/security/pam_winbind.so
auth sufficient
/lib/security/pam_unix.so
use_first_pass
auth required
pam_stack.so  service=system-
auth
auth required
pam_nologin.so
account sufficient
>/lib/security
/pam_winbind.so
account required
pam_stack.so  service=system-
auth
password required
pam_stack.so  service=system-
auth
session required
pam_stack.so  service=system-
auth
session optional
pam_console.so
```

Restartăm xinetd:

```
/etc/init.d/xinetd restart
```

Apoi creăm un nou fişier in `/etc/init.d` pe care îl numim `samba_winbind`. Acesta este necesar pentru a activa serviciile Samba odată cu pornirea sistemului. În acest fişier scriem următoarele linii:

```
/opt/samba/sbin/smbd -D
/opt/samba/sbin/nmbd -D
/opt/samba/sbin/winbindd
```

Apoi setăm permisiunile asupra fişierului:

```
chmod
755 /etc/init.d/samba_winbind
```

Ne asigurăm ca serviciile Samba sunt pornite in runlevel-urile 3 şi 5:

```
cd /etc/init.d/rc3.d
sn -s
/etc/init.d/samba_winbind
S99samba_winbind
S99samba_winbind
```

```
cd /etcinit.d/rc5.d
sn -s
/etc/init.d/samba_winbind
S99samba_winbind
S99samba_winbind
```

Acum modificăm configuraţia şi e PDC-ul care rulează 2003 Server:*Start->settings->control panel->administrative tools->domain controller policies*. Selectăm local policies în partea stângă a ferestrei, apoi modificăm la opţiunea "*Microsoft Network Server: Digitally sign communications*" always cu disable.

Pornim serviciile Samba:

```
/etc/init.d/samba_winbind
```

După care ne logăm în domeniu cu contul de administrator:

```
/usr/kerberos/bin/kinit
administrator@TEST.LOCAL
```

Introduceţi parola de administrator când vi se cere acest lucru. De asemenea fiţi siguri că ceasurile celor două calculatoare sunt sincronizate între ele. În caz contrar veţi primi un mesaj de eroare de genul: "`kinit(v5): Clock skew too great while getting initial credentials`".

Apoi putem adăuga serverul Samba la domeniul Windows 2003 cu comanda:

```
/opt/samba/bin/net ads join
```

Veţi primi un mesaj prin care sunteţi informat că serverul Samba a fost adăugat cu succes la domeniul Windows 2003.

Vom testa dacă ne putem lega la un share administrativ de pe PDC:

```
/opt/samba/bin/smbclient
//w2k3/c$ -k
```

Introduceţi parola dacă vi se cere acest lucru. Nu ar trebui, întrucât sunteţi deja logat ca administrator. In mod normal ar trebui să vă fie afişat un promt `smbclient (smb: >)` şi puteţi sa navigaţi prin `c$`. Pentru a ieşi tastaţi "`exit`".

Acum serverul Samba a fost adaugat cu succes la domeniul controlat de Windows 2003 server şi puteţi accesa resursele partajate din reţea. Acelaşi lucru poate fi făcut şi de pe orice staţie windows din reţea. Utilizatorii se pot lega la resusele de pe serverul Samba cu propriile lor conturi din domeniu, lucru realizat prin intermediul `winbind`.

Singura caracteristică de securitate dezactivată la controllerul de domeniu a fost procedura implicită de negociere a conturilor. La Windows 2000 si NT aceasta era implicit dezactivată.

Autor:

radu.popa@linux360.ro

Ripping şi encoding: de la CD-DA la MP3

Radu - Eosif Mihăilescu

În acest articol am onoarea să vă prezint o metodă (din cele câteva existente) de a transcoda conținutul unui CD audio (CD-DA) de la clasicul format de modulație în cod de impulsuri (PCM) cu rata de eşantionare de 44 kHz şi lățimea eşantionului de 16 biți şi două canale; la unul din două formate moderne, folosite pe larg la stocarea informației audio (muzică).

Aceste două formate amintite sunt MPEG1 Layer III (cunoscut ca MP3) şi Ogg Vorbis (cunoscut ca OGG). După cum poate ştiți, OGG a devenit popular pentru ca nu este grevat de nici un patent, iar pe de altă parte, MP3 este grevat de diferite patente şi/sau restricții de utilizare/implementare ce țin de proprietatea intelectuală. Acest gen de detalii (deşi foarte importante având în vedere că țin de dreptul legal de a utiliza sau nu un program sau o tehnologie) nu fac obiectul prezentului articol.

Metoda descrisă în continuare poate părea spartană (nu seamănă nici pe departe cu, de exemplu, AudioCatalyst de pe Windows) dar este cel puțin la fel de eficientă în sensul că, la final, produce acelaşi set de date (fişierele .MP3).

Ingrediente

În cele ce urmează ne vom folosi de un script dedicat acestui scop, pe numele său "`cdmp3`". Acest script bash, scris de Roland Riegel poate fi obținut de la adresa http://www.roland-riegel.de/cdmp3/index_en.html. Acest script are rolul de a automatiza procesul de extracție a fluxului de date de pe CD-ul audio, obținerea (în cazul în care acest lucru este posibil) denumirii pistei curente şi, în fine, transcodarea acesteia în formatul dorit (MP3 sau OGG). În cazul în care informațiile de nomenclatură pot fi obținute (prin CDDB), cdmp3 va denumi fişierele rezultat folosind această informație şi, de asemenea, va stoca aceste informații în interiorul fişierelor rezultante folosind sistemele proprii fiecărui format de a stoca metainformație (e.g. ID3 pentru MP3).

Am spus că acest script nu face decât sa coordoneze procesul, deci, iată de ce mai avem nevoie (informații menționate şi în pagina dată):

- o modalitate de a extrage fluxul de date de pe CD-ul audio. Avem două opțiuni: `cdparanoia` sau `cdda2wav`
- o modalitate de face transcodarea. Şi aici avem două opțiuni: `lame` pentru MP3 şi `vorbis-tools` pentru OGG
- o modalitate de a obține informația de nomenclatură (numele melodiilor, autorilor, albumelor etc.). Aici avem o singură variantă, un modul Perl pe nume `CDDB_get`.

Pregatirea componentelor

Să spunem câteva cuvinte despre componente. Obținerea lor este foarte facilă, pagina de web a lui `cdmp3` furnizând toate informațiile necesare.

`cdparanoia`, `vorbis-tools`, `libao`, `libogg` şi `libvorbis` sunt componente foarte populare, nu sunt grevate de patente sau drepturi de proprietate intelectuala şi este deci foarte posibil ca ele să fie deja incluse în distribuția voastră - chiar dacă nu sunt instalate încă. De aceea, este recomandabil să consultați lista de pachete a distribuției înainte de a încerca să luați versiunile de pe Internet şi să le instalați pe sistem.

`lame` este, dimpotrivă, cel mai puțin probabil să-l găsiți în distribuția voastră aşa că va trebui să-l luați de pe Internet şi să-l instalați. Configurarea surselor, compilarea şi instalarea nu pun probleme, `lame` fiind un program "cuminte" din acest punct de vedere (putem aplica metoda "`tar zxvf lame-<versiune>.tar.gz; cd lame-<versiune>; ./configure; make; make install`" fără probleme).

`CDDB_get` se instalează ca oricare alt modul Perl "`tar zxvf CDDB_get-<versiune>.tar.gz; cd CDDB_get-<versiune>; perl Makefile.PL;make; make test; make install`".

Dacă aveți infrastructura CPAN configurată şi funcțională în instalarea voastră de Perl, puteți folosi clasica invocație`perl -MCPAN -eshell;` şi apoi `install CDDB_get` pentru a descărca şi apoi instala pachetul.

La treabă!

Sintaxa de apel a lui cdmp3 este destul de simplă: apelat fără parametrii se apucă singur să proceseze CD-ul introdus în prima unitate de CD, pistă după pistă, transcodând în MP3 cu o lățime de bandă de 192 kbps. Apelat aşa, cdmp3 nu scrie metainformație in fişierele MP3 rezultante, deci (şi poate şi pentru alte motive) vom dori să-l apelăm cu parametrii.

Un apel de forma "`cdmp3 --buffer --write-meta --write-m3u`" este arhisuficient pentru a duce treaba la bun sfârşit. Pentru OGG putem adăuga "`--ogg`" sau ne putem adresa cu "`cdogg`" scriptului.

Autor:

radu.mihailescu@linux360.ro

Best Common Practice

Radu - Eosif Mihăilescu

Vom continua această serie cu BCP-uri privind aspecte ale funcționării semi-automate, "din culise" a sistemului Linux în posesia căruia ne aflăm.

BCP numărul şaisprezece: *atunci când ceva se cere făcut de mai mult de două ori, în maniera identică şi la intervale de timp fixe este deja treaba lui* **cron**. Nu vă apucați să vă faceți bilețele (precum cele de cumpărături) cu ce operații periodice trebuie rulate când şi, pe parcursul zilei să le urmați şi să bifați una câte una operațiile de pe bilet. Nu vă apucați să scrieți script-uri care să compare ora exactă cu un interval predefinit şi să execute diferite operații la coincidență. Pentru toate acestea există o întreagă infrastructură, tot moştenire de la UNIX şi anume `cron`.

Acesta este un daemon, un proces server, care rulează în fundal şi execută acțiuni la intervale configurate de timp. Are fişier de configurare, are implementat pe majoritatea distribuțiilor un mecanism facil pentru execuția la intervale "standard" (o oră, o zi, o săptămână, o lună) şi are până şi un sistem integrat de sub-configurare prin care fiecare utilizator poate defini propriile sale intervale orare şi acțiunile de executat. Aşa că, folosiți-l cu încredere, cron nu uita să-şi facă treaba.

Să vorbim acum de câteva mecanisme care depind sau se folosesc de cron.

Cu toții ştiți ce este acela un jurnal (log). Este un fişier text în care se află înregistrări, câte una pe linie, ce descriu activitatea unui program, proces sau infrastructuri. Ca şi un jurnal din lumea reală, aceste fişiere cresc în continuu - ritmul diferă de la caz la caz, dar de crescut vor creşte mereu. Se pune întrebarea cum procedăm spre a nu epuiza spațiul de stocare existent în sistem?

Ei bine, iată BCP numărul şaptesprezece: *dați Cezarului ce este al Cezarului - log-urile sunt ale lui* **logrotate***, lăsați-l să se ocupe de ele*.

`logrotate` este un program mic dar eficient. El este rulat de `cron`, uzual în fiecare zi. Logrotate are o infrastructură de configurare bazată pe un fişier de configurare central (ce defineşte valorile implicite) şi un director cu fişiere de configurare corespunzătoare fiecărui program care produce log-uri. El are sarcina de a executa *rotația jurnalelor*, adică oprirea (sau semnalizarea, în unele cazuri) programului care scrie în jurnal, redenumirea fişierului jurnal curent sub un alt nume, recrearea unui fişier jurnal gol cu numele inițial şi repornirea programului. După acest pas, rămâne la decizia utilizatorului (exprimată prin configurare) dacă fişierele vechi se păstrează (şi dacă da, câte astfel de fişiere se păstrează), dacă se comprimă, daca se trimit prin poştă electronică şi tot aşa.

Folosind pe `logrotate` nu o să avem surpriza de a rămâne fără spațiu util de stocare din cauza jurnalelor care l-au ocupat - situație care, în ciuda aspectului comic, este din păcate destul de întâlnită.

Şi, că veni vorba de spațiu, să vorbim şi de "salubrizare". BCP de-al optsprezecelea: *sistemele Linux vin cu "gunoier" în dotarea standard, trebuie doar să-i spunem unde să caute*. Da, este vorba de utilitarul `tmpwatch`. Acest utilitar, inițial scris de RedHat pentru distribuția proprie (deşi el se găseşte liber şi poate fi instalat pe orice distribuție care nu ar veni cu el), implementează funcționalitatea de a căuta recursiv prin directoarele date ca argument fişierele care nu au mai fost accesate (atenție, nu *modificate!*) de un număr dat de ore (acesta din urmă transmis tot ca argument).

Odată identificate astfel de fişiere, ele sunt şterse de utilitare - rezultă astfel că avem la dispoziție un mecanism foarte puternic pentru a "face curat" în mod automat în locurile unde se adună fişiere temporare.

Să vorbim acum de două mecanisme - unul de "documentare" şi altul de "indexare" care se alfă în strânsă legătura cu `cron`.

BCP numărul nouăsprezece: *atunci când nu ştiți prea bine ce doriți a căuta prin documentația sistemului (documentație în sens UNIX, adică pagini de manual (man pages)), folosiți cu încredere pe* **whatis** *si* **apropos**. `whatis` şi `apropos` lucrează pe un index construit de `makewhatis` ce este chemat la rândul său de `cron`. Primul utilitar afişează titlurile paginilor (şi secțiunea unde se află) de manual care corespund exact numelui dat ca argument. Cel de al doilea afişează toate paginile care conțin în denumire (subiect) numele dat ca argument.

Cu speranța utilității şi urări de succes la înțelegerea şi folosirea Linux-ului.

Autor:

radu.mihailescu@linux360.ro

Migrare: alternative software

Ioana-Rebeca Gliția

Pe când am instalat pentru prima oară Linux-ul cu fratele meu lângă mine, începusem să fiu eu cea care "dădăceşte" computerul familiei. Planul era să mutăm toate activitățile pe Linux iar schimbarea să fie cât mai puțin dureroasă. De fapt, fratele meu m-a convins şi tot el a fost cel care a trebuit să primească reclamațiile. Părinții se plângeau mie iar eu duceam reclamația la următorul nivel: fratele mai mare "în Windows puteam face aia. Aici nu pot! Mai bine era în Windows!". Probabil că ceva asemănător e nevoit să audă în primele zile orice entuziast de Linux care îşi convinge apropriații să migreze de la Windows la Linux.

Din fericire, în ultima vreme este din ce în ce mai uşoară trecerea, mai ales dacă ştii unde să cauți. Scopul următoarelor episoade din seria de migrare este de a demonstra începătorilor sau viitorilor începători că da, există alternative.

Internet

Prima problemă care trebuia rezolvată rapid, pentru a nu primi reclamații a fost clientul de e-mail. Ai mei foloseau MS Outlook. Cel mai simplu le-a fost cu Kmail, dacă erau în KDE -poate şi pentru că pe acesta l-am găsit primul. S-au obişnuit atât de bine încât când au mai avut ocazia să folosească MS Outlook au fost nemulțumiți: "mai bine era cu celălalt program de mail". În KDE o altă alternativă bună ar fi Aethera. Ar mai fi Kiltdown, Liamail (suportă doar pop3 şi smtp) sau Empath dar acestea par a fi proiecte moarte.

În Gnome, cel mai cunoscut client de mail e Evolution. E stabil, lista de facilități e mare şi jos pălăria la aspect... Tot cunoscute în Gnome sunt Balsa, Spruce şi Sylpheed. Ar mai fi şi Mahogamy (rulează pe mai multe platforme).

Alți clienți de mail pentru X sunt XPine, XFMail sau TkMail.

Ce vine de la Mozilla...

Thunderbird. Nu e prea mult de spus despre el întrucât îi lipsesc foarte puține. A adus un nou aspect în lumea clienților de mail.

Bineînțeles că mai rămân clienții de mail de la browser-ele Mozilla şi Netscape. Şi cu aceştia tranziția este uşoară, dacă îi folosiți.

Pentru browsere... e foarte simplu. Lumea OpenSource pune la dispoziție o varietate de browsere pentru Linux (Mozilla, Konqueror, Galeon, Epiphany, Netscape etc).

Dacă sunteți dintre cei muncitori, pentru editare HTML găsim cu uşurință Quanta Plus sau Bluefish.

IRC-ul şi programele de Instant Messaging sunt şi ele foarte bine acoperite: Kvirc, KSirc, XChat, clienții de Jabber (http://www.jabber.org/software/clients), clienții multi-funcționali (ii puteți folosi simultan pentru AIM, ICQ, MSN, Yahoo, IRC, Jabber ş.a.: Gabber, Gaim, Kopete etc.) sau clienții pentru Linux veniți de la clasicii Yahoo sau AOL.

Multimedia

Acum e acum. Fără muzică nu se poate şi nici fără filme. Să nu uităm că geamăna mai vrea să şi modifice unele melodii pe calculator.

Astfel printre cele mai populare playere de filme în Linux sunt Mplayer, Xine şi aviplayer. Se pare însă că bătălia se dă între Mplayer şi Xine. Cât despre programe pentru muzică, diversitatea este şi mai mare. Avem XMMS, Zinf, noatun, Kaboodle, Snackamp, chiar Winamp în Linux pentru mp3. Veți fi surpinsi de stabilitatea XMMS-ului si varietatea opțiunilor precum şi a funcționalităților disponibile separat pentru el. Pentru CD-uri audio, în Linux avem la dispoziție programe ca şi KsCD, xmcd ş.a. Ajunge un search pe www.google.ro/linux şi veți primi rezultatele.

Pentru lucrul cu partiturile, veți găsi de asemenea soluții software cum ar fi NoteEdit în timp ce, dacă aveți surori ca a mea, le puteți mulțumi cu MusE.

Office

Aici poate fi puțin mai dificil. Deocamdată, inport-ul fişierelor tipice MS Office-ului nu este complet, dar suite ca OpenOffice sau KOffice îşi fac bine slujba la care au fost chemate. Pentru fişierele .doc o alternativă bună este de asemenea AbiWord.

Despre alternativele în grafică, editare de texte, scriere de CD-uri ş.a. vom discuta în episoadele următoare.

Resurse

- http://www.linuxlinks.com/Software/

Autor:

ioana.glitia@linux360.ro

Tips & tricks

Pentru a monitoriza resursele folosite de diferite aplicatii folositi comanda: `#top`, iar pentru a monitoriza resursele folosite de catre o anume aplicatie utilizati comenzile:`#TPID=`pidof <nume aplicatie>`; top `for i in ${TPID}; do echo -p $i; done`` sau `#top -p <pid>`, unde <pid> reprezinta PID-ul respectivei aplicatii.

Pentru a rula imediat dupa boot-are, la fiecare pornire a calculatorului, o aplicatie va trebui sa adaugati in fisierul `rc.local` linia de comanda folosita pentru rularea programului respectiv. Acest lucru se poate face fie prin editarea fisierului `rc.local` cu un editor text fie folosind comanda `echo`.

Pentru a adauga rapid linii text la sfârsitul unui fisier utilizati comanda: `#echo "text text text" >> /cale/catre/fisier`. ATENTIE: trebuie neaparat sa folositi ">>" pentru ca scriind doar ">" continutul fisierului respectiv va fi sters iar apoi va fi scrisa linia "text text text".

Pentru conversia rapida a unui sistem de fisiere ext2 in ext3 folositi comanda: `#tune2fs -j /dev/hdX` , unde hdX poate fi hda1, hda2, hdb3, in functie de partitia pe care doriti s-o transformati.

Pentru a copia, muta fisiere mari mai rapid puteti folosi comanda: `#cat /cale/fisier1 > /alta/cale/fisier1`. Astfel puteti sa scrieti imagini de dischete de boot (vezi linux pe o singura discheta) folosind: `#cat /cale/imagine > /dev/fd0`.

Pentru a monitoriza traficul de date facut de calculatorul dvs. folositi comanda: `#iptraf -i ethX`, unde ethX este interfata pe care vreti sa o monitorizati.

Folositi comenzile: `#eject` si `#eject -t` pentru a deschide si închide cdromul. Comanda `#eject` este foarte utila pentru ca scuteste utilizatorul de folosirea comenzii `#umount /dev/cdrom`, înainte de scoaterea manuala a CD-ului din unitate.

Pentru a vedea spatiul folosit si cel liber de pe o partitie montata folositi comanda: `#df -h` . Datorita parametrului "-h" marimile respective sunt afisate în Gb si Mb, fiind astfel mai usor de citit.

Glosar comenzi

`arch`

Afişează tipul de arhitectură al maşinii.

`cut [listă de caractere] [opţiuni] [fişier...]`

Reţine bucăţi/cămpuri din liniile unui fişier text. Cut se foloseşte pentru a afişa, de exemplu, doar primele cinci caractere ale fiecărei linii sau câmpurile doi până la şase dintr-o astfel de linie.

`dd [opţiuni]`

`dd` transferă un fişier (de la intrarea standard la ieşirea standard, in mod implicit) folosind blocuri de o mărime aleasa arbitrar de utilizator si efectuând conversii opţionale pe fluxul de date transferat.

`dmesg`

Afişează sau controlează tamponul circular de mesaje de jurnal al nucleului. Programul ajută utilzatorii să listeze mesajele afişate in timpul procesului de bootare. Ei pot folosi astfel comanda `dmesg > boot.messages`

`gzip [opţiuni] [nume]`
`gunzip [opţiuni] [nume]`
`zcat [opţiuni] [nume ]`

`gzip` reduce dimensiunea fişierului specificat folosind un tip de compresie numit Lempel-Ziv (LZ77). De câte ori este posibil, fiecare fişier este înlocuit cu un fişier nou cu extensia .gz, pastrând aceleaşi permisiuni de acces şi data ultimei modificări.

Fişierele comprimate pot fi aduse la forma iniţială folosind `gzip -d`, `gunzip` sau `zcat`

`gunzip` înlocuieşte fiecare fişier cu extensia .gz, -gz, .z, -z, z, sau .Z dintr-o listă de fişiere cu un fişier decomprimat fără extensia originală. `gunzip` recunoaşte de asemenea extensiile speciale .tgz si .taz ca fiind prescurtări ale .tar.gz, .tar.z si respectiv .tar.Z

`zcat` poate decomprima fie o listă de fişiere ale căror nume au fost date ca parametrii pe linia de comandă, fie intrarea standard. După realitarea decomprimării, rezultatul va fi scris pe iesirea standard. `zcat` va decomprima fişierele care posedă semnatura corecta in antet (GZip) indiferent daca numele lor se termina in .gz sau nu.

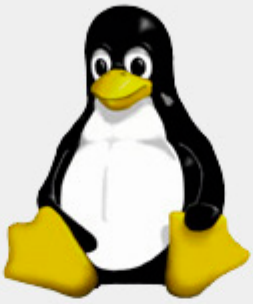
`hostname [opţiuni] nume`

Această comandă afişează sau setează numele nodului. Numele este utlizat de mai multe programe pentru a identifica maşina. Singurul tip de utilizator care poate schimba acest hostname este superuser-ul.

linux360

Ovidiu Lixandru - director general
Răzvan Şocu - director general
Radu Eosif Mihăilescu - redactor şef
Daniel Secăreanu - redactor
Radu Popa - redactor
Ioana Rebeca Gliția - redactor
Andrei Ciubotică - redactor
Cristian Bidea - redactor

Cosmin Staicu - tehnoredactor
Ion Mudreac - colaborator
Ciprian Negrilă - colaborator
Florin Vereş - colaborator
Răzvan Popa - colaborator
Silviu Foca - colaborator
Răzvan Corneliu Vilt - colaborator
Dan Marcu - colaborator



"Vă rog să mă credeți, nu caut să distrug Microsoft.
Acesta va fi doar un efect colateral complet neintenționat."

Linus Torvalds

linux360 - numărul 06 - februarie - martie 2004