

LA BIBBIA

Contenuti

Script di avvio del sistema e della bash	4
Apache + Tomcat + SSL	6
Ricompilare il kernel e i moduli	8
Riconfigurare e ricompilare il Kernel	8
Gestione dei moduli	9
Initrd	9
MySQL	11
Configurazione di rete e firewall	13
Configurazione tunnel Ipv6 con Iproute2	15
Configurazione di Xfree86	16
Masterizzare con un 2.4 e un masterizzatore ide	19
Leggere DVD video	21
Lettore CD pianta il sistema al mount	22
Il magico mondo della CLI	23
Ssh	23
Scp	24
Rsync	24
Trickle	25
Du, df e ls	26
Monitoraggio di sistema e Hardware	26
Screen	27
Find	27
Dig	28
Split	28
Tar, gz e bz2	29
Zip e Unzip	29
Rar e Unrar	30
Formattazione	30
Partizionare	30
Aumix	31
Su	31
md5sum	31
GnuPGP	31
Emacs	33
RTorrent	34
Sitecopy	34
Log di sistema	36
Cron	37
Atd	38
Dhcp	39
Xfree client-server scatenato	41

Autenticazione del client.....	41
Xnest.....	42
Xdmcp.....	42
Samba	44
Bind	46
Apache in configurazioni avanzate	50
Utilizzo con tomcat.....	51
Un solo server per più hosts.....	51
Un solo host con più server.....	52
Un host per più server replicati.....	52
Apache + Tomcat +SSL con directory con differenti diritti di accesso.....	52
Situazioni più complicate.....	54
Apache e .htaccess	55
Gestione di una Slackware	56
Bash scripting	61
CVS	66
Icone di Windows sotto Linux	69
Video Editing	70
I Tools.....	70
DVD Backup.....	76
Editing\Modifica video.....	76
File di Grandi Dimensioni.....	76
Jboss + Nukes	78
Patch	82
Patchare un file.....	82
Creare una patch.....	82
Abilitare il supporto SMP nel Kernel	83
Importare kernel da altre distribuzioni su Slackware	84
Modificare /etc/fstab	86
Modificare /etc/fstab.....	86
Slackware: il sistema di pacchetti	88
Creare un pacchetto.....	88
Un esempio di SlackBuild.....	90
Come distribuire i propri pacchetti.....	91
Il sistema d'installazione.....	94
Autofs e Automount	96
Ide e gestione di progetti	98
Java.....	98
QTDesigner e Kdevelop.....	98
Anjuta (o Kdevelop) e Glade.....	99
Postfix come relay con autenticazione TLS+SASL	101
Postfix come server di posta con autenticazione TLS+SASL	103
GNU Java Tools	105
Il suono con linux	107
Device rimovibili e Kernel 2.6	108
La nuova generazione del server X	110
Backup di Sistema	111
Virtualizzazione ed Emulazione	112
Postfix + Cyrus-Sasl come server di posta su Slackware con autenticazione	114

<u>ssh senza password</u>	115
<u>Spegnere la Slackware</u>	117
<u>Kppp da utente</u>	118
<u>Synergy</u>	119
<u>Salvare pagine web in firefox</u>	121
<u>Aggiungere il cestino sul desktop in Kubuntu</u>	122
<u>Alltray</u>	123
<u>Ubuntu: avvio in modalita' console</u>	124
<u>Vi col comportamento di VIM su Ubuntu</u>	125
<u>Risolvere i problemi di apt-get</u>	126
<u>Configurazione di Mldonkey</u>	127
<u>Dual Monitor con schede NVidia</u>	128
<u>Vim e spellchecking</u>	129
<u>Sudo e redirezione</u>	130
<u>Criptare partizioni</u>	131

Script di avvio del sistema e della bash

Caso BSD

/etc/inittab

contiene la configurazione del demone Init: per ogni evento è segnalato il file di script da mettere in esecuzione:

```
# These are the default runlevels in Slackware:
# 0 = halt -> rc.0
# 1 = single user mode -> rc.K
# 2 = unused (but configured the same as runlevel 3)
# 3 = multiuser mode (default Slackware runlevel) -> rc.M
# 4 = X11 with KDM/GDM/XDM (session managers) -> rc.4
# 5 = unused (but configured the same as runlevel 3)
# 6 = reboot -> rc.6
```

/etc/rc.d

è la cartella che contiene tutti i files di script da mettere in esecuzione precedentemente specificati. Tali file vengono eseguiti dall'Init, rc.M, rc.0, ecc. A loro volta controllano se gli altri files di script presenti in questa directory sono eseguibili e se si li eseguono uno alla volta per nome.

Quindi per mettere in esecuzione un nuovo file non basta porlo in questa directory e poi metterlo eseguibile, ma bisogna anche chiamarlo direttamente da uno di questi files (rc.M per l'avvio).

Caso SystemV

Nel caso il sistema sia un SysV l'organizzazione è differente: vi sono le seguenti directory:

```
/etc/rc.d/rc0.d
/etc/rc.d/rc1.d
/etc/rc.d/rc2.d
```

ecc. ognuna corrispondente ad un runlevel. All'occorrenza di ogni runlevel vengono messi in esecuzione TUTTI i files in quelle directory, indipendentemente dal nome.

Bash

All'avvio la bash esegue

```
/etc/profile
```

file in cui vi sono le configurazioni generali delle variabili d'ambiente, a sua volta questo file mette in esecuzione tutti gli script presenti in

```
/etc/profile.d
```

con le variabili d'ambiente di alcuni programmi in particolare.

Per variabili personali di un singolo utente e script d'avvio la bash all'avvio esegue:

```
~/.bash_profile
```

```
~/.bash_login  
~/.profile
```

Se la shell è una shell remota (invocata da rsh) allora esegue:

```
~/.bashrc
```

Apache + Tomcat + SSL

Installazione dell'Apache e del Tomcat

Scaricare il Tomcat (la versione non è importante) e semplicemente scompattarlo.

Riporto i passi per installare l'apache2 da sorgente, perchè l'Apache 1.x si trova già pacchettizzato per Slackware da anni.

```
./configure --prefix=/usr/local/apache --enable-so --enable-ssl
make
make install
```

Installazione e configurazione del connector

Scaricare l'rpm precompilato del mod_jk dal sito dell'Apache, e scompattarlo in una directory temporanea senza installarlo.

Tra i file estratti c'è mod_jk.so, copiarlo in /usr/local/apache/modules

Tra i files estratti c'è workers2.properties, copiarlo in /usr/local/apache/conf

Modificare workers2.properties aggiungendo una voce come la seguente:

```
[uri:/cmv/*]
info=The Tomcat /jkstatus handler
#group=status:
```

corrispondente alla propria applicazione web.

Tra i files estratti c'è jk2.properties, copiarlo in /usr/local/tomcat/conf

Configurazione del Tomcat

Basta editare /usr/local/tomcat/conf/server.xml e assicurarsi che il connector per il JK2 sulla porta 8009 non sia commentato:

```
Define a Coyote/JK2 AJP 1.3 Connector on port 8009
```

Configurazione dell'Apache

Editare /usr/local/apache/conf/httpd.conf:

aggiungere una direttiva ALIAS per la directory dell'applicazione sotto Tomcat, esempio:

```
Alias /cmv/ "/usr/local/tomcat/webapps/cmv/"
```

aggiungere anche i comandi per caricare il modulo JK2 e la sua configurazione:

```
LoadModule jk2_module modules/mod_jk2.so
JkSet config.file /usr/local/apache/conf/workers2.properties
```

Configurazione dell'SSL

In genere si fa gestire l'ssl da Apache, il quale comunica poi in chiaro, normalmente in modo

trasparente con Tomcat.

Andare in /usr/local/apache/conf ed eseguire i seguenti comandi:

```
openssl req -new -out server.csr
openssl rsa -in privkey.pem -out server.key
openssl x509 -in server.csr -out server.crt -req -signkey
server.key -days 365
openssl x509 -in server.crt -out server.der.crt -outform DER
```

creare le cartelle ssl.key e ssl.crt e spostare server.key in ssl.key e server.crt in ssl.crt.
Infine editare ssl.conf e commentare <IfDefine SSL> e </IfDefine SSL>.

Ricompilare il kernel e i moduli

Riconfigurare e ricompilare il Kernel

I comandi necessari sono i seguenti:

```
make menuconfig (oppure make xconfig)
make dep
make bzImage
make modules
make modules_install
```

a seguito di queste operazioni, l'immagine del kernel si trova in

```
{/usr/src/linux/}arch/i386/boot/
```

e i moduli vengono installati in

```
/lib/modules/{versione del kernel}/
```

Da notare che la configurazione del kernel è salvata nel file:

```
{/usr/src/linux/}.config
```

e che solitamente le distribuzioni ne salvano un link o una copia anche nella directory:

```
/boot
```

la configurazione di lilo, che permette di cambiare o aggiungere kernel all'avvio, è in /etc/lilo.conf; una volta modificata bisogna lanciare il comando lilo per rendere effettive le modifiche.

Per aggiungere un kernel basta aggiungere alla fine del file:

```
image = /boot/vmlinuz      [il path della nuova immagine]
root = /dev/hda7           [la partizione con linux]
label = Linux              [l'etichetta col nome in lilo]
append="hdd=ide-scsi"     [parametri da passare al kernel: questo è necessario per
                           masterizzare con un 2.4 col device /dev/hdd]

read-only
```

All'inizio del file vi sono le seguenti:

```
lba32
boot = /dev/hda           [l'hdd dove installate lilo]
prompt                      [se lilo chiede all'avvio cosa far partire oppure no]
timeout = 50               [attesa]
```

Gestione dei moduli

I comandi per la gestione dei moduli sono i seguenti:

```
insmod    [carica un modulo]
rmmod     [scarica un modulo]
```

lsmod [elenco dei moduli caricati]
depmod [elenco delle dipendenze di un modulo]
modprobe [carica un modulo tenendo conto della configurazione]
modinfo [informazioni su un modulo]

importante è capire bene cosa serve modprobe: insmod carica un modulo dal suo filename, mentre modprobe tiene conto di

/etc/modules.conf in caso di 2.4
/etc/modprobe.conf in caso di 2.6

nei quali sono specificati degli alias semplificati per i nomi dei moduli e loro eventuali dipendenze. Per esempio:

```
alias char-major-10-135 rtc
```

quindi con un semplice comando come “modprobe snd” carichi tutti i moduli relativi al suono in un colpo solo.

Solitamente i moduli che servono vengono caricati automaticamente dal sistema, ma se è necessario caricare un modulo all'avvio basta aggiungere una linea a

```
/etc/rc.d/rc.modules
```

per esempio:

```
/sbin/modprobe ide-scsi
```

Initrd

Capita spesso di voler creare un kernel minimale con meno funzionalità possibile compilate in modo monolitico, ed invece il massimo supporto esterno per tutti i moduli di cui si può aver bisogno.

In questo modo spesso nasce un problema: il kernel ha così poche funzionalità che non è in grado di bootare il sistema da solo, ma ha bisogno di moduli esterni, i quali non sono però disponibili finché almeno il root filesystem non è montato. Quindi per esempio se avete compilato il supporto a ReiserFS come modulo e il vostro filesystem di root è proprio ReiserFS allora il sistema non è in grado di bootare ma all'avvio avrete un kernel panic.

Per risolvere il problema è necessario creare un piccolo filesystem da caricare su RAM nella fase di boot iniziale con i moduli necessari per avviare correttamente il sistema. Per fare ciò si utilizza il comando mkinitrd:

```
mkinitrd -c -k 2.6.11 -m reiserfs:ext3 -f reiserfs -r /dev/hda7
```

dove -k specifica la versione del kernel e dei moduli da utilizzare, -m specifica la lista dei moduli da inserire, -f specifica il filesystem di root e -r la partizione di root.

Mkinitrd produce l'immagine zippata da caricare in ram **/boot/initrd.gz** a questo punto basta solamente aggiungere a lilo.conf la linea:

```
initrd=/boot/initrd.gz
```

e naturalmente rieseguire l'ilo.

MySQL

Directory e configurazione in Slackware

Una volta installato direttamente dai cd della Slackware, prima di mettere in funzione mysql bisogna lanciare:

```
mysql_install  
mysql_install_db
```

dopodichè i database con anche la configurazione del permessi si trovano in:

```
/var/lib/mysql
```

per lanciare il server basta poi un semplice

```
/usr/bin/mysqld_safe &
```

oppure

```
/usr/libexec/mysqld --user mysql &
```

per configurare gli accessi e i diritti utilizzare il tool a linea di comando mysql sostituendo il valore di host con il carattere speciale % per permettere l'accesso da remoto da qualsiasi host.

Copiare un database: metodo grezzo

Per copiare un database bisogna prima distinguere due casi: se è MyISAM o InnoDB.

Nel primo caso basta andare in /var/lib/mysql e si troverà una directory col nome del db da copiare contenente tutto il necessario, quindi basta copiarla pari pari.

Nel caso di InnoDB oltre a questo è necessario copiare anche i files presenti in /var/lib/mysql che iniziano per "ib", ma poichè tutti i database InnoDB hanno i dati contenuti insieme in quegli stessi files, in questo modo in realtà si copiano tutti i databases.

Si può in alternativa duplicare pari pari tutta /var/lib/mysql e poi eliminare con mysql i databases che non ci interessano, oppure convertire in MyISAM, copiare e riconvertire in InnoDB con mysql. Assicurarsi sempre che tutti i files in /var/lib/mysql siano di proprietà dell'utente mysql.

Copiare un database: metodo corretto

Prima si crea un backup del database:

```
mysqldump {sheepmail} > {sheepmail.sql}
```

Poi si copia il file di backup sul secondo server, si crea sul secondo server un database di nome {sheepmail} e poi:

```
mysql -u USER -p {sheepmail} < {sheepmail.sql}
```

Configurazione di rete e firewall

Configurazione della rete

Primaditutto è necessario configurare la scheda di rete:

```
ifconfig eth0 192.168.0.1 netmask 255.255.255.0 up
```

poi la tabella di routing:

```
route add -net 192.168.0.0 netmask 255.255.255.0 eth0
route add default ppp0
```

per permettere l'ip forwarding:

```
echo "1" /proc/sys/net/ipv4/ip_forward
```

Configurazione della rete con IPRROUTE2

```
ip link set d ev eth0 up
```

```
ip addr add 192.168.0.1 dev eth0
```

```
ip route add 192.168.2.0/24 via 192.168.0.1 dev eth0
```

Firewall e nat

In iptables ci sono tre tabelle:

nat, per il cambimento di indirizzi e port dei pacchetti

mangle, per agire su TOS e TTL, la meno utilizzata

filter, per filtrare il traffico.

Per nat ci sono le 3 catene **OUTPUT**, **PREROUTING** e **POSTROUTING**;

per filter ci sono le 3 catene: **INPUT**, **OUTPUT** e **FORWARD**.

Per nat ci sono le seguenti azioni: **SNAT** ovvero cambiare l'indirizzo sorgente, **DNAT** per l'indirizzo di destinazione, **MASQUERADE** per il masquerading.

Per filter le azioni sono **ACCEPT** e **DROP**.

Se è necesario configurare il nat:

```
iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -d !
192.168.0.0/24 -j MASQUERADE
```

se è necessario fare anche modifiche sulle porte di destinazione dei pacchetti:

```
iptables -t nat -A PREROUTING -p tcp -d 192.168.0.2 -dport 8081 -
to 192.168.0.1:80
```

per configurare il firewall con logica positiva invece:

```
#<inutili>
```

```
iptables -t filter -A FORWARD -s 192.168.0.0/24 -j ACCEPT
```

```
iptables -t filter -A FORWARD -d 192.168.0.0/24 -j ACCEPT
iptables -t filter -A INPUT -p tcp -s 127.0.0.0/24 -j ACCEPT
iptables -t filter -A INPUT -p tcp -s 192.168.0.0/24 -j ACCEPT
#</inutili>
iptables -t filter -A INPUT -p tcp --dport 23 -s ! 192.168.0.0/24
-j DROP
```

Altre configurazioni per il login

Infine altre configurazioni che possono risultare importanti per la sicurezza del sistema sono:

`/etc/securetty` – definisce le console fisiche e remote da cui si può loggare root

`/etc/login.access` – definisce quali utenti si possono loggare da dove e con che diritti. La sintassi è la seguente: `+:root:LOCAL` che stabilisce che l'utente root si può loggare da locale

`/etc/hosts.allow` – gli host che possono accedere ai servizi di `inetd`

`/etc/hosts.deny` – gli hosts che non possono accedere ai servizi di `inetd`

`/etc/hosts.equiv` - gli host da considerare “equivalenti” nel senso che gli user con lo stesso username possono eseguire comandi in remoto senza autenticazione con `rsh`

Configurazione tunnel Ipv6 con Iproute2

I comandi sono i seguenti:

```
ip tunnel add he mode sit remote $IP4DELTUNNEL
```

crea il tunnel

```
ip link set dev he up
```

attiva il tunnel

```
ip -6 addr add $LOCALIPV6 dev he
```

aggiunge l'indirizzo ipv6 del lato locale del tunnel

```
ip -6 route add 2000::/3 via $IPV6DELTUNNEL dev he
```

semplice tabella di routing

```
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

per attivare il forwarding su ipv6

Configurazione di Xfree86

Il file di configurazione di Xfree86 è /etc/X11/XF86Config.

Può essere tranquillamente editato a mano, oppure si può utilizzare qualche tool da console che permetta di visualizzare tutti i drivers disponibili :

```
/usr/X11R6/bin/xf86config
```

I moduli relativi ai drivers delle schede video sono anch'essi sotto /usr/X11R6, solitamente in

```
/usr/X11R6/lib/modules/drivers
```

A grandi linee il file /etc/X11/XF86Config, nelle sezioni riguardanti lo schermo, è organizzato come segue:

una o più sezioni monitor che descrivono i monitor a disposizione:

```
Section "Monitor"
    Identifier "Philips 107S"
    HorizSync 30-70
    VertRefresh 50-160
    vendorname "[Philips 107S]"
    modelname "[Philips 107S]"
    ModeLine "1024x768/85Hz" 98.9 1024 1056 1216 1408 768 782 788
822 -HSync -VSync
EndSection
```

una o più sezioni device che descrivono una o più schede video:

```
Section "Device"
    Identifier "3D Blaster"
    Driver "nvidia"
    Option "CloneDisplay" "2"
EndSection
```

una o più sezioni screen che descrivono le associazioni device\monitor:

```
Section "Screen"
    Identifier "Screen 1"
    Device "3D Blaster"
    Monitor "Philips 107S"
    DefaultDepth 24

    Subsection "Display"
        Depth 24
        Modes "1024x768" "1024x768" "800x600" "640x480"
    EndSubsection
EndSection
```

ed infine le sezioni serverlayout che specificano come gestire la presenza contemporanea di più screens e i corrispettivi dispositivi di Input:

```

Section "ServerLayout"

    Identifier "Simple Layout"
    Option      "Clone"          "on"
    Screen "Screen 1" Absolute 0 0
    Screen "Screen 2" Absolute 0 0
    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection

```

La configurazione corretta di un mouse con rotellina è la seguente:

```

Section "InputDevice"
    Identifier "Mouse1"
    Driver     "mouse"
    Option     "Protocol" "IMPS/2"
    Option     "ZAxisMapping"          "4 5"
    Option     "Device" "/dev/psaux"
    Option     "Emulate3Buttons" "true"
    Option     "Emulate3Timeout" "70"
    Option     "SendCoreEvents" "true"
EndSection

```

Una volta configurato correttamente XF86Config non ci resta che eseguire startx e la sessione X si avvia automaticamente con il gestore finestre predefinito. Per modificare il Window Manager scelto ed altri gestori d'avvio della sessione editare:

```
{/home/kaball}/.xinitrc
```

oppure il file globale di configurazione:

```
/etc/X11/xinit/xinitrc
/usr/X11R6/lib/X11/xinit/xinitrc
```

Inoltre è anche possibile avviare più X sessioni contemporaneamente, per fare ciò è necessario avviare le altre dopo la prima su display differenti:

```
startx -- :{display number}
```

per esempio:

```
startx -- :1
```

Masterizzare con un 2.4 e un masterizzatore ide

Configurazione

Per masterizzare con un kernel 2.4 e un masterizzatore ide è necessario emulare il masterizzatore scsi; per fare questo è necessario che nel kernel siano abilitate le seguenti voci:

Dalla sezione "SCSI support" :

```
SCSI support
SCSI CD-ROM support
Enable vendor-specific extensions (for SCSI CDROM)
SCSI generic support
```

Dalla sezione "File systems" abilitate:

```
ISO 9660 CDROM file system support
Microsoft Joliet CDROM extensions
```

Dalla sezione "Block devices" abilitate:

```
Loopback device support
```

Dalla sezione "ATA/IDE/MFM/RLL support" scegliete quindi "IDE, ATA and ATAPI Block devices" e abilitate:

```
SCSI emulation support
```

non ha importanza se vengno abilitati come moduli oppure built-in.

A questo punto basta aggiungere la linea

```
append="hdb=ide-scsi"
```

in lilo.conf nella parte relativa al kernel selezionato, in modo da specificare che il device hdb (o chi per esso) debba essere emulato scsi.

Per sapere quale device corrisponde al masterizzatore basta un `dmesg|less` e dare un'occhiata.

Se tutto va bene al prossimo riavvio, con un `cdrecord -scanbus` si vede il proprio masterizzatore.

Masterizzare da linea di comando

Per masterizzare da linea di comando utilizzare:

```
mkisofs -v -o cdimage.cd -R /home/mario/cd
```

per creare l'immagine, e:

```
cdrecord -v -dev=0,1,0 -eject -speed=x -data cdimage.cd
```

per masterizzarla.

Leggere DVD video

Per potersi guardare i films in dvd sotto linux, è necessario aver installato una serie di librerie e configurato il kernel a dovere.

Configurazione del kernel

Per vedere i dvd senza che i video scattino bisogna abilitare il DMA per il lettore e l'hard disk, e per fare questo è necessario aver configurato il kernel attivando il driver corretto per la propria scheda madre.

Dalla sezione "ATA/IDE/MFM/RLL support" scegliete quindi "IDE, ATA and ATAPI Block devices" e abilitate:

```
Generic PCI IDE chipset support
Generic PCI bus-master DMA support
Using PCI DMA by default when available
Intel PIIXn chipsets support
```

disabilitare:

```
Enable DMA only for disks
```

Per abilitare manualmente il DMA oppure per vedere se è attivato utilizzare:

```
hdparm -d1 /dev/hdd
```

Librerie e programmi

Le librerie necessarie sono:

```
libcdio
libdvdcss
libdvddread
```

Utilizzando il dvdrip megapack di Slacky si prende tutto il necessario (e anche di più) in un colpo solo.

Lettore CD pianta il sistema al mount

Se al momento di fare il mount di un cd il sistema si pianta, allora il problema è dovuto alla gestione del lettore cd da parte del kernel. Si può quindi provare a disabilitare il DMA se prima era abilitato:

```
hdparm -d0 {/dev/hdb}
```

oppure si può tentare di cambiare il driver usato dal sistema per il proprio controller ide:

```
lspci
```

per vedere qual'è il proprio controller ide, quelli supportati dal proprio kernel sono quelli che si trovano in:

```
ATA/IDE... -> IDE,ATA -> IDE Chipset Support
```

Il magico mondo della CLI

Ssh

Ssh non ha bisogno di presentazioni.
Utilizzo tipico:

```
ssh -l kaball -p 2222
```

-l utente
-p porta

è anche possibile utilizzarlo per eseguire uno o più comandi in batch:

```
ssh -q -o "BatchMode=yes" kaball@stabellini.net {decompall}
```

-q e -o "BatchMode=yes" sono due opzioni consigliabili per l'utilizzo in questa modalità, decompall invece è lo script che viene eseguito sulla macchina remota.

Le chiavi per la crittazione delle altre macchine conosciute sono memorizzate host per host e vengono conservate in `{/home/kaball/}.ssh/known_hosts`.
`/etc/ssh` contiene tutta la configurazione:

`/etc/ssh/ssh_config` la configurazione del client
`/etc/ssh/sshd_config` la configurazione del server
le varie chiavi pubbliche e private per la crittazione

Tra le varie opzioni interessante è:

X11Forwarding	sul server
ForwardX11	sul client

che se abilitate permettono al client di lanciare interfacce grafiche sulla sua macchina a partire da quella remota.

Un'altra utilissima feature di ssh è il cosiddetto **port forwarding**. In poche parole è possibile aprire una connessione ssh da una macchina A ad una macchina B e fare in modo che tutte le richieste di connessione alla porta portB della macchina B siano forwardate sulla macchina A sulla porta portA. Per esempio supponiamo che la macchina A sia dietro un nat quindi non raggiungibile dall'esterno, ma che invece la macchina B lo sia. E' possibile aprire una connessione ssh particolare dalla macchina A alla macchina B e fare in modo che tutte le connessioni sulla porta 2222 della macchina B siano in realtà ridirezionate sulla porta 22 della macchina A:

```
ssh -R *:2222:127.0.0.1:22 kaball@stabellini.net
```

In realtà naturalmente possiamo scegliere qualsiasi porta della macchina B e qualsiasi porta e host a cui forwardare la connessione dalla macchina A.

Oppure possiamo persino fare il contrario: scegliere una porta sulla macchina A ed accettare connessioni, le quali verranno poi tunnelizzate sulla macchina B e ridirezionate sull'host e porta scelti.

Un problema che spesso si incontra con ssh accade quando si hanno più server ssh sullo stesso IP agganciati a porte differenti. Per esempio nel caso di una intera rete dietro un NAT, con due differenti server ssh su questa rete, uno raggiungibile tramite port forwarding sulla porta 22

dell'interfaccia esterna del router e l'altro sulla porta 2222. Oppure se si ospitano sullo stesso sistema operativo più macchine virtuali ognuna con un server ssh. Nel primo caso dall'esterno appare un unico ip con 2 server ssh su due porte diverse, e la stessa cosa nel secondo caso, solo che l'ip in questione è 127.0.0.1. Il problema è che per evitare gli attacchi man-in-the-middle ssh memorizza le associazioni ip-chiave rsa nel file `.ssh/known_hosts` e passando da una connessione ad un server all'altra sullo stesso ip sembrerà che sia cambiata la chiave e quindi ci siano problemi di sicurezza. Per ovviare al problema il metodo più semplice è cancellare il file `known_host` oppure disabilitare il controllo sulle coppie IP-chiave settando l'opzione `CheckHostIP` a `no` nel file di configurazione `/etc/ssh/ssh_config`. Oppure è possibile disabilitare tale controllo solo per localhost settando sempre nello stesso file:

```
NoHostAuthenticationForLocalhost yes
```

Scp

Per copiare files tra macchine differenti basandosi su ssh l'utility base è scp. E' comunque meglio per queste cose utilizzare rsync.

Esempio tipico:

```
scp -P 2222 -r cmv kaball@81.74.17.71:/home/kaball
```

-P porta
-r ricorsivo

Se è importante il resume della copia nel caso di un'interruzione si deve utilizzare rsync.

Rsync

Per copiare o spostare files tra diversi computer o anche sullo stesso filesystem, per mantenere sincronizzate due directory, non c'è nulla di meglio che rsync.

Oltre che copie locali, è in grado di fare copie remote tramite shell remote come rsh o ssh, oppure utilizzando il suo server specifico rsync; infine è in grado di utilizzare un server rsync anche passando attraverso una shell remota per il trasporto.

Le sue caratteristiche principali sono 2:

- una grande potenza e flessibilità, che permette di copiare anche link simboli e non, agire sui diritti, ecc.;
- estrema efficienza nella banda utilizzata tramite la capacità di comprimere i dati da scambiare e uno speciale algoritmo che fa sì che vengano copiati automaticamente solo le parti dei files che cambiano effettivamente.

Esempio tipico di utilizzo: copia di files in remoto appoggiandosi a ssh, nessun server rsync

```
rsync -zr -e "ssh -l tomcat -p 2222" documenti  
81.74.17.71:/usr/local/tomcat/webapps/cmv/
```

-z comprime
-r ricorsivo
-e utilizza la shell specificata

importante non scrivere documenti/ con lo slash finale, perchè in questo caso copia il contenuto senza la directory padre.

Per trasferire un file con supporto per il resume in caso di interruzione si può utilizzare l'opzione -P:

```
rsync -P file user@host:/path
```

Per limitare la banda utilizzata usare l'opzione **-bwlimit=KBPS**.

Nel caso in cui si abbia a che fare con files di grosse dimensioni, almeno 1GB, e si abbia a disposizione una rete molto veloce (una lan), utilizzare l'opzione **--inplace** puo' migliorare parecchio le prestazioni.

Trickle

Trickle è un piccolo programmino che serve a limitare la banda utilizzata dal programma chiamato come argomento.

Per esempio:

```
trickle -u 10 -d 20 ncftp
```

limita la banda in upload a 10 kbps e quella in download a 20 kbps di ncftp che viene successivamente eseguito.

Du, df e ls

Per vedere quanti bytes occupa una directory con tutto il suo contenuto c'è il semplice comando du.

```
du -m documenti
```

per vedere in MB quanto contiene la dir documenti.

Ls invece è il semplice comando per vedere il contenuto di una directory, ls -la per vedere anche i diritti e la dimensione di ogni singolo file.

Df è il comando per vedere mo stato di occupazione dei deschi presenti nel sistema.

Monitoraggio di sistema e Hardware

I comando utili per accertarsi lo stato di salute del proprio server sono:

ps – elenca i processi in esecuzione, in particolare ps -eaf o ps aux elencano tutti i processi con tutte le info util

df – visualizza lo stato di occupazione dei dischi rigidi presenti

top – visualizza l'utilizzo di memoria e l'occupazione percentuale della cpu per ogni processo

free – visualizza lo stato di occupazione complessivo delle memorie

lsof – visualizza un grafico che rappresenta il carico del sistema

vmstat – visualizza lo stato di utilizzo complessivo di memorie e cpu

who – visualizza gli utenti presenti

w – visualizza gli utenti presenti con più dettagli anche su cosa stanno facendo

last – visualizza l'elenco di tutti i login e i reboot che sono avvenuti nel sistema con la data e l'ora

lastlog – visualizza la data, l'ora e la console degli ultimi login per ogni utente

faillog - visualizza gli errori di inserimento user e pass per un utente

dmesg - stampa i messaggi di bootup della macchina

lspci – stampa l'elenco dei dispositivi sulla PCI

lsusb – stampa l'elenco dei dispositivi connessi tramite usb

lsmod – visualizza l'elenco dei moduli del kernel caricati

Screen

Se si deve mettere in esecuzione un'applicazione con un'interfaccia grafica testuale e si ha necessità di lasciarla in esecuzione anche dopo essere usciti dal sistema, per poi riprendere la sessione al prossimo login, si deve utilizzare screen.

Screen è un'utility che permette di eseguire un'applicazione e poi distaccarne l'interfaccia lasciandola in esecuzione per poi riattaccare l'interfaccia quando si vuole.

Si lancia in questo modo:

```
screen {mute_text_gui}
```

per fare il detach:

```
ctrl+a+d
```

per riportare la finestra in foreground:

```
screen -R
```

Se vi siete dimenticati di detachare una sessione e volete ricollegarvi lo stesso utilizzate:

```
screen -x
```

Find

Per fare ricerche tra file si utilizza find. E' molto complesso e potente, riporto alcuni casi tipici di utilizzo.

Per cercare tutti i files ricorsivamente dalla directory corrente che abbiano il nome che contiene prova:

```
find ./ -name \*prova*
```

per cercare tutti i files della dir corrente e non nelle sottodirectory con maschera dei permessi 777, e stampare il contenuto di ognuno di essi:

```
find ./ -maxdepth 1 -perm 777 -exec cat {} ';' 
```

da notare che -exec non mette in esecuzione i comandi che seguono da shell, ma direttamente; quindi non si possono mettere in esecuzione tramite -exec più comandi in pipe.

-exec cat {} '|' grep non ha senso

Dig

Il tool più completo per la risoluzione dei nomi è dig.

Per risolvere semplicemente un hostname:

```
dig {host}
```

per la risoluzione inversa:

```
dig -x {ip}
```

per risolvere un hostname in un ipv6:

```
dig {host} AAAA
```

per la risoluzione inversa di un indirizzo ipv6:

```
dig -n -x {ipv6}
```

Split

split è una semplice utility per spezzare un file in uno o più altri file di una certa dimensione massima che possiamo fissare a nostro piacere.

Per esempio se vogliamo splittare il file huge in tanti file di massimo un gigabyte chiamati con prefisso nuovo:

```
split -b 1000000000 {huge} {nuovo}
```

se poi vogliamo riunirli in un unico file chiamato ancora huge:

```
cat {nuovo}* > {huge}
```

Tar, gz e bz2

tar è l'utility base per la compressione e la decompressione di archivi sotto unix.
Per decomprimere un archivio tar.gz:

```
tar xvzf {archivio.tar.gz}
```

per decomprimere un archivio tar.bz2:

```
tar xvjf {archivio.tar.bz2}
```

per comprimere una directory in un archivio tar.gz:

```
tar cvzf {archivio.tar.gz} {directory}
```

per comprimere una directory in un archivio tar.bz2:

```
tar cvjf {archivio.tar.bz2} {directory}
```

per vedere il contenuto di un archivio tar.gz:

```
tar tvzf {archivio.tar.gz}
```

per vedere il contenuto di un archivio tar.bz2:

```
tar tvjf {archivio.tar.bz2}
```

Zip e Unzip

A volte ci si trova costretti per una lunga serie di motivi a dover utilizzare anzichè tar.gz il classico formato zip di windows.

In questo caso per comprimere:

```
zip {archivio.zip} -r {directory}
```

per decomprimere:

```
unzip {archivio.zip} -b {directory}
```

per vedere il contenuto:

```
unzip -l {archivio.zip}
```

Rar e Unrar

Il rar è un formato decisamente antipatico, ma in alcuni casi può risultare molto comodo, soprattutto per le sue capacità di compressione, per la diffusione nel mondo windows, e per la sua capacità di creare volumi differenti di uno stesso file di una dimensione precisa.

Per comprimere:

```
rar a -v{1000000}k -r -m{4} {ut2004.rar} {/mnt/win/UT2004/}
```

dove **a** è il comando per comprimere, **-v** server per specificare la dimensione dei volumi, il numero che segue esprime appunto la dimensione in kilobytes, **-r** sta a significare “recurse subdirectories”, **-m**

invece serve per specificare il livello di compressione, che va da 1 a 5. Seguono il nome del file di destinazione e la lista dei files o directories da includere.

Per scompattare invece:

```
unrar x {archivio.rar} {destinazione}
```

Per vedere solo il contenuto:

```
unrar l {archivio.rar}
```

Formattazione

Per formattare una partizione di un'hard disk l'utility base è **mkfs** che poi chiama in causa i vari `mkfs.msdos`, `mkfs.ext2`, `mkfs.reiserfs`, ecc. a seconda del filesystem scelto.

Per utilizzarlo basta:

```
mkfs -t {ext3} {/dev/hda1}
```

Partizionare

Per partizionare un disco il comando è **fdisk**, ma si può anche utilizzare **cdisk**, un'utility testuale con un comodo menù guidato che rende tutto più semplice e intuitivo.

Aumix

Mixer con interfaccia sia grafica che testuale, utile per fissare tramite uno script i settaggi desiderati all'avvio. Prima si esegue `aumix` normalmente con l'interfaccia grafica, si settano tutti i volumi e le altre proprietà desiderate, si salvano i settaggi in `~/aumixrc`, poi si aggiunge in uno degli script di avvio al login, come per esempio `.bash_login`:

```
aumix -L
```

in questo modo richiama i settaggi precedentemente salvati e li rende attivi nel sistema.

Su

Comando di sistema per cambiare utente. Di solito è utilizzato da un utente normale per diventare root momentaneamente per eseguire qualche comando che necessita dei privilegi di amministratore, ma viene anche utilizzato dall'amministratore per lanciare un demone come altro utente:

```
su nobody -c "/usr/bin/squid"
```

md5sum

Comando che esegue l'algoritmo md5 su un file per controllarne la correttezza.
Per calcolare l'md5 di un file:

```
md5sum file.ext > file.md5
```

per controllare l'md5 di un file:

```
md5sum -c file.md5
```

GnuPGP

Implementazione GNU dell'algoritmo di criptazione PGP.
Per creare una propria chiave:

```
gpg -gen-key
```

la pass-phrase che viene chiesta durante la generazione della chiave vi verrà altresì chiesta tutte le volte che dovrete usare la vostra chiave privata. Inoltre alla chiave viene associato un ID univoco:

```
pub 1024D/6F957033 2005-01-24 Stefano Stabellini  
(http://www.stabellini.net) <stefano@stabellini.net>  
sub 1024g/D18172F0 2005-01-24
```

l'ID è: 6F957033

Per vedere l'elenco delle chiavi pubbliche di cui siete in possesso:

```
gpg -list-keys
```

oppure

```
gpg -fingerprint
```

per avere anche i corrispettivi fingerprint (piccoli codici che possono essere calcolati dalla chiave pubblica, utili per stampare i promemoria da scambiarsi).

Per esportare la propria chiave pubblica in un file:

```
gpg -a --export <mio_id> > miachiave.asc
```

il -a serve ad indicare di utilizzare il formato testuale.

Per esportare la propria chiave in un keyserver:

```
gpg --keyserver pgp.mit.edu --send-key <mio_id>
```

Per importare una chiave pubblica da un file o da un keyserver:

```
gpg --import fabiux.asc  
gpg --keyserver pgp.mit.edu --recv-key <id_esterno>
```

Una volta che avete importato una chiave pubblica potrete decriptare i messaggi criptati dall'utente relativo a quella chiave oppure potrete controllare le firme dei suoi messaggi, ma la chiave non sarà comunque nel circolo delle vostre chiavi “fidate”.

Per fare ciò bisogna “firmare” la chiave pubblica importata:

```
gpg --edit-key <id_esterno>
```

dopodiché in modalità interattiva si dà il comando **sign**. A questo punto se esporterete la chiave pubblica con id <id_esterno>, essa conterrà anche la vostra firma ad identificare che voi la considerate fidata.

Per firmare un file senza codificarlo:

```
gpg -sba {file}
```

per controllare la firma di un file:

```
gpg -verify {file.gpg}
```

per crittare un file per il destinatario che ha come id della sua chiave pubblica <id>:

```
gpg -r <id> -encrypt-files {file}
```

per decriptare un file:

```
gpg -decrypt-files {file}
```

Emacs

emacs è un editor di testo molto potente, di seguito sono riassunti i comandi base per il suo utilizzo in questo senso:

C-s string

ricerca incrementale:

ripetendo ulteriormente C-s si passa all'occorrenza successiva della stringa, premendo invio si ritorna in modalità normale e la ricerca termina. Tuttavia se si vuole riprendere la stessa ricerca successivamente lo si può fare ancora una volta premendo C-s C-s.

C-r string

ricerca incrementale all'indietro.

```
M-x replace-string <RET> string <RET> newstring <RET>
```

sostituzione di stringhe.

```
M-x query-replace <RET> string <RET> newstring <RET>
```

sostituzione di stringhe con richiesta di conferma per ogni sostituzione.

```
M-x replace-regexp <RET> c[ad]+r <RET> \&-safe <RET>
```

sostituzione di regular expressions senza richiesta di conferma; \& sta per l'intera stringa che è stata matchata.

```
M-x query-replace-regexp <RET> regexp <RET> newstring <RET>
```

stessa cosa ma con richiesta di conferma.

```
C- _
```

annulla l'ultima azione. Dopo una sequenza di azioni **undo** se si effettua un'altra azione che non esegue cambiamenti, e successivamente si esegue un altro C- _ si effettua un **redo**

```
M-x font-lock-mode
```

per abilitare il **syntax highlighting**.

RTorrent

Per tutti coloro che hanno bisogno di un client da linea di comando per bittorrent e si sentono limitati da btdownloadcurses la soluzione è Rtorrent.

Il suo utilizzo non è immediato ma in compenso il programma è molto potente.

Ecco come utilizzarlo:

```
rtorrent -d `pwd` -O download_rate=40 -O  
upload_rate=15 ./kubuntu-6.10-rc-desktop-powerpc.iso.torrent
```

dove -d specifica la directory in cui salvare i files, -O setta un'opzione, in questo caso vengono settate la massima velocità di upload e di download.

Una volta partito si può uscire da un programma premendo ctrl+q mentre si può cancellare un download tramite ctrl+d.

Sitecopy

Se avete un vostro sito web e l'unico tipo di accesso che avete sul server è quello ftp, può essere parecchio scomodo da utilizzare direttamente per fare l'upload di ogni cambiamento.

L'ideale sarebbe poter usare qualcosa come rsync, ma questo necessita di una shell sul server che spesso non è disponibile.

Per risolvere il problema c'è sitecopy, un piccolo programmino che offre funzionalità simili a rsync ma al di sopra di ftp, rendendo quindi il vostro lavoro molto più semplice.

Configurarlo è molto semplice: andate nella vostra home directory e poi eseguire:

```
mkdir -m 700 .sitecopy
touch .sitecopyrc
chmod 600 .sitecopyrc
```

a questo punto editate `.sitecopyrc`:

```
site nomesito
    server ip
    remote /
    local /var/www/htdocs
    username nomeutente
    password password
```

dove **nomesito** è un nome qualsiasi indicativo del vostro sito, **ip** è l'ip del vostro server, **/var/www/htdocs** è la directory nel filesystem local col vostro sito, **nomeutente** e **password** sono quelli per accedere al server via ftp.

Ora se sul server non è ancora presente nulla eseguite:

```
sitecopy --init nomesito
```

invece se sul vostro server è già presente una versione del vostro sito e questa è sincronizzata con quella presente in locale:

```
sitecopy --catchup nomesito
```

A questo punto ogni volta dopo avere modificato i files in locale eseguite:

```
sitecopy -update nomesito
```

è il gioco è fatto: sitecopy uploaderà solo i files effettivamente aggiornati.

Log di sistema

`{/home/kaball/}.bash_history` – contiene l'elenco di tutti i comandi eseguiti da un utente, viene aggiornato al logout. Il comando su viene considerato logout e ingresso di un nuovo utente.

`/var/log` – in questa dir e nelle sue sottodirectory sono contenuti i files di log di tutti i programmi di sistema, per esempio apache, sendmail, ecc.

`/var/log/syslog` - log di tutti i messaggi di sistema: log falliti per timeout di ssh, messaggi di bootup, errori nel caricamento moduli, ecc.

`/var/log/secure` – messaggi di **login degli utenti**

`/var/log/wtmp`, `/var/log/faillog`, `/var/log/lastlog` – contengono log in formato binario consultabili tramite i comandi **last**, **faillog** e **lastlog**

`/var/log/messages` – messaggi del kernel e altre applicazioni, tra cui **login falliti tramite ssh**.

`/var/run/utmp` – contiene l'ultimo accesso degli utenti in formato binario, è consultato da `lastlog`

Il demone di sistema che si occupa di loggare gli eventi è `/usr/sbin/syslogd`; viene messo in esecuzione al boot da `/etc/rc.d/rc.syslog`, e può essere configurato da `/etc/syslog.conf`.

In cron viene messo automaticamente un demone `logrotate` che gestisce la dimensione dei files di log eventualmente suddividendoli su più files. Può essere configurato tramite `/etc/logrotate.conf`.

Cron

Cron è un demone che viene fatto partire al boot della macchina da /etc/rc.d/rc.M; mette in esecuzione i lavori definiti in /var/spool/cron/crontabs/{utente} in un preciso giorno e ora della settimana .

Il formato da utilizzare è il seguente:

```
MIN HOUR DAY MONTH DAYOFWEEK  COMMAND
```

Per aggiungere dei lavori

```
crontab -e
```

oppure mettere degli script in

```
/etc/cron.daily  
/etc/cron.weekly  
/etc/cron.hourly  
/etc/cron.monthly
```

Atd

Atd è un demone che viene lanciato al boot della macchina da /etc/rc.d/rc.M, e mette in esecuzione i job setati col comando at

```
at -f {file} {time}
```

dove nel file c'è la lista dei comandi da eseguire. Notare bene che tale script viene messo in esecuzione una volta sola, e non in maniera ricorrente come con crond.

I jobs da eseguire vengono memorizzati in /var/spool/atjobs.

Dhcp

Dhcp è un sistema per assegnare ip e configurazione di rete automaticamente alle macchine in modo centralizzato, dinamico e controllato.

Dhcpd

dhcpd è semplicemente il demone client che sta in ascolto per ricevere le configurazioni di rete dal server. Per lanciarlo basta eseguire il comando dhcpd e penserà tutto lui.

Dhcpd

dhcpd è il demone server che assegna le configurazioni alle altre macchine; la sua configurazione è in /etc/dhcpd.conf. Si mette in esecuzione:

```
dhcpd eth0
```

dhcpd.conf

la sintassi è molto semplice. Per configurare un pool di indirizzi ip da essere assegnati a caso, assegnando anche il router di riferimento, il dominio e il dns server:

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.1 192.168.0.255;
    option domain-name-servers 192.168.0.1;
    option domain-name "darklight.net";
    option routers 192.168.0.1;
}
```

Per configurare una macchina con un indirizzo particolare a livello 2 con un indirizzo a livello 3, un host, un dns server e un router:

```
host poltergaist {
    hardware ethernet 00:E0:18:0F:B3:44;
    fixed-address poltergaist.darklight.net;
    option host-name "poltergaist";
    option domain-name-servers 192.168.0.1;
    option domain-name "darklight.net";
    option routers 192.168.0.1;
}
```

Xfree client-server scatenato

In questa sezione sono spiegate le funzionalità più avanzate di Xfree riguardo le sue potenzialità client-server.

Autenticazione del client

Come dicevo prima, le funzionalità che offre Xfree86 sono basate su un'architettura client-server che permette quindi, per esempio, di lanciare un'applicazione la cui interfaccia grafica venga visualizzata su una macchina differente.

Dal punto di vista di chi lancia l'applicazione, ovvero del client, la prima cosa di cui si deve occupare è settare opportunamente la variabile d'ambiente DISPLAY, che specifica appunto dove visualizzare la gui delle applicazioni grafiche.

```
export DISPLAY={host}:{display}.{screen}
```

quindi per esempio:

```
export DISPLAY=81.74.17.71:0.0
```

In alternativa per ogni applicazione che si lancia con un'interfaccia grafica si può aggiungere l'argomento `-display`:

```
{mozilla} -display {host}:{display}.{screen}
```

una volta specificato dove si vuole vedere l'interfaccia il problema da superare è l'autenticazione sul server X specificato in uno dei 2 modi appena spiegati.

L'X server infatti ascolta sulla porta 6000 e non permette a tutti di lanciare apps grafiche su di esso. La possibilità più semplice è l'autenticazione basata sull'host del client: sul server è possibile specificare una serie di hosts dai quali accettare connessioni, questo si fa tramite il comando `xhost`:

```
xhost +name
```

```
xhost -name
```

L'altra opportunità è di utilizzare un magic-cookie che normalmente quando un utente lancia `startx` viene posto nel file `{/home/kaball/.Xauthority}`. Chi possiede tale cookie può lanciare applicazioni su quel server; in locale infatti se un utente si copia l'.Xauthority di un altro utente nella sua cartella e setta opportunamente `$DISPLAY`, può tranquillamente lanciare le sue applicazioni grafiche nel desktop dell'altro utente.

Nel caso remoto la cosa si complica di più perchè nel magic-cookie è specificato anche l'host quindi oltre che copiarsi l'.Xauthority bisogna anche modificarlo a dovere.

Sul server si lancia:

```
xauth list $DISPLAY
```

come output c'è il codice esadecimale del cookie. Sul client dopo essersi loggati da remoto si lancia

```
xauth add {DISPLAY} . {cookie}
```

Nel caso che si utilizzi ssh per le connessioni la procedura può essere automatizzata e affidata del tutto al proprio client e server ssh abilitando l'X11 forwarding su entrambi.

E' bene ricordare che l'autenticazione automatica tramite ssh avviene solo per il primo dei display presenti sul client ssh (e server X), quindi se si sta facendo girare Xnest, e si vuole vedere su Xnest le proprie applicazioni, allora bisogna utilizzare uno dei primi due metodi

Xnest

Xnest è un programma che apre una finestra che non è altro che un altro X server annidato.

```
Xnest :1 &
```

Questo comando apre il nuovo display come :1. Per l'autenticazione utilizzare xhost, oppure l'opzione -ac che permette a tutti di connettersi.

Nel caso di autenticazione tramite cookie, anche tra utenti locali è necessario modificare l'.Xauthority, poiché il numero del display è differente. Per farlo utilizzare la stessa procedura, sul server:

```
xauth list $DISPLAY
```

e sul client:

```
xauth add {DISPLAY} . {cookie}
```

Xdmcp

Così come con rsh si può ottenere una shell da remoto, con Xdmcp si può ottenere un intero Xserver in remoto. Sul server gira un gestore dei login grafico come xdm, kdm, o gdm; il client all'avvio del suo Xserver lo ottiene connettendosi in remoto sul server xdm e ottenendo una sessione X in remoto. Un esempio di questa configurazione è sui pc di ingegneria.

In questo modo il calcolo viene anch'esso distribuito tra client e server.

Il client per ottenere una sessione X remota deve semplicemente lanciare:

```
X -indirect {ip}
```

mentre l'xdm va configurato opportunatamente per accettare connessioni remote.

Primaditutto ricordo che lo script d'avvio relativo alla modalità di login grafico è /etc/rc.d/rc.4, e proprio in questo script viene quindi messo in esecuzione il gestore grafico di login. Quindi modificando questo file si può scegliere il proprio gestore preferito, xdm, kdm, o gdm.

Per quanto riguarda xdm, la configurazione è nella cartella /etc/X11/xdm.

/etc/X11/xdm/Xaccess - contiene l'elenco degli host che si possono connettere; per permettere chiunque basta aggiungere un *

/etc/X11/xdm/xdm-config - contiene la configurazione generale, ricordarsi di commentare l'ultima riga per permettere il login da remoto

/etc/X11/xdm/Xservers - i server X da mettere in esecuzione

Per quanto riguarda kdm invece la configurazione è in /opt/kde/share/config/kdm:

/opt/kde/share/config/kdm/kdmrc

dove deve essere presente:

```
[Xdmcp]
Enable=true
Port=177
Xaccess=/opt/kde/share/config/kdm/Xaccess
HonorIndirect=false
MaxPending=4
MaxPendingIndirect=4
```

ricordarsi quindi anche di configurare il file /opt/kde/share/config/kdm/Xaccess con gli hosts corretti.

Samba

Samba è gestore di files system distribuiti compatibili con il sistema windows.

Il server è messo in esecuzione all'avvio da /etc/rc.d/rc.samba, e si chiama smbd, che è accompagnato da nmbd per la risoluzione dei nomi, il client testuale è smbclient, ma si possono tranquillamente aggiungere i filesystem remoti all'fstab come segue:

```
//poltergaist/c /mnt/poltc smbfs username={user}
```

oppure montarli manualmente con:

```
mount -t smbfs -o  
username={user},password={pass} //poltergaist/c /mnt/poltc
```

La configurazione del server è in /etc/samba/smb.conf, anche se può sembrare molto complicata per far funzionare il server basta:

```
[global]  
  
workgroup = DARKLIGHT  
  
server string = Samba Server  
  
hosts allow = 192.168.1. 192.168.2. 127. 192. 10.  
  
log file = /var/log/samba.%m  
  
max log size = 50  
  
security = share  
  
ssl CA certDir = /etc/ssl/certs  
  
socket options = TCP_NODELAY  
  
local master = yes  
  
os level = 33  
  
domain master = yes  
  
preferred master = yes  
  
dns proxy = no
```

in questo modo si configura il server in modo da essere quello predefinito e principale della rete, con livello di sicurezza share, ovvero tutti possono accedere alle risorse condivise senza bisogno di loggarsi. Anche in questa modalità viene comunque utilizzato uno username per i diritti di accesso ai files sul server, tale user può essere quello che viene inviato dal client, ma siccome è facoltativo in questa modalità, può essere lo user specificato come “guest account”, che è nobody di default. Se si vuole far sì che gli utenti si debbano loggare per poter accedere, allora bisogna cambiare

security:

```
security=user
```

in questo modo samba utilizza come utenti e password quelli di sistema, invece aggiungendo anche:

```
encrypt passwords = yes
```

gli utenti samba sono sempre quelli di sistema, ma la loro password è differente e va settata con:

```
smbpasswd -a {utente}
```

Per condividere una directory semplicemente:

```
[public]
path = /
public = yes
writable = yes
```

dove writeable sta ad indicare se gli utenti hanno diritto di scrittura. Nel caso di security=user si deve tenere conto anche degli effettivi diritti di lettura e scrittura dei vari utenti su quella directory.

Bind

Bind è il nameserver più diffuso al mondo, la sua configurazione principale è in /etc/named.conf, dove viene anche specificata la directory di sistema dove vi sono le configurazioni delle singole zone dns, che di default è /var/named.

Vediamo la configurazione base di /etc/named.conf:

```
options {
    forward first;
    forwarders {
        193.70.192.25;
    };
};
```

le opzioni precedenti specificano se appoggiarsi ad un altro nameserver oppure no, 193.70.192.25 è l'ip del dns di riferimento e forward first fa sì che la query sia prima inviata all'ip specificato poi agli altri servers.

```
    directory "/var/named";
```

questa è la directory dove sono contenuti i files di configurazione delle singole zone dns.

```
    allow-recursion {192.168.0.0/16; localhost};;
```

permette query ricorsive ai pool di ip specificati tra parentesi graffe

```
    allow-transfer { none; };
```

permette la sincronizzazione dei database ai server secondari specificati tra parentesi

```
    listen-on-v6 { any; };
```

fa sì che il dns server risponda anche a query per ipv6

```
};
```

```
zone "darklight.net" IN {
    type master;
    allow-update{none;};
    file "caching-example/darklight.zone";
};
```

esempio di zona diretta. Type master significa che il server in questione è il master per questa zona, allow-update serve per specificare se altri server possono aggiornare i record sul nostro server riguardo questa zona.

```
zone "0.168.192.in-addr.arpa" IN {
    type master;
    file "caching-example/darkinv.zone";
    allow-update{none;};
};
```

esempio di zona inversa.

Ora diamo un'occhiata ai files specificati come esempio nelle due zone precedent. Il seguente è il contenuto di darklight.zone:

```
$TTL      3D
```

specifica il tempo di vita del record fornito, in questo caso 3 giorni

```
@                IN SOA  kaball.darklight.net. 192.168.0.1  
(
```

kaball.darklight.net, con ip 192.168.0.1 è l'host di riferimento, solitamente anche il nameserver della zona. Da notare il punto obbligatorio al termine di kaball.darklight.net.

```
                42                ; serial  
(d. adams)  
                3H                ; refresh  
                15M               ; retry  
                1W                ; expiry  
                1D )              ; minimum
```

alcune opzioni tipiche di riferimento per la zona

```
;  
                NS      kaball
```

il nameserver predefinito della zona

```
;  
kaball          IN      A      192.168.0.1
```

kaball ha come indirizzo 192.168.0.1

```
poltergaist    IN      A      192.168.0.2
```

ora diamo un'occhiata a darkinv.zone il file di riferimento per la risoluzione inversa dei nomi della zona:

```
$TTL      86400
```

il tempo di vita in secondi del record

```
@                IN      SOA      kaball.darklight.net. 192.168.0.1 (
```

kaball.darklight.net, con ip 192.168.0.1 è l'host di riferimento, solitamente anche il nameserver della zona. Da notare il punto obbligatorio al termine di kaball.darklight.net.

```
                2003072106 ; Serial  
                28800     ; Refresh  
                14400     ; Retry
```

```
3600000 ; Expire
86400 ) ; Minimum
```

alcune opzioni tipiche di riferimento per la zona

```
IN NS kaball.darklight.net.
```

specifica qual'è il nameserver della zona

```
1 IN PTR kaball.darklight.net.
```

l'indirizzo 192.168.0.1 corrisponde a kaball.darklight.net. Da notare il punto obbligatorio al terminet dell'hostname.

```
2 IN PTR poltergaist.darklight.net.
```

l'indirizzo 192.168.0.2 corrisponde a poltergaist.darklight.net. Da notare il punto obbligatorio al terminet dell'hostname.

Per terminare porto un esempio di configurazione di una zona diretta e inversa ipv6. In named.conf:

```
zone "saw.something-at.area51.net" IN {
    type master;
    file "caching-example/lamero.zone";
    allow-update {none;};
};

zone "0.2.7.0.0.0.f.1.0.7.4.0.1.0.0.2.ip6.int" IN {
```

i primi 16 byte dell'indirizzo

```
    type master;
    file "caching-example/inv.zone";
    allow-update { none; };
};
```

lamero.zone:

```
$TTL 3D
@ IN SOA kaball.saw.something-
at.area51.net. sono.io (
                                42 ; serial
(d. adams)
                                3H ; refresh
                                15M ; retry
                                1W ; expiry
                                1D ) ; minimum
kaball IN NS kaball
IN AAAA 2001:470:1F00:720::1
```

inv.zone:

Apache in configurazioni avanzate

La configurazione dell'apache è in /etc/apache/httpd.conf, la directory radice è /var/www e la pagina di default d'avvio è in /var/www/htdocs.

Il file di configurazione è basato su una struttura del tipo:

```
Direttiva Valore
```

```
<sezione>
```

```
Direttiva Valore
```

```
</sezione>
```

Alcune tra le direttive più importanti sono:

```
Listen {ip}
```

che nel caso la macchina abbia più di un IP fa agganciare l'apache a solo un indirizzo;

```
ServerName {name}
```

che specifica l'hostname a cui l'apache si riferisce per chiamare se stesso;

```
UserDir {dir}
```

che specifica dove mappare le cartelle pubblicate di ogni utente, di default è public_html;

```
Redirect {dir richiesta nell'url} {url}
```

ridireziona tutte le chiamate a una dir all'url specificato;

```
UseCanonicalName {on\off}
```

specifica se l'apache deve utilizzare il servername immesso nella configurazione oppure quello immesso dall'utente;

```
<Directory />
```

```
..
```

```
</Directory>
```

```
<Location />
```

```
..
```

```
</Location>
```

la configurazione dei diritti di accesso ad ogni directory e di ogni location.

Il match con l'argomento della direttiva Directory viene fatto con un path sul filesystem, in cui si possono utilizzare wildcards come sulla bash. In particolare quando si usano wildcards bisogna far precedere l'argomento da "~". Bisogna anche prestare attenzione che un * non comprende mai il carattere "/", quindi ad esempio */home non matcha con /primo/secondo/home.

Le Location invece fanno match con l'url digitato dall'utente, non è possibile utilizzare wildcards (altre direttive come LocationMatches comunque lo permettono), e vengono processate dopo tutte le Directory. Sia le Directory che le Location, nel caso più di una matchi contemporaneamente, vengono eseguite nell'ordine in cui compaiono nel file di configurazione.

Notare anche che prima vengono matchate ed eseguite le direttive al di fuori di una sezione VirtualHost, poi quelle all'interno.

```
Alias {dir url} {dir fisica}
```

stabilisce degli alias tra le cartelle richieste nell'url e la loro posizione reale nel filesystem

```
NameVirtualHost *
```

se gli argomenti delle direttive VirtualHost devono matchare con l'host o con l'ip

```
<VirtualHost *>
    ServerAdmin webmaster@dummy-host.example.com
    DocumentRoot /var/www/htdocs/manual
    ServerName dragonsclaws.maginetnetwork.biz
    ErrorLog logs/dummy-host.example.com-error_log
    CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

permette di configurare più host o più ip con lo stesso server apache, mappandoli in maniera differente sulle directory del filesystem.

Utilizzo con tomcat

Utile in questo caso utilizzare la direttiva alias:

```
Alias /cmv/ "/usr/local/tomcat/webapps/cmv/"
```

Un solo server per più hosts

In questo caso bisogna configurare l'apache tramite VirtualHosts estensivamente differenziando le configurazioni.

```
<VirtualHost localhost>
    ServerAdmin webmaster@dummy-host.example.com
    JkMount *jsp worker1
    DocumentRoot /usr/local/tomcat/webapps/tollok
    ServerName localhost
    ErrorLog logs/dummy-host.example.com-error_log
    CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

```
<VirtualHost *>
    ServerAdmin webmaster@dummy-host.example.com
    DocumentRoot /var/www/htdocs/manual
    ServerName localhost
    ErrorLog logs/dummy-host.example.com-error_log
    CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

Un solo host con più server

Se si ha a disposizione un solo hostname, ma più macchine con più server apache, ognuna con un differente contenuto, allora la configurazione migliore è quella di settare una macchina come server principale, sulla quale girerà l'apache sulla porta 80 e sarà il server puntato di default dal router. Sulle altre macchine invece l'apache girerà su di una porta differente per esempio l'8081 e 8082, e il router sarà configurato per forwardare a loro quelle porte.

Inoltre per non costringere gli utenti a digitare sempre il numero di porta differente per raggiungere i contenuti che sono presenti nei server secondari si utilizza nella configurazione del server principale la direttiva Redirect:

Redirect /sheepmail <http://www.maginetwork.biz:8081/sheepmail>

Un host per più server replicati

In caso di replicazione dei server il problema non si pone nemmeno: basta settare il server dns in modo tale che per il dato hostname vi siano più ip corrispettivi, uno per ogni macchina replicata. In questo modo i client riceveranno ip differenti in modo casuale nella risoluzione dell'hostname e il carico risulta distribuito in modo uniforme automaticamente.

Apache + Tomcat +SSL con directory con differenti diritti di accesso

Se si ha configurato il proprio sistema in modo da far girare contemporaneamente Apache e Tomcat, con anche la possibilità di connessione SSL gestite in maniera trasparente da Apache, come mostrato in uno dei primi capitoli su questa guida, allora si vorrà anche fare in modo che la directory con le informazioni delicate sia accessibile solo tramite connessione SSL.

Per fare questo è necessario configurare correttamente i diritti della directory protetta e di tutte le altre directory del sistema sia in httpd.conf, file di configurazione dell'apache in chiaro, che in ssl.conf, configurazione di mod_ssl, sfruttando le capacità di virtualhosting di Apache.

In httpd.conf:

```
# NameVirtualHost *  
  
<VirtualHost _default_:80>  
  
<Location "/cmv/admin">  
AllowOverride None  
Deny from all  
</Location>  
  
</VirtualHost>
```

che specifica di utilizzare un virtualhosting ip based e che tramite porta 80 non si può accedere alla location cmv/admin, mentre in ssl.conf all'interno della sezione VirtualHost 443:

```
<Location "/">  
Deny from all  
</Location>  
  
<Location "/cmv/admin">  
Options Indexes FollowSymLinks  
Allow from all
```

```
</Location>
```

in modo tale che tutte le location tranne /cmv/admin matchano solo con la prima Location e quindi ne viene privato l'accesso tramite ssl, mentre /cmv/admin match prima con la prima Location e ne viene vietato l'accesso, poi con la seconda, quindi l'accesso viene riabilitato. Come risultato tramite ssl si può accedere solo a /cmv/admin.

Quando il parser dell'apache analizzerà la configurazione, dopo il primo include del file ssl.conf risulta:

```
<VirtualHost _default_:80>
<Location "/cmv/admin">
AllowOverride None
Deny from all
</Location>
</VirtualHost>
```

```
<VirtualHost _default_:443>
<Location "/">
Deny from all
</Location>
```

```
<Location "/cmv/admin">
Options Indexes FollowSymLinks
Allow from all
</Location>
</VirtualHost>
```

quindi come è evidente è stato differenziato il comportamento a seconda della porta di connessione, ed in ognuno dei due casi sono stati specificati diversi diritti di accesso delle directory.

Situazioni più complicate

E' meglio non trovarsi in situazioni più complesse, ma se non se ne può fare a meno si può sempre utilizzare le enormi capacità di Url Rewriting di Apache, con le quali si può praticamente fare tutto. Per contro vi è una certa difficoltà di configurazione, per una trattazione completa rimando al sito di Apache.

Apache e .htaccess

Se comprare hosting non avete accesso alla configurazione di apache, tutto quello che potete configurare lo dovete fare tramite il file .htaccess, che è un file “speciale” che viene interpretato da apache come direttive particolari per la directory in cui il file è contenuto. Naturalmente è possibile disabilitare o limitare le possibilità di riconfigurazione del webserver tramite questi file tramite il file di configurazione principale di apache.

Un esempio classico di utilizzo del file .htaccess è quello di voler configurare una directory in modo tale che quando ci si accede col browser apache mostri il suo contenuto, in questo caso basta creare un file .htaccess con la seguente linea di testo:

```
Options +Indexes +MultiViews +FollowSymlinks
```

Un altro esempio è voler proteggere con username e password l'accesso ad una cartella, in questo caso è necessario creare un file .htaccess di questo tipo:

```
AuthUserFile /path/to/.htpasswd  
AuthName EnterPassword  
AuthType Basic  
require valid-user
```

e successivamente eseguire il seguente comando:

```
htpasswd -c /path/to/.htpasswd nomeutente
```

il quale vi chiede la password per lo username nomeutente. In entrambi casi /path/to/.htpasswd è il path assoluto del file .htpasswd ovvero il file che conterrà l'elenco degli username e corrispettive password. E' preferibile porlo al di fuori delle directory del vostro sito internet.

Se volete aggiungere ulteriori utenti semplicemente eseguite ancora:

```
htpasswd /path/to/.htpasswd nomeutente
```

Gestione di una Slackware

In questa sezione spiegherò quali tool utilizzare per tenere aggiornata una slackware e come utilizzarli correttamente, chi pensa che il sistema di pacchettizzazione slack sia inferiore a rpm o deb si sbaglia di grosso.

I pacchetti

Sinteticamente i pacchetti Slackware tgz non sono altro che degli archivi tar.gz contenenti tutti i files di un programma con le directory all'interno del pacchetto assolute. In più vi è solo un'ulteriore directory install nella quale viene posto un file di testo description con la descrizione e uno script da eseguire all'avvio in bash chiamato doinst.sh.

Pkgtools

I tool che ci vengono messi a disposizione per la gestione dei pacchetti, naturalmente tutti a linea di comando, sono principalmente installpkg, removepkg, upgradepkg e pkgtool.

```
installpkg {pacchetto}
```

semplicemente installa un pacchetto tgz nel sistema, avendo cura di loggarlo correttamente in /var/log/packages, directory nella quale viene mantenuto un file di testo avente nome uguale a quello del pacchetto e contenente un elenco di tutti i files installati relativi a quel pacchetto. Siccome molti pacchetti tgz sono accompagnati da bash scripts da lanciare al momento dell'installazione, viene mantenuta una copia di backup anche dello script in /var/log/scripts. Allo stesso modo

```
removepkg {pacchetto}
```

rimuove un pacchetto dal sistema e sposta il suo file di log e il suo script loggato in /var/log/removed_packages e in /var/log/removed_scripts.

Infine upgradepkg aggiorna un pacchetto già presente nel sistema:

```
upgradepkg {pacchetto}
```

è anche in grado di upgradare il pacchetto se già presente se no installarlo:

```
upgradepkg {pacchetto} --install-new
```

oppure reinstallare un pacchetto:

```
upgradepkg {pacchetto} --reinstall
```

Infine pkgtool è un tool visuale che visualizza la lista dei pacchetti installati e permette di rimuoverli.

Swaret - configurazione

Swaret è il miglior tool da usare per scaricare e aggiornare pacchetti Slackware tgz, è anche in grado di effettuare controlli sulle dipendenze.
Il suo file di configurazione è swaret.conf, nel quale è necessario specificare primaditutto la corretta versione del proprio sistema:

```
VERSION=9.1
```

dopodichè si devono specificare gli Slackware mirrors preferiti, per esempio:

```
ROOT=ftp://ftp.slackware.no/pub/linux/slackware/slackware-\$VERSION
```

notare che molti protocolli sono supportati, tra cui anche rsync, che probabilmente è il migliore per mantenere continuamente aggiornati i pacchetti sulla current-version.

A questo punto si possono specificare anche uno o più Slackware Repositories, ovvero siti dove si possono trovare pacchetti slackware sfusi:

```
REPOS_ROOT=Slacky.it%http://www.slacky.it/download  
REPOS_ROOT=LinuxPackagesDOTNET  
%ftp://ftp.linuxpackages.net/pub/Slackware-9.1
```

Poi le liste di librerie, usate per i controlli sulle dipendenze, per esempio:

```
DEP_ROOT=http://www.swaret.org/swaret  
DEP_ROOT=ftp://ftp.swaret.org/swaret
```

Infine i settaggi veri e propri di swaret, nel suo file di configurazione di default la maggiorparte è corretta. Quelli importanti che io ho modificato sono:

per abilitare il controllo delle dipendenze

```
DEPENDENCY=1
```

per abilitare l'md5 check

```
MD5CHECK=1
```

per disabilitare il GPG check

```
GPGCHECK=0
```

la network interface da usare

```
NIC=ppp0
```

ricordarsi anche di personalizzare la lista dei pacchetti da escludere dalla gestione di swaret, che è presente nel file di configurazione sotto questa forma:

```
EXCLUDE=kernel
```

Possono essere utilizzati anche * ed espressioni tipiche da bash per il matching coi nomi dei pacchetti.

Di default in pacchetti sono scaricati in /var/swaret e il file di log è /var/log/swaret, ma possono essere cambiati entrambi nel file di configurazione.

Swaret - uso

Swaret è molto potente e può essere utilizzato in molti modi differenti.

Primaditutto occorre scaricare la lista aggiornata dei pacchetti presenti nei server specificati con:

```
swaret --update
```

poi si può cercare un pacchetto particolare in maniera interattiva con:

```
swaret --search {pacchetto}
```

tra la lista dei risultati poi se ce n'è uno interessante allora:

```
swaret --install {risultato}
```

oppure

```
swaret --upgrade {risultato}
```

per scaricare soltanto il pacchetto e poi eventualmente installarlo a mano:

```
swaret --get {risultato}
```

Se siamo invece interessati a mettere in cron un job di aggiornamento automatico di tutto il sistema allora:

```
swaret --upgrade -a
```

per upgradare tutto il sistema, invece per upgradare solo i pacchetti patchati:

```
swaret --upgrade -ap
```

Alien

Nel caso in cui riuscite a trovare il software che vi interessa soltanto in formato rpm o deb potete sempre provare a convertirlo in Slackware tgz con alien:

```
alien --to-tgz {pacchetto}
```

Checkinstall

Ma di gran lunga il migliore tool da utilizzare nel caso in cui non troviate il software già pacchettizzato per Slackware è checkinstall.

Scaricate i sorgenti dell'applicativo da installare, lanciate come al solito configure e make, ma invece di terminare con make install terminate con checkinstall, il programma vi chiederà qualche informazione poi genererà per voi il pacchetto tgz slackware e lo installerà nel sistema. Perfetto no? Inoltre il pacchetto che viene generato è di ottima qualità e segue abbastanza bene tutte le specifiche tradizionali per creare corretti pacchetti slack, praticamente è come se ve lo foste costruiti minuziosamente a mano.

Note su altri packets managers per Slackware

Esistono in realtà parecchi altri packets managers per Slackware sempre non ufficiali, ma anche di grande valore. Swaret fondamentale non fa altro che rendere più semplice l'uso dei pkgtools e aggiungere un hack per il controllo a posteriori delle dipendenze tramite ldd. Completamente differente è slapt-get, attivamente sviluppato e mantenuto, scritto completamente in C, è un sistema apt-get like per slackware, ovvero offre un'interfaccia notevolmente semplificata per l'utilizzo dei pkgtools, e inoltre effettua il controllo delle dipendenze tramite le informazioni sulle dipendenze stesse del pacchetto che devono essere specificate nel file slack-requires all'interno del pacchetto stesso. Purtroppo però non sono molti i pacchetti che contengono tali informazioni, sicuramente non quelli ufficiali, e nemmeno molti di quelli di linuxpackages.net. E' possibile utilizzarlo insieme ad un piccolo script in bash che aggiunge un rudimentale controllo delle dipendenze tramite ldd. Di notevole valore anche portpkg, un port system per Slackware, ovvero un sistema simile a portage di Gentoo ma scritto esplicitamente per slack. Offre un semplice sistema di controllo delle dipendenze automatico, grazie ad informazioni sulle dipendenze stesse collezionate durante la fase di "configure".

Infine vi sono molti sistemi che permettono l'integrazione di altri sistemi di pacchetti all'interno di Slackware, per esempio emerde (da emerge di Gentoo) e pkgsrc (da NetBSD). Ne sconsiglio l'utilizzo perchè può essere pericoloso mantenere contemporaneamente due diversi sistemi di pacchetti sullo stesso OS.

Conclusioni

Per gestire correttamente la vostra slack, quando volete installare o upgradarne i pacchetti utilizzate swaret o slapt-get più che potete, sono comodi potenti e sicuri. Non contate troppo sui controlli automatici delle dipendenze perchè non servono e poi non sono sempre così affidabili (nemmeno su Debian). Prima di upgradare l'intera distribuzione date sempre un occhio al file UPGRADE.TXT e al Changelog, per eventuali particolarità.

Se ottenete pacchetti tgz in altro modo, tramite download tradizionale per esempio, allora utilizzate upgradepkg o installpkg.

Se non riuscite a trovare pacchetti tgz ma solo i sorgenti, scaricateli poi utilizzate checkinstall per creare il pacchetto tgz e installarlo.

Se trovate solo rpm o deb prima di disperarvi fate un tentativo con alien, ma deve essere davvero l'ultima vostra possibilità.

Per gestire i pacchetti già installati utilizzate semplicemente pkgtool.

Bash scripting

Quanto segue intende essere qualche appunto avanzato di bash scripting che va oltre i fondamentali che mi sono stati impartiti nell'esame di Sistemi Operativi.

Sed

Per poter filtrare le linee di un file di testo o quelle in stdin con opzioni più avanzate di tail e top si usa sed. Qui di seguito mostrerò due esempi tipici di utilizzo.

Filtraggio della seconda linea dello stdin:

```
| sed 2\!d
```

dove si suppone che prima della pipe sia stato dato il comando che produce in stdout le linee che ci interessano.

In questa modalità dopo sed il primo carattere è il numero della linea che ci interessa, poi c'è un ! opzionale preceduto da uno \ che serve per evitare che ! sia mal interpretato dalla bash; lo ! serve per invertire la condizione di selezione. Infine il comando da eseguire che in questo caso è d e sta per delete.

Riassunto il comando vuole dire: considera tutto tranne la seconda linea, e cancellalo.

In questa modalità la sintassi è

```
| sed {address}{!}{command}
```

Dove address può essere il numero della linea oppure una regexp (/regexp/), ! è opzionale, e l'elenco dei command è in man sed, tra i comandi c'è d che sta per delete e p che sta per print.

Da notare che viene sempre stampato a video o in stdout tutto quanto viene processato quindi utilizzare p fa stampare la stringa due volte.

L'altra modalità con cui viene spesso utilizzato sed è la seguente:

```
| sed s/{regexp}/{replacement}/{flag}
```

che significa cerca tutte le linee che contengono regexp, sostituisci regexp con replacement, ed esegui il comando opzionale messo nel campo {flag}, che può essere g, che sta per "applica la sostituzione a tutte le istanze di regexp trovate e non solo alla prima", p, che fa stampare la nuova stringa, e un numero qualsiasi che fa sostituire solo l'istanza numero pari al numero inserito.

Da notare che anche in questo caso viene sempre stampato a video o in stdout tutto quanto viene processato quindi utilizzare p fa stampare la stringa due volte.

Esempio:

```
echo cretino | sed s/cretino/intelligente/g
```

Awk

Awk anche se molto potente viene quasi sempre utilizzato per fare solo una cosa: parsare una singola linea in stdin, suddividerla nelle singole parole che la compongono, stampare solo una di questa parole. Ad esempio:

```
| awk '{print$1}'
```

stampa soltanto la prima parola, con \$2 si stampa la seconda, ecc.
Si può inoltre stampare una parola solo se il numero totale delle parole soddisfa una certa condizione aritmetica, ad esempio:

```
| awk '{if(NR==1)print$1}'
```

dove la variabile NR contiene sempre il numero totale di parole processate.
Una semplice alternativa ad awk è cut:

```
cat /etc/mstab | cut -d ' ' -f1,2
```

Manopolazioni di stringhe con expr, \${} e tr

operazioni matematiche:

```
a=`expr 5 + 3`  
a=$((5 + 3))
```

il secondo modo è MOLTO più veloce.

Per quanto riguarda l'editing sulle stringhe, in generale si possono utilizzare costrutti built-in come \${}, oppure expr, quest'ultimo è più potente ma meno veloce, quindi consiglio di utilizzare in tutti i casi possibili i primi:

```
a=1234zipper43231
```

```
# length: length of string
```

```
b=`expr length $a`  
b=${#a}
```

```
# index: position of first character in substring  
# that matches a character in string  
#echo "Numerical position of first \"2\" in \"$a\" is \"$b\"."
```

```
b=`expr index $a 23`
```

```
# substr: extract substring, starting position & length specified  
#"Substring of \"$a\", starting at position 2,\n#and 6 chars long is \"$b\"."
```

```
b=`expr substr $a 2 6`  
b=${s:2:6}
```

```
# The default behavior of the 'match' operations is to  
#+ search for the specified match at the ***beginning*** of the string.  
#  
# uses Regular Expressions  
#echo Number of digits at the beginning of \"$a\" is $b.
```

```
b=`expr match "$a" '[0-9]*'` # Numerical count.
```

```
echo "The digits at the beginning of \"$a\" are \"$b\"."
```

```
b=`expr match "$a" '\([0-9]*\) '`
```

Test con matching con una regular expression:

```
[[ $a == [0-9]* ]]
```

Nei casi non coperti da `expr` si può utilizzare al suo posto il costrutto `${}`, ecco alcuni esempi:

Extracts substring from `$string` at `$position`.

```
${string:position}
```

Strips shortest match of `$substring` from *front* of `$string`.

```
${string#substring}
```

Replace first match of `$substring` with `$replacement`.

```
${string/substring/replacement}
```

Infine `tr` è utilizzato per traslazioni di caratteri, qualche altro esempio:

```
# Deletes all digits from the file "filename".
```

```
tr -d 0-9 <filename
```

```
#Transforms a file to all uppercase.
```

```
tr a-z A-Z <"$1"
```

Arrays

```
area[1]="ciao"
```

```
echo ${area[1]}
```

```
# Length of first element of array.
```

```
echo ${#array[0]}
```

```
# Number of elements in array.
```

```
echo ${#array[@]}
```

```
arrayZ=( one two three four five five )
```

```
# All elements following element[0].
```

```
echo ${arrayZ[@]:1}
```

```
# two three
```

```
echo ${arrayZ[@]:1:2}
```

```
# Matches "four" and removes it.
```

```
echo ${arrayZ[@]#f*r}
```

```
# one two three four XYZe XYZe
```

```
echo ${arrayZ[@]/fiv/XYZ}
```

TIPS

`` equivale a \$(), il secondo può anche essere innestato

```
for ((i = 0; i < 4; i++))
```

Leggere un file per linee

Capita molto spesso di dover leggere un file una linea alla volta, un metodo per fare questo è il seguente:

```
cat file.lst |while read line; do echo "${line}"; done
```

un altro metodo è utilizzare `exec < file` che ridireziona il file in `stdin`:

```
exec 6<&0          # Link file descriptor #6 with stdin.

exec < data-file  # stdin replaced by file "data-file"

read a1          # Reads first line of file "data-file".
read a2          # Reads second line of file "data-file."

echo
echo "Following lines read from file."
echo "-----"
echo $a1
echo $a2

echo; echo; echo

exec 0<&6 6<&-
# Now restore stdin from fd #6, where it had been saved,
# and close fd #6 ( 6<&- ) to free it for other processes to use.
# <&6 6<&-      also works.
```

Dialog

Come fare per creare programmi con interfaccia grafica utilizzando solo script in bash? Si utilizza `dialog`, sul quale per esempio è basato `pkgtool`.

Il suo utilizzo è abbastanza semplice: basta chiamare `dialog` con un delle opzioni disponibili e il programma stampa a video il box creato e stampa su `stderr` il risultato dell'interazione con l'utente. Utilizzare `man dialog` per vedere tutte le opzioni a disposizione.

CVS

CVS è un sistema per gestire lo sviluppo di un'applicazione da parte di più sviluppatori contemporaneamente in ambiente distribuito.

Per poterlo utilizzare prima di tutto bisogna settare alcune variabili d'ambiente:

```
export \  
  CVSROOT=":pserver:nomeutente@server:/export/cvs/nomads.root"  
export EDITOR=vi
```

la prima serve per definire qual'è il server CVS, il nostro username sul server, e il repository sul server.

La seconda per definire l'editor di testo predefinito che verrà poi richiamato al momento di fare un commit per scrivere il log.

A questo punto basta scegliere una directory sulla macchina locale, entrarci, e poi da lì eseguire:

```
cvs init
```

vi verrà richiesta la password relativa al vostro username.

A questo punto si può scaricare una copia locale di un modulo (ovvero una sottodirectory della directory principale appena inizializzata):

```
cvs checkout {modulo}
```

Una volta effettuate delle modifiche si può renderle definitive pubblicandole sul cvs server col comando:

```
cvs commit
```

che uploada tutte le modifiche di tutti i files presenti nella directory su cui è richiamato e nelle sottodirectory.

Oppure se si vuole committare solo un preciso file:

```
cvs commit nomefile
```

Per scaricare dal cvs server gli aggiornamenti effettuati dagli altri sviluppatori:

```
cvs update
```

Anche in questo caso vengono aggiornati tutti i files presenti nella cartella e nelle sottocartelle da cui questo comando è chiamato.

Se si vuole fare un update sovrascrivendo le proprie modifiche locali eseguire:

```
cvs update -C
```

se si vuole fare un update scaricando anche eventuali nuove sottocartelle eseguire:

```
cvs update -d
```

il sistema cvs cercherà di gestire le problematiche di modifiche contemporanee conflittuali per quanto possibile.

Altri utili comandi disponibili sono:

```
cvs diff {file}
```

per vedere le differenze tra un proprio file locale e quello sul cvs

```
cvs add {file}
```

```
cvs remove {file}
```

per aggiungere o rimuovere files dalla propria copia locale del modulo, le modifiche sono rese definitive anche sul server cvs principale dopo un commit.

Se si vuole far in modo che il file venga automaticamente cancellato anche dal filesystem locale allora bisogna usare:

```
cvs remove -f {file}
```

da notare che non e' possibile cancellare una directory dal cvs, se non manualmente dal server con un grezzo `rm -r`.

Se si vuole aggiungere un file binario allora:

```
cvs add -kb
```

Molto spesso si incontra il problema di voler fare il rollback di una versione di un file presente sul cvs. Il caso più semplice è quello in cui si vuole annullare tutte le modifiche effettuate dall'ultimo commit. Questo è molto semplice da fare e basta cancellare il file in locale e poi fare un `cvs update`. Nel caso invece in cui si voglia tornare ad una specifica "vecchia" versione di un file modificato da se stessi o da altri allora è più complicato, prima di tutto bisogna scegliere la versione a cui si vuole tornare e questo lo si può fare utilizzando il comando:

```
cvs log
```

dopodichè per scaricare tale versione senza creare nuovi branch sul cvs eseguire:

```
cvs update -p -r numversione nomefile > nomefile
```

Per creare diversi branch di un modulo si usa il comando `tag`, che permette di associare un simbolo alla propria versione modificata del modulo:

```
cvs tag {nome}
```

se poi qualche altro sviluppatore vuole scaricarsi la tua versione differente del modulo:

```
cvs checkout -r {nome} {modulo}
```

per fare un merge delle modifiche fatte tra due branch differenti:

```
cvs -d {dir} update -j <merge_tag> -j <branch_tag>
```

Se il cvs server si trova in remoto e ci si vuole connettere tramite ssh per effettuare le operazioni previste bisognerà prestare maggiore attenzione nel definire il cvs repository root. Non è più sufficiente l'absolute path ma è necessario utilizzare una sintassi come segue:

```
:ext:{root}@{81.74.17.71}:{dir}
```

per esempio:

```
:ext:root@127.0.0.1:/root/filent/development/
```

Anche se descritto in questo modo il suo utilizzo può sembrare un po' macchinoso, in tutti gli IDE più moderni, in particolare ho testato personalmente NetBeans, Kdevelop ed Eclipse, permettono di settare alla creazione del progetto il repository e il sistema di gestione delle versioni (CVS o altri), in modo tale che può i vari comandi possano essere dati in maniera più comoda graficamente. Sul repository ci sarà solo la versione definitiva del progetto, quella testata e a cui noi abbiamo dato il commit, in locale nella directory del progetto, solo i files temporanei su cui stiamo lavorando. Infine faccio presente che e' possibile sincronizzare senza problemi due directory cvs su due macchine diverse utilizzando rsync, a patto che gli utenti cvs sulle due macchine siano lo stesso.

Icone di Windows sotto Linux

Sotto linux sostanzialmente ci sono due diversi sistemi di rendering dei fonts: il Core Font System e Xft. Il primo ha più di 15 anni ed è parecchio limitato nelle funzionalità, il secondo invece è più nuovo, supporta l'antialiasing ed è quello di default per ogni applicazione scritta con gtk2 o le qt. Per tutte le altre applicazioni viene invece usato il Core Font System.

Per aggiungere nuovi fonts nel sistema quindi bisogna fare in modo che vengano visti da entrambi i sistemi di rendering.

Per prima cosa bisogna creare una nuova cartella nella directory che contiene tutti i fonts, per esempio:

```
mkdir /usr/X11R6/lib/X11/fonts/TrueType
```

dopodichè si copiano tutti i fonts di Windows li dentro. Si possono trovare in WINDOWS\Fonts. A questo punto bisogna lanciare una serie di comandi per generare dei files necessari al Core Font System:

```
/usr/X11R6/bin/mkfontscale /usr/X11R6/lib/X11/fonts/TrueType/  
/usr/X11R6/bin/mkfontdir /usr/X11R6/lib/X11/fonts/TrueType/  
/usr/X11R6/bin/mkfontdir -e /usr/X11R6/lib/X11/fonts/encodings
```

Infine bisogna editare i file di configurazione per aggiungere la nuova directory. Prima il file di configurazione del Core Font System, che essendo parte di Xfree86 è /etc/X11/XF86Config. Si aggiunge nell'elenco il path della nuova directory:

```
FontPath "/usr/X11R6/lib/X11/fonts/TrueType"
```

e si carica all'avvio il modulo per il rendering dei fonts truetype:

```
Load "freetype"
```

Poi nel file di configurazione /etc/fonts/local.conf, relativo a Xft, si aggiunge:

```
<dir>/usr/X11R6/lib/X11/fonts/TrueType/</dir>
```

a questo punto basta riavviare la X session.

Prima di terminare ricordo però che OpenOffice è un'eccezione e va trattato come tale: ha un proprio sistema di rendering con i propri fonts. Per aggiungere fonts anche a OpenOffice basta copiare i fonts anche in:

```
{/opt/OpenOffice.org1.1.1}/share/fonts/truetype
```

Video Editing

In questa sezione verranno esposte le tecniche a disposizione degli utenti linux per editare video, rippare dvd, ecc.

I Tools

Prima di tutto una carrellata sugli strumenti a nostra disposizione:

vobcopy

programma utile per estrarre tutti i vob da dvd e anche per fare una copia mirror di un dvd sull'hdd. Come output c'è una directory con il nome del dvdvideo contenente i files che ci interessano. Per esempio per effettuare una copia mirror:

```
vobcopy -m -i {/dev/dvd} -o {/mnt/winarchive/tmp}
```

dvdbackup

programma equivalente al precedente, forse meno stabile ma ha il vantaggio che permette di stabilire a priori il nome della directory di output per il mirror del dvd. Per esempio per effettuare una copia mirror:

```
dvdbackup -M -i {/dev/dvd} -o {/mnt/winarchive/tmp} -n backup
```

avidemux2

programma grafico molto potente per l'editing di video. Permette di tagliare un video, unire più video, ricodificare audio e/o video, rencodare l'audio e/o video, filtrare l'audio e/o video. Se compilato manualmente permette di generare anche un'interfaccia testuale che potrebbe tornare utile.

mencoder

programma da linea di comando per l'encoding e il decoding video anche da dvd o tv. Ha il vantaggio di essere veloce efficiente e, nonostante l'interfaccia testuale, di semplice utilizzo. Per esempio per rippare una traccia da dvd senza encodarla:

```
mencoder -dvd-device {/dev/dvd} dvd://{numero traccia} -dvdangle  
{numero angolo} -alang {it} -chapter {2-0} -ovc copy -oac copy -o  
{file.avi}
```

-chapter serve per indicare quali capitoli estrarre. Per esempio 12-20 estrae tutti i capitoli da 12 al 20. 2-0 invece estrae tutti i capitoli da secondo alla fine del dvd.

Invece per rippare in mp3 solo l'audio di una traccia di un dvd e lasciare il video così com'è:

```
mencoder -dvd-device {/dev/dvd} dvd://{numero traccia} -dvdangle  
{numero angolo} -alang {it} -chapter {2-0} -ovc frameno -oac  
mp3lame -lameopts abr:br={bitrate}:q=2:vol=8 -srate 44100 -o  
{file.avi}
```

per rippare il video da un vob sull'hdd in xvid a 2 passi e lasciare l'audio intatto:

```
mencoder -nosound -o /dev/null -ovc xvid -xvidencopts
bitrate=1000:me_quality=6:4mv:pass=1:vhq=4 -vop
scale=640:480,crop=638:368:29:92,pp=0x20000 {/space/film.vob}
```

```
mencoder -oac copy -o {ally.avi} -ovc xvid -xvidencopts
bitrate=1000:me_quality=6:4mv:pass=2:vhq=2 -vop
scale=640:480,crop=638:368:29:92,pp=0x20000 -mc 1
0{space/film.vob}
```

crop nelle opzioni server per tagliare le bande nere ai lati del file e i valori sono solo di esempio. Infine un esempio dell'utilizzo di mencoder per registrare da tv in MPEG4 sfruttando i velocissimi lavcodecs:

```
mencoder tv:// -tv
driver=v4l:input=2:brightness=-10:contrast=40:width=640:height=480
:forceaudio:adevice=/dev/dsp:amode=1:audiorate=44100 -ovc lavc
-lavcopts vbitrate=2200:
keyint=250:vqscale=2:vrc_buf_size=400000 -vf lavcdeint -oac pcm
-endpos 01:00:00 -o /root/video/$1
```

dove **tv://** è la sorgente e **-tv** precede tutte le opzioni specifiche per trattarla:

driver=v4l sta ad indicare di utilizzare Video4Linux per accedere ai dati;

input=2 specifica di utilizzare l'entrata video composita;

brightness e **contrast** servono per sistemare luce e colori;

width e **height** servono per la dimensione dei frames;

forceaudio specifica di registrare anche l'audio;

adevice=/dev/dsp specifica di utilizzare la scheda sonora come sorgente audio;

audiorate=44100 è il rate audio: 44100 Khz.

-ovc lavc specifica di utilizzare i codecs lavc e **-lavcopts** precede tutte le opzioni del codec:

vbitrate=2200 è il bitrate;

keyint=250 significa che c'è un keyframe ogni 250 frames. I keyframe sono quelli ai quali si può saltare manualmente durante la riproduzione;

vqscale=2 specifica di tenere la qualità del quantizzatore costante e al massimo (grande dimensione del file di output);

vrc_buf_size=400000 specifica la dimensione del buffer.

-vf lavcdeint aggiunge un filtro al volo per il deinterlacciamento;

-oac pcm specifica di tenere l'audio non compresso in pcm;

-endpos serve per specificare la durata di registrazione e infine **-o** precede il nome del file di output.

transode

altro programma a linea di comando per ripping, encoding, decoding, ecc.

E' veramente molto potente anche se molto difficile da utilizzare, non è un programma solo ma tutto un insieme di tool.

Giusto per dare un'idea di come funziona e di quali sono le sue potenzialità nomino alcuni casi di utilizzo e singoli tool.

Tcprobe e lsdvd da utili informazioni sul dvd.

```
tcprobe -i {/dev/dvd} &>&1
lsdvd -i {dev/dvd}
```

Tccat serve per estrarre da un dvd una traccia, la esporta in stdout:

```
tccat -i {/dev/dvd} -T {1,-1} |
```

il T specifica quale traccia; 1,-1 sta per traccia 1 e tutti i suoi capitoli.

Tcextract server per estrarre l'audio o il video da un file video stdin, tipicamente un vob, e lo encoda:

```
| tcextract -t vob -x mp3 -a 1 > {file.mp3}
tcextract -i {video.vob} -t vob -x mpeg2 > {file.m2v}
```

dove -a 1 sta per traccia audio 1 e -i serve per prendere come input un file anziché stdin.

Transcode è invece l'interfaccia testuale omnicomprensiva che racchiude tutte le funzionalità. Per estrarre una traccia da dvd e encodarne l'audio soltanto:

```
transcode -i {/dev/dvd} -x dvd -T 1,-1 -y null,lame -b 192 -o
{file.mp3}
```

dove -x specifica il codec utilizzato per interpretare l'input, -x dvd sta ad indicare che l'input è interpretato come un dvd. -T è la solita opzione per specificare traccia e capitoli e -i la sorgente.

-y serve per specificare i codec da utilizzare per codificare l'uscita; il primo è per il video, in questo null, e il secondo per l'audio. -b serve per specificare il bitrate di uscita dell'audio.

Per ricodificare il video di un dvd in maniera più compressa:

```
transcode -i {/dev/dvd} -x dvd -T 1,-1 -P 1 -y raw -w {1.5} -o
{file.mp3}
```

dove -P 1 specifica che si effettua un pass-through del video, -y raw è il tipo di codifica necessaria per queste cose, e -w specifica il fattore di compressione scelto.

Anche in questo caso termino con un esempio completo di utilizzo di transcode per registrare da tv tramite l'interfaccia Video4Linux in MPEG4 tramite Xvid:

```
transcode -i /dev/video0 --import_v4l 2 -p /dev/dsp -x v4l -g
384x288 -e 44100,16,2 -M 0 -n 0x01 -y xvid4,raw -Q 4 -w 1700 -N
0x01 -o prova.avi -V -u 500 -q 2
```

anche in questo caso -i specifica la sorgente, **--import_v4l 2** serve per selezionare l'entrata video composita ed è un'opzione specifica di questo tipo di sorgente;

-p /dev/dsp sta ad indicare di prelevare l'audio da /dev/dsp;

-x v4l è il codec necessario per decodificare l'entrata;

-g 384x288 è la dimensione dei frames;

-e 44100,16,2 la qualità della sorgente audio da utilizzare;

-n 0x01 seleziona il formato di entrata dell'audio;

-y xvid4,raw seleziona il codec xvid per il video e specifica di lasciare non compresso l'audio;

-Q 4 è la qualità di codifica, in questo caso 4/5;

-w 1700 è il bitrate in kbit/sec;

-N 0x01 il formato audio di uscita, in questo caso PCM;

-o precede il file di output;

-V seleziona il codec interno da utilizzare per i frame decompressi;

-u 500 è la quantità di frames che possono stare nel buffer;

-q 2 un'opzione di debug.

dvd::rip

dvdrrip è una comodissima gui d'interfaccia a transcode, che facilita e ottimizza tutte le funzionalità per rippare un dvd in un divx o mpeg. Permette di scegliere le tracce da rippare, di fare cropping visuale, rippare sottotitoli, configurare codec video in base alla dimensione del file d'uscita, configurare codec audio, encodare più tracce audio differenti nello stesso avi, e anche generare anteprime.

Utilizzando dvdrrip, la prima cosa da fare è scegliere la traccia che ci interessa rippare e darci un'occhiata, per vedere se è interlacciata, ovvero se si vedono delle linee orizzontali vicino ai margini delle figure. In questo caso bisogna ricordarsi poi di mettere "Smart Deinterlacing" in Deinterlace mode nella slide Transcode.

Nella sezione clip&zoom bisogna cercare di mantenere la dimensione maggiore possibile di output dei frame, in modo da non essere costretti ad ingradirli troppo successivamente durante la riproduzione. Solitamente Big Frame Size è la dimensione corretta.

Nella sezione transcode ricordarsi che utilizzando le audio options opportunamente si possono tenere più tracce audio contemporaneamente nell'avi finale. Le General Options servono per le preview o per passare manualmente parametri a transcode: nel caso delle preview basta selezionare un frame di inizio e uno di fine in Frame range per limitare la sezione di encoding, per ottenere i valori basta tenere presente il video framerate che solitamente è 25 frame al secondo, quindi basta fare un piccolo calcolo per scegliere il secondo di inizio e quello di fine della preview.

Può tornare utile anche passare manualmente parametri a transcode, in particolare per migliorare la sincronizzazione audio\video: basta porre in transcode options "-D {valore}" per cambiare la sincronizzazione; il valore è in millisecondi e può essere anche negativo.

streamdvd

streamdvd è un programma a linea di comando per effettuare il transcoding di una traccia di un dvd con un singolo audio e senza sottotitoli:

```
dvdauthor -t -o {directory} -f "streamdvd -i {/dev/dvd} -t {1} -f {1.5} |"
```

nell'esempio viene utilizzato insieme a dvdauthor per generare una nuova traccia dvd con una maggiore compressione video. -f specificare il fattore di compressione e sta tra 1 e 2, dove 2 imezza la dimensione del video, e 1 la lascia intatta. Oltre 1.5 la perdita di qualità diventa consistente.

streamanalyze

programma a linea di comando che da utili informazioni su una traccia di un dvd calcolando anche il fattore di compressione necessario da dare a streamdvd per encodarla da sola con solo una traccia audio in un unico dvdr.

```
streamanalyze -i {/dev/dvd} -t {1}
```

dvdauthor

è il programma che fa authoring sotto linux, ovvero dato un video lo mette nel formato adeguato per essere masterizzato come dvd video.

In unione con streamdvd può essere utilizzato così:

```
dvdauthor -t -o {directory} -f "streamdvd -i {/dev/dvd} -t {1} -f
```

```
{1.5} |"
```

-t significa genera una traccia, -o l'output, -f il file da aggiungere.

Può essere anche utilizzato tramite la gui QdvdAuthor oppure tramite dei file xml di configurazione.

subtleripper

subtleripper è un piccolo insieme di programmi per convertire i sottotitoli rippati da dvd in un formato testuale leggibile da mplayer. Utilizza pesantemente transcode.

Utilizzarlo è facile: prima bisogna scegliere la traccia sottotitoli da rippare, lo si può fare con mplayer:

```
mplayer -dvd-device {/dev/dvd} dvd://1 {1} -vo null -ao null  
-frames 0 -v 2>&1 | grep sid
```

questo comando da come output un elenco di sottotitoli disponibili per la traccia del dvd scelta.

A questo punto rippiamo i sottotitoli scelti utilizzando transcode:

```
tccat -i {/dev/dvd} -T {1} -L | tcextract -x ps1 -t vob -a 0x22 >  
{file}
```

dove 0x22 è il sottotitolo scelto. Il numero si ricava in questo modo: 0x20 + il numero della traccia sottotitolo. Se il suo numero visualizzato da mplayer poco prima è per esempio 1, allora risulta 0x21.

Ora la conversione vera e propria in formato testuale tramite subtleripper:

```
subtitle2pgm -o {sottotitoli} < {file}
```

```
pgm2txt {sottotitoli}
```

```
srttool -s -w < {sottotitoli.srtx} > {sottotitoli.srt}
```

tutte queste operazioni posso semplicemente essere automatizzate utilizzando il mio script substabe.

lxdvdrip

Programma equivalente al famoso dvdshrink per windows, utilizza i vari tools visti in precedenza (transcode, mplayer, streamdvd, ecc.) per effettuare la copia di un dvd9 in un dvd5 effettuando il transcoding automatico delle tracce video.

Xdvdshrink

Anche questo programma permette di effettuare copie di dvd9 su dvd5 effettuando il transcoding delle tracce video.

DVD Backup

Per fare una copia di backup di un dvd abbiamo principalmente tre possibilità differenti di operare.

Prima di tutto possiamo ripparlo in un divx o più divx nel caso ci siano anche contenuti

speciali\extra: in questo caso non c'è niente di meglio che dvd::rip. E' comodo completo ed efficace

e considerando che permette di croppare, zoomare e mantenere doppio audio è una soluzione perfetta.

Se intendiamo rippare anche i sottotitoli utilizziamo invece subtitleripper magari tramite lo script substabe, poiché dvdrip gestisce i sottotitoli renderizzandoli direttamente sulle immagini, e non mantenendoli a parte.

Possiamo invece voler doppiare il dvd in un altro dvd per tenere anche i menu ed avere una copia più completa, purtroppo però la maggior parte dei dvd video in circolazione sono dvd9, nel senso che sono da 9 Gb, invece quelli da masterizzare, i dvd-r, sono solo da 4,7 Gb.

Quindi se siamo fortunati e il dvd video occupa meno di 4,7Gb allora possiamo utilizzare vobcopy o dvdbackup per decriptare il dvd e copiarlo sull'hdd e poi semplicemente k3b per masterizzarlo.

Se invece occupa poco più di 4,7 Gb possiamo scegliere di mantenerlo intatto in tutti i suoi contenuti e anche i menu e comprimere un po' di più le tracce video in modo farle stare in 4.7 Gb.

In questo caso possiamo utilizzare dvdshrink in emulazione tramite wine, oppure utilizzare una soluzione “tutta linux” con lxdvdrip oppure xdvdshrink.

Infine possiamo voler mantenere tutto il dvd com'è, non comprimelo ulteriormente ma splittarlo in 2. In questo caso si può utilizzare dvd2iso in emulazione con wine oppure ancora una volta lxdvdrip.

Editing\Modifica video

Se abbiamo bisogno di aggiustare un divx o un file video in generale, la sincronizzazione audio\video, passare ad un altro formato, ritagiarlo, accorciarlo, allungarlo, ecc. la cosa migliore è utilizzare il potentissimo Avidemux2.

File di Grandi Dimensioni

Se abbiamo a che fare con files di grosse dimensioni, maggiori di 2GB per intenderci, abbiamo primaditutto necessità di un filesystem che li supporti, quindi niente fat32 ma ext2, ext3, ecc.

Inoltre dobbiamo affrontare il fatto che il vecchio formato AVI incontra grosse difficoltà dai 2GB in su, e queste aumentano sopra i 4GB: sono pochi i programmi che possono gestire avi sopra i 2GB, e nessuno comunque riesce a scrivere l'index block di questi file, ovvero l'indice che serve ai programmi che riproducono video a effettuare il seeking in modo corretto.

Per questo è stato creato il formato AVI 2.0, chiamato anche OpenDML dal nome dei suoi creatori; necessitiamo quindi di software che supporta questo nuovo standard: Mencoder dalla versione 1.0pre5 in su e Transcode dalla 0.6.12 in su per encodare, Avidemux2 dalla versione 2.0.26 in su per editare e Xine dalla versione 1-rc3 in su per riprodurre.

Utilizzando questi programmi dovremmo evitarci molti grattacapi.

Jboss + Nukes

Jboss è l'Application Server Open Source per eccellenza, Nukes è un CMS sviluppato sempre dalla Jboss e gira proprio all'interno del loro server.

Installazione

L'installazione è molto semplice, basta scaricare il pacchetto preconfigurato di Jboss+Tomcat+Nukes tutto compreso, il quale farà girare Jboss più una versione embedded di Tomcat, con Nukes naturalmente già correttamente installato.

Altrimenti è anche molto facile scaricare Jboss e Nukes a parte, installare Nukes su Jboss e magari appoggiarsi ad un database esterno come Mysql, basta seguire le istruzioni dell'installazione di Nukes. Anche questa soluzione si basa su una versione di Tomcat embedded in Jboss.

Risultano problematiche invece due cose:

- utilizzare la versione di Tomcat già compresa in Jboss per fare girare altri siti, perchè la sua configurazione non è affatto facile e inoltre non è stato pensato per essere usato senza Jboss.
- cercare di utilizzare una versione separata esterna di Tomcat per far girare Nukes. In questo caso è semplice riuscire a non far eseguire la versione embedded di Tomcat ed invece sfruttare una esterna la quale possa connettersi a Jboss per la logica delle applicazioni. Il problema risulta riuscire a "suddividere" Nukes tra il Tomcat esterno e Jboss, poiché viene rilasciato in un formato adatto ad essere eseguito completamente in Jboss. Per altre web applications invece nessun problema.

Aggiungere Moduli e Blocchi

Una volta installato Jboss, Tomcat, un DBMS e Nukes, il problema è come aggiungere altri moduli o blocchi, magari utilizzando un IDE come NetBeans.

Prima naturalmente si crea un nuovo progetto con NetBeans, poi è necessario importare alcuni jar nel progetto in modo da poter importare poi nelle classi da noi create le corrette classi java di Jboss e Nukes.

I jar che servono sono:

```
jboss-common.jar      $JBOSS_HOME/lib
jboss-system.jar      $JBOSS_HOME/lib
jbossmx-ant.jar       $JBOSS_HOME/client
nukes-lib.jar
```

l'ultimo jar è invece presente all'interno di un altro archivio zip chiamato **nukes.ear** presente in **\$JBOSS_HOME/server/default/depoly**.

Importati questi jar nel progetto di NetBeans sarà possibile aggiungere senza problemi i seguenti import nelle classi da creare, necessari per utilizzare le api di jboss e nukes:

```
import org.jboss.nukes.block.*;
import org.jboss.nukes.module.*;
import org.jboss.nukes.*;
import org.jboss.nukes.html.*;
import org.jboss.*;
```

fatto questo possiamo a vedere qual'è il modello tipico per la creazione di un modulo:

```
public class StupidModule extends ModuleSupport
```

```

{

public StupidModule()
{
    super("stupid");
}

public void provametodo(Page page)
{
    Api api = getApi();
    String stronzo = page.getParameter("pippo");
    if(pippo!=null) page.print("Pippo è alto "+pippo+" cm");
}

}

```

da notare il costruttore che chiama `super`, con il nome del modulo, un metodo qualsiasi che verrà poi chiamato dall'esterno che reagisce utilizzando **`page.getParameter`** per leggere i parametri passati dall'utente e **`page.print`** per restituire l'html in uscita.

Per poter aggiungere questo modulo a Nukes è necessario porlo nel corretto package, ovvero:

```
package org.jboss.nukes.core.modules;
```

quindi è necessario anche ricostruire correttamente l'albero delle directory all'interno della cartella del progetto di netbeans:

```
{~/ingegneria/development/fantasy}/org/jboss/nukes/core/modules/StupidModule.class
```

creata correttamente la classe si deve scrivere un file xml di questo tipo:

```

<?xml version="1.0" encoding="UTF-8"?>
<mbean
  code="org.jboss.nukes.core.modules.{StupidModule}"
  name="nukes.modules:name={stupid}"
  xmbean-dd=""
  xmbean-code="org.jboss.nukes.component.NukesMBean">
  <xmbean>
    <depends>nukes.modules:name=core</depends>
    <attribute name="DisplayName">{Stupid} module</attribute>
    <attribute name="Description">{Stupid module is only my
first try}</attribute>
  </xmbean>
</mbean>

```

e salvarlo con nome **`jboss-service.xml`**.

Per finire creiamo un archivio zip con all'interno la nostra classe posta con tutte le directorys che la contengono a partire da `org`, e un'altra directory chiamata **`META-INF`** con dentro il file xml appena creato. Diamo come nome all'archivio:

```
{quellochevuoi}.sar
```

e lo poniamo in **\$JBOSS_HOME/server/default/nuke**s. Jboss si occuperà di caricarlo a caldo. Allo stesso modo per creare un blocco personalizzato scriviamo la classe:

```
package org.jboss.nukes.core.blocks;

import org.jboss.nukes.block.*;
import org.jboss.nukes.module.*;
import org.jboss.nukes.*;
import org.jboss.nukes.html.*;
import org.jboss.*;

public class StupidBlock extends BlockSupport
{

    public StupidBlock()
    {
        // provide a name for the block
        super("stupid");
    }

    public boolean getDisplayed(Page page)
    {
        // displays the block only if the user is logged in
        return getApi().userLoggedIn();
    }

    public void renderContent(Page page)
    {
        page.print("SONO MOLTO IDIOTA");
    }

    public void render(Page page){
        page.print("SONO MOLTO IDIOTA");
    }

}
```

posta all'interno della cartella del progetto di netbeans in **org/jboss/nuke**s/core/blocks.

GetDisplayed è il metodo che serve a jboss per capire se il modulo va fatto vedere nella parte pubblica o nella parte riservata, **render** e **renderContent** servono invece per restituire in output il codice html per la visualizzazione del blocco.

A questo punto si scrive il file xml **jboss-service.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>
<server>
  <mbean
    code="org.jboss.nukes.core.blocks.{StupidBlock}"
    name="nuke.s.blocks:name={stupid}"
    xmbean-dd=""
    xmbean-code="org.jboss.nukes.component.NukesMBean">
    <depends>nuke.s.modules:name=core</depends>
```

```
<xmlbean>
  <attribute name="Title">{Stupid} block</attribute>
  <attribute name="Side">{0}</attribute>
  <attribute name="Weight">{3}</attribute>
</xmlbean>
</mbean>
</server>
```

e lo si pone nella cartella **META-INF**.
Si crea l'archivio prova.sar con all'interno:

```
META-INF
  jboss-service.xml
org/jboss/nukes/core/blocks
  StupidBlock.class
```

e si pone tale archivio in **\$JBOSS_HOME/server/default/nukes**.

Patch

Le patch sono piccoli files che contengono delle correzioni per il codice di un programma.

Patchare un file

Patchare un file significa quindi applicare tali correzioni al codice e si fa andando nella directory col codice e:

```
patch -p0 < {patch file}
```

ed eventualmente indicando su richiesta il nome del file da patchare.

Creare una patch

Per creare una patch invece bisogna utilizzare l'utility diff, che confronta due files di testo e ne stampa le differenze:

```
diff -c file-old file-new > patch_file
```

Abilitare il supporto SMP nel Kernel

Se si ha a disposizione un sistema multiprocessore o semplicemente un Pentium 4 HT (Hyper-Threading), per ottenere il massimo dal proprio sistema è necessario abilitare il supporto multiprocessore nel kernel, mai abilitato di default nella Slackware.

Nel caso del Pentium 4 HT anche se il processore è uno solo, siccome è in grado di eseguire 2 thread in parallelo, viene visto dal sistema come un processore dual-core; quindi se non si abilita l'SMP nel kernel l'Hyper-Threading sostanzialmente non viene utilizzato.

Data la configurazione iniziale di default del kernel su Slack, è necessario operare alcune piccole modifiche:

In "Processor type and features":

scegliere la propria corretta famiglia di processori, abilitare High Memory Support se si possiede più di 1GB di RAM, abilitare MTRR e Simmetric multi-processing support;

in "Character Devices":

assicurarsi che sia abilitato Enhanced Real Time Clock;

A questo punto bisogna ricompilare il kernel e anche tutti i moduli; visto che probabilmente si vorrà tenere anche una versione originale del kernel della stessa versione, bisogna editare il Makefile ponendo:

```
EXTRAVERSION = {SMP}
```

a questo punto procedere normalmente con:

```
make dep
make bzImage
make modules
make modules_install
```

e aggiornando correttamente lilo.

A volte capita che il modulo acpi nel kernel funzioni male in presenza di un processore HT, quindi bisogna passare come argomento al kernel `acpi=ht`, aggiungendo a `lilo.conf` la linea:

```
append="acpi=ht"
```

Importare kernel da altre distribuzioni su Slackware

Supponiamo di voler importare su Slackware il kernel utilizzato dalla nuovissima Knoppix, che abbiamo provato da cd, e abbiamo scoperto che rileva perfettamente tutto il nostro hardware.

Utilizzando il cd Knoppix carichiamo questa distribuzione, da lì montiamo la partizione di root del nostro harddisk, dov'è installata la nostra Slackware. A questo punto leggiamo /etc/lilo.conf per vedere qual'è l'immagine del kernel che utilizza, di solito è /boot/vmlinuz, ma non si sa mai, poi copiamo il file a cui punta vmlinuz (che è quasi sempre un link simbolico) nella /boot della Slackware stando attenti a non sovrascrivere altri file.

Supponiamo che la versione del kernel sia 2.6.11 (uname -a per scoprirlo), copiamo i moduli sulla Slackware con:

```
cp -R /lib/modules/2.6.11 /mnt/hda7/lib/modules
```

dove /mnt/hda7 è il punto di mount della root della Slackware.

Fatto questo rimane solo da copiare i sorgenti:

```
cp -R /usr/src/linux-2.6.11 /mnt/hda7/usr/src
```

Ora non dobbiamo fare altro che riavviare togliendo il cd della Knoppix, aggiungere il nuovo kernel al lilo.conf della Slackware e rieseguire lilo.

A volte però al successivo riavvio il sistema non parte ma va in kernel panic, per esempio perchè manca il supporto nel kernel al filesystem ReiserFS e il filesystem di root della nostra Slackware è proprio ReiserFS (potrebbe succedere la stessa cosa anche con i drivers SCSI o SATA), allora dobbiamo creare con mkinitrd un piccolo filesystem da caricare in RAM durante la fase di boot con i moduli che servono al kernel per avviare il sistema, per esempio:

```
mkinitrd -c -k 2.6.11 -m reiserfs:ext3 -f reiserfs -r /dev/hda7
```

dove -k specifica la versione del kernel e dei moduli da utilizzare, -m specifica la lista dei moduli da inserire, -f specifica il filesystem di root e -r la partizione di root.

Mkinitrd produce **/boot/initrd.gz** a questo punto basta solamente aggiungere a lilo.conf la linea:

```
initrd=/boot/initrd.gz
```

e naturalmente rieseguire lilo.

Infine a volte si può aver bisogno anche dei sorgenti veri e propri del kernel, per esempio per compilare i drivers dell'Nvidia, in questo caso basta scaricare in /usr/src da kernel.org i sorgenti della versione del kernel coincidente con quella della Knoppix, dopodichè entriamo in /usr/src e:

```
mv linux-2.6.11 linux-knoppix
```

per non sovrascrivere quelli compilati della Knoppix, poi:

```
tar xvjf linux-2.6.11.tar.bz2
rm -r linux-2.6.11/include
rm -r linux-2.6.11/arch
cd linux-knoppix
cp * /usr/src/linux-2.6.11
cp -R arch /usr/src/linux-2.6.11
cp -R include /usr/src/linux-2.6.11
```

Modificare /etc/fstab

/etc/fstab è il file di configurazione che descrive tutte le partizioni presenti nel sistema e specifica come devono essere montate.

Un esempio è il seguente:

```
/dev/hda6      swap          swap          defaults      0
0
/dev/hda7      /             ext3          defaults      1
1
/dev/hda1      /mnt/win      ntfs          ro            1
0
/dev/hda5      /mnt/winarchive ext3          defaults      1
0
/dev/sr0       /mnt/cdrom    iso9660       noauto,owner,ro 0
0
```

ogni linea è del tipo:

```
{file descriptor} {directory da montare} {filesystem} {opzioni}
{boolean per dump} {boolean per fsck}
```

dove gli ultimi due valori booleani non ci interessano, mentre sono importanti le opzioni, il cui elenco completo si trova con **man mount**.

Alcune opzioni importanti sono:

auto\noauto – specificano se il filesystem può essere montato tramite l'opzione -a

nouser\users – specificano che anche gli utenti possono montare o smontare il filesystem o solo root

ro\rw – specificano se il filesystem viene montato read-only o read-write

exec\noexec – permette l'esecuzione di binari

dev\nodev - interpreta i character devices

suid\nosuid – abilita gli effetti di suid

umask=*** - permette di settare l'umask nel caso di filesystem fat o ntfs

defaults equivale a: rw, suid, dev, exec, auto, nouser, and async

Slackware di default è molto restrittiva con i diritti di accesso lasciati agli utenti comuni, se si vuole permettere ad un'utente qualsiasi di poter smontare o montare un fs, oppure potervi semplicemente accedere (nel caso di fat o ntfs) occorre cambiare le opzioni come segue:

```
auto,users,rw,umask=002
```

nel caso di ntfs (che è read-only) invece:

```
auto,users,ro,umask=222
```

Slackware: il sistema di pacchetti

Creare un pacchetto

La procedura corretta per creare un pacchetto Slackware tgz è la seguente:

-si scompattano i sorgenti

-nella directory contenete i sorgenti si lancia il comando configure, con gli opportuni parametri, in modo tale che la successiva compilazione sia ottimizzata per un 486, e in modo tale che ogni parte del programma vada installata nella directory giusta.

```
CFLAGS="-O2 -march=i486 -mcpu=i686" ./configure --prefix=/usr -
sysconfdir=/etc - localstatedir=/var/lib
```

-si lancia finalmente il comando make, per compilare il tutto

-si installa il pacchetto in una directory temporanea:

```
make install DESTDIR={/tmp/pacchettotmp}
```

-infine copiano i files doc informativi e si strippano binari e librerie

```
( cd $PKG
  find . | xargs file | grep "executable" | grep ELF | cut -f 1 -d
: | xargs strip --strip-unneeded 2> /dev/null
  find . | xargs file | grep "shared object" | grep ELF | cut -f 1
-d : | xargs strip --strip-unneeded 2> /dev/null
)
mkdir -p $PKG/usr/doc/{k3b}-$VERSION
cp -a \
  AUTHORS COPYING FAQ INSTALL README TODO \
  $PKG/usr/doc/{k3b}-$VERSION
```

-Ora che abbiamo sostanzialmente il pacchetto completo creiamo in DESTDIR la directory **install**, nella quale creiamo se necessario lo script in bash **doint.sh**, che viene eseguito durante l'installazione per effettuare qualche modifica necessaria al corretto funzionamento del programma. Sempre in \$PKG/install creiamo il file **slack-desc** con la descrizione del pacchetto. Il file slack-desc deve essere del tipo:

```
# HOW TO EDIT THIS FILE:
# The "handy ruler" below makes it easier to edit a package
# description. Line
# up the first '|' above the ':' following the base package name,
# and the '|' on
# the right side marks the last column you can put a character in.
# You must make
# exactly 11 lines for the formatting to be correct. It's also
# customary to
# leave one space after the ':'.

|-----handy-
ruler-----|
xcdroast: xcdroast (Graphical frontend for cdrtools)
```

```
xcdroast:
xcdroast: X-CD-Roast is a graphical frontend for the command line
cdrtools.
xcdroast: You can do anything X-CD-Roast does yourself using the
command line
xcdroast: tools - but it's nicer and easier with the frontend.
The cdrtools
xcdroast: contain "cdrecord" (which does the actual writing of
CDs), "readcd"
xcdroast: (reads data tracks of CDs), "mkisofs" (masters CD
images), and
xcdroast: "cdda2wav" (reads audio tracks). Cdrecord, readcd and
mkisofs are
xcdroast: maintained by Joerg Schilling, cdda2wav by Heiko
Eissfeldt, and
xcdroast: X-CD-Roast by Thomas Niederreiter.
xcdroast:
```

Creiamo, se necessario, il file **slack-required**, nel quale precisiamo se ci sono delle dipendenze particolari per quel pacchetto. Questo file deve essere del tipo:

```
flex >= 2.5.4
bison >= 1.875
python >= 2.3
perl >= 5.8.0
tcl >= 8.4.4
```

-Ora non manca altro che creare il pacchetto vero e proprio, e questo lo facciamo eseguendo il comando `makepkg` nella directory `DESTDIR`:

```
makepkg -l y -p y -c n {xcdroast-$VERSION-$ARCH-$BUILD.tgz}
```

A parte i primi 3 passi, i seguenti possono essere eseguiti in automatico utilizzando `checkinstall`, che è molto comodo ma non crea sempre pacchetti abbastanza corretti da essere utilizzati su più sistemi differenti, lo scopo di `checkinstall` più che altro è quello di aggiungere il software come pacchetto in modo tale che successivamente possa essere disinstallato più comodamente.

Un esempio di SlackBuild

In ogni caso possiamo crearci uno scriptino noi che compia tutte le operazioni più ripetitive per noi, quello che segue è un esempio di SlackBuild di Patrick per `k3b`:

```
#!/bin/sh
CWD=`pwd`
TMP=${TMP:-/tmp}
PKG=${TMP}/package-k3b

VERSION=0.11.11
ARCH=${ARCH:-i486}
BUILD=1
```

```

if [ "$ARCH" = "i386" ]; then
    SLKCFLAGS="-O2 -march=i386 -mcpu=i686"
elif [ "$ARCH" = "i486" ]; then
    SLKCFLAGS="-O2 -march=i486 -mcpu=i686"
elif [ "$ARCH" = "s390" ]; then
    SLKCFLAGS="-O2"
elif [ "$ARCH" = "x86_64" ]; then
    SLKCFLAGS="-O2"
fi

if [ ! -d $TMP ]; then
    mkdir -p $TMP
fi
rm -rf $PKG
mkdir -p $PKG
cd $TMP
rm -rf k3b-$VERSION
tar xjvf $CWD/k3b-$VERSION.tar.bz2
cd k3b-$VERSION
chown -R root.root .
find . -perm 777 -exec chmod 755 {} \;
find . -perm 775 -exec chmod 755 {} \;
find . -perm 700 -exec chmod 755 {} \;
find . -perm 744 -exec chmod 755 {} \;
find . -perm 666 -exec chmod 644 {} \;
find . -perm 664 -exec chmod 644 {} \;
find . -perm 600 -exec chmod 644 {} \;
CFLAGS="$SLKCFLAGS" \
./configure \
    --prefix=/opt/kde \
    --program-prefix="" \
    --program-suffix="" \
    $ARCH-slackware-linux
make -j4
make install DESTDIR=$PKG
( cd $PKG
  find . | xargs file | grep "executable" | grep ELF | cut -f 1 -d
: | xargs strip --strip-unneeded 2> /dev/null
  find . | xargs file | grep "shared object" | grep ELF | cut -f 1
-d : | xargs strip --strip-unneeded 2> /dev/null
)
mkdir -p $PKG/usr/doc/k3b-$VERSION
cp -a \
    AUTHORS COPYING FAQ INSTALL README TODO k3b.lsm \
    $PKG/usr/doc/k3b-$VERSION
mkdir -p $PKG/install
cat $CWD/slack-desc > $PKG/install/slack-desc

cd $PKG
makepkg -l y -c n $TMP/k3b-$VERSION-$ARCH-$BUILD.tgz

```

Come distribuire i propri pacchetti

La Slackware ha un proprio sistema consolidato per organizzare i pacchetti da distribuire, che è usato sia per le ISO ufficiali, che per i Repository di Swaret.

Vi è per incominciare una directory radice, che è quella alla quale punta il sito web nel caso di una repository, per esempio la directory extra nel terzo cd Slackware.

All'interno della radice vi è poi una sottodirectory per ogni applicazione o gruppo di applicazioni chiamata col nome di tale pacchetto che contiene, per esempio:

```
drwxr-xr-x    7 root root    2048 2003-02-28 01:54 3dfx-glide/
-rw-r--r--    1 root root   32548 2004-06-22 08:58 CHECKSUMS.md5
-rw-r--r--    1 root root    189 2004-06-22 08:58 CHECKSUMS.md5.asc
-rw-r--r--    1 root root   49362 2004-06-22 08:58 FILE_LIST
-rw-r--r--    1 root root  202061 2004-06-22 08:58 MANIFEST.bz2
-rw-r--r--    1 root root   44698 2004-06-22 08:35 PACKAGES.TXT
-rw-r--r--    1 root root    149 2002-02-09 00:18 README.TXT
drwxr-xr-x    3 root root    6144 2004-06-22 08:53 aspell-word-
lists/
drwxr-xr-x    2 root root    2048 2004-06-16 01:30 bash-completion/
drwxr-xr-x    2 root root    2048 2003-02-28 01:54 bison-1.875/
drwxr-xr-x    2 root root    2048 2004-05-15 22:18 bittorrent/
drwxr-xr-x    2 root root    2048 2004-05-25 05:04 brltty/
drwxr-xr-x    2 root root    2048 2003-03-16 11:06 btmgr-3.7/
drwxr-xr-x    2 root root    2048 2004-03-31 06:57 checkinstall/
drwxr-xr-x    2 root root    2048 2003-02-28 01:54 dip-3.3.7p/
drwxr-xr-x    2 root root    2048 2004-05-25 05:04 emacspeak/
drwxr-xr-x    2 root root    2048 2003-02-28 01:54 emacspeak-
ss-1.9.1/
drwxr-xr-x    2 root root    2048 2003-09-14 07:21 emu-tools-0.9.4/
drwxr-xr-x    2 root root    2048 2004-05-25 05:04 fluxbox-0.9.9/
drwxr-xr-x    2 root root    2048 2004-05-25 05:04 glibc-extra-
packages/
drwxr-xr-x    2 root root   14336 2004-06-21 01:26 ham/
drwxr-xr-x    2 root root    2048 2004-01-14 21:08 inn/
drwxr-xr-x    2 root root    2048 2004-05-24 23:48 isdn4k-utils/
drwxr-xr-x    2 root root    2048 2004-06-16 01:32 k3b/
drwxr-xr-x    2 root root    2048 2004-06-20 09:07 kfiresaver3d/
drwxr-xr-x    2 root root    2048 2003-02-28 01:54 libsafe-2.0-16/
drwxr-xr-x    2 root root    2048 2004-06-17 05:18 linux-wlan-ng/
drwxr-xr-x    2 root root    2048 2003-02-28 01:54 mpg123-0.59r/
drwxr-xr-x    2 root root    2048 2003-09-12 18:46 openmotif-2.2.2/
drwxr-xr-x    2 root root    2048 2004-06-08 03:59 parted/
drwxr-xr-x    2 root root    2048 2004-06-22 04:01 slackpkg/
drwxr-xr-x    2 root root    2048 2004-06-20 09:07 slacktrack/
```

In ognuna di queste sottodirectory, nel caso contengano un solo pacchetto o un paio di pacchetti contengono:

il file tgz del pacchetto;
una file di testo con lo stesso nome del file tgz ma con estensione txt con un contenuto uguale a quello del file slack-desc all'interno del pacchetto;
eventuali file asc per firmare il pacchetto, solitamente presenti solo nei cd ufficiali;
eventuali file hash di checksum per garantire che il pacchetto sia corretto.
Un esempio di sottodirectory con un solo pacchetto:

```
CHECKSUMS.md5  
CHECKSUMS.md5.asc  
xcdroast-0.98alpha15-i486-1.tgz  
xcdroast-0.98alpha15-i486-1.tgz.asc  
xcdroast-0.98alpha15-i486-1.txt
```

Per creare il file CHECKSUMS.md5 si usa il comando md5sum direttamente dalla sottodirectory:

```
md5sum * > CHECKSUMS.md5
```

Il file quindi risulterà del tipo:

```
cee276e5e6458af093a334bd75e20937  acidrip-0.14-i486-2ota.tgz  
fc98c72ab3a1aaf2b5a69633efd402d2  acidrip-0.14-i486-2ota.txt
```

A questo punto per controllarne l'autenticità basterà:

```
md5sum -c CHECKSUMS.md5 | less
```

Nel caso invece che la sottodirectory contenga più pacchetti, allora oltre che ai files visti finora, ne vengono aggiunti alcuni per permetterne un'installazione automatica semplificata con menu di scelta esattamente uguale a quella presente nelle directory delle ISO ufficiali.

I files aggiuntivi sono:

```
install-packages  
install.end  
maketag  
maketag.ez  
tagfile
```

I primi 2 basta copiarli pari pari da una qualsiasi directory con i pacchetti nei cd ufficiali, mentre gli ultimi 3 devono essere modificati.

In particolare i 2 maketag, molto simili, contengono i comandi per creare i menu di scelta dei pacchetti da installare e bisogna in entrambi cambiare l'elenco dei pacchetti, delle relative descrizioni, dei pacchetti attivati di default, e, infine, l'elenco dei nomi dei pacchetti nei due cicli for per la loro installazione.

Il tagfile è invece un file utilizzato nella relativa modalità d'installazione di Slackware, per automatizzare la scelta dei pacchetti. E' costituito come segue:

```
# Tagfile for postfix/packet series  
amavis: OPT  
arc: ADD  
clamav: REC  
Compress-Zlib: SKP
```

Dove a sinistra vi sono i nomi di tutti i pacchetti presenti nella sottodirectory, e i tag associati possono essere:

OPT - opzionale
REC - recommended
ADD - required
SKP - skip

I pacchetti settati come ADD vengono automaticamente installati, mentre quelli SKP vengono automaticamente saltati, per gli altri viene chiesto all'utente cosa fare. Ora non rimane altro che creare nella directory radice i files CHECKSUMS.md5, FILELIST.txt MANIFEST.bz2 PACKAGES.TXT e libraries-repository, i quali possono venire creato tutti insieme in maniera automatica tramite lo script **repos** presente negli swaret-tools:

```
repos {directory radice}
```

Il sistema d'installazione

Diamo un occhio al cd1 ufficiale della Slackware e cerchiamo di capire come funziona il sistema d'installazione. Oltre i vari HOWTO e altri file di testo nella radice del cd vi sono le directory:

```
bootdisks/  
isolinux/  
kernels/  
rootdisks/  
slackware/
```

dove bootdisks e rootdisks sono due directory che contengono tutto il necessario per il boot tramite floppy dell'installazione, in particolare la prima il disco d'avvio e la seconda il disco contenente il filesystem.

kernels contiene una collezione di kernel percompilati, utilizzabili per l'installazione e anche per il proprio sistema da installare, slackware non è altro che la directory radice del "repository" con i pacchetti e ne ha la stessa struttura.

La directory fondamentale per l'installazione da cd è isolinux, che contiene appunto il programma isolinux che serve appunto per rendere avviabile un cdrom con linux. I files che contiene interessanti sono:

message.txt

il messaggio di avvio del cd

isolinux.cfg

il file di configurazione di isolinux

initrd.img

il filesystem

diamo un'occhiata a isolinux.cfg per capire come funziona isolinux:

```
default /kernels/bare.i/bzImage initrd=initrd.img load_ramdisk=1  
prompt_ramdisk=0 ramdisk_size=6464 rw root=/dev/ram  
SLACK_KERNEL=bare.i  
prompt 1
```

```
timeout 1200
display message.txt
F1 message.txt
F2 f2.txt
F3 f3.txt
label linux
    kernel /kernels/bare.i/bzImage
```

[altre linee di configurazioni per la scelta di un differente kernel per il sistema]

Possiamo notare come in pratica viene specificato quale kernel caricare per il boot del sistema da cd e l'immagine contenente il filesystem da caricare direttamente in RAM, `initrd.img`, con le opportune dimensioni e opzioni varie.

Quindi per modificare l'installazione Slackware basta cambiare il file `initrd.img` creando l'immagine del proprio filesystem personalizzato.

Per creare `initrd.img` si utilizza il comando **mkinitrd** che a partire da una root directory specificata crea il file di immagine per tutte le sottodirectory e i files contenuti.

Nel filesystem di default di slackware tutti gli script per l'installazione sono in **/usr/lib/setup**: quello principale è `setup`, che poi chiama a sua volta tutti gli altri quando necessario, come per esempio `slackinstall` per effettuare l'effettiva installazione dei pacchetti.

Autofs e Automount

Autofs è un sistema per il mount automatico di dispositivi, per funzionare deve essere abilitato nel kernel il "kernel automounter support" che di default è quasi sempre attivato in tutte le distribuzioni, in Slackware sempre.

Dopodiché è necessario lanciare il demone **automount** all'avvio del sistema con l'opportuna configurazione. In molte distribuzioni è già presente uno script d'avvio rc.autofs, mentre in Slackware dovremo scriverlo a mano, renderlo avviabile e aggiungerlo a rc.local.

```
/usr/bin/automount -t {1} {/mnt/removable/autofs} file
{/etc/autofs}
```

questa è la linea da usare nello script d'avvio per lanciare automount.

-t è l'opzione che permette di specificare dopo quanto tempo che rimane inutilizzato un filesystem deve essere smontato: -t 1 specifica di smontarlo dopo 1 secondo.

Il primo parametro è la directory radice all'interno della quale porre tutti i dispositivi montati. Ho specificato /mnt/removable/autofs anziché direttamente /mnt perchè nella root directory di automount non si possono vedere altre directory se non quelle relative a dispositivi già montati, quindi è meglio lasciarlo lavorare in una differente directory, nel caso ci siano direttamente in /mnt altre partizioni da lasciare sempre montate. Inoltre ho preferito creare un'ulteriore sottodirectory (./autofs)così da linkare le varie directory cdrom, dvd, ecc. da /mnt/removable/autofs a /mnt/removable in modo da poter vedere l'elenco di dispositivi montabili, che, ricordo, non si può vedere direttamente dalla directory di root di autofs. Ho anche lasciato separata la directory con questi ultimi link da /mnt perchè ogni volta che si esegue un semplice ls sulla directory contenente i link ai mountpoint dei dispositivi, autofs checkerà tutte le periferiche relative facendovi perdere qualche decina di secondi.

Il secondo parametro specifica che tipo di configurazione si usa, un semplice file come in questo caso oppure configurazioni più complessi tramite l'uso di ldap o nis.

Infine l'ultimo parametro specifica dove trovare la configurazione.

Ora vediamo la sintassi da utilizzare per il file di configurazione:

```
cdrom
-fstype=iso9660,ro,nosuid,nodev,exec,mode=0665,uid=99,gid=99 :/
dev/sr0
dvd
-fstype=iso9660,ro,nosuid,nodev,exec,mode=0665,uid=99,gid=99 :/
dev/sr1
hdusb
-fstype=auto,rw,nosuid,nodev,exec,umask=0000,uid=99,gid=99 :/
dev/sda1
floppy
-fstype=auto,rw,nosuid,nodev,exec,umask=0000,uid=99,gid=99 :/
dev/fd0
```

il primo elemento di ogni riga è il mountpoint (una volta montato un cd sarà per sempio /mnt/removable/cdrom), il secondo le opzioni per il mount, a parte -fstype={filesystem} le altre sono le solite opzioni di mount presenti anche in fstab. L'ultimo elemento è il dispositivo. Per sapere per certo quali sono i filesystem montati in un dato istante si può utilizzare il comando df.

Come dicevo precedentemente per comodità consiglio di fare dei link simbolici ai mountpoints dei dispositivi in /mnt/removable:

```
ln -s /mnt/removable/autofs/cdrom /mnt/removable/cdrom
```

Infine consiglio anche di linkare queste ultime directory create anche sul desktop, e di modificare fstab in modo tale che i mountpoints dei device rimovibili siano ancora una volta questi ultimi links.

Ide e gestione di progetti

Java

Uno dei vantaggi principali nell'utilizzare java è la presenza di ottimi IDE gratuiti e Open Source; parlo di **Netbeans** e **Eclipse**.

Utilizzando Netbeans si può tranquillamente gestire il proprio progetto, aggiungervi una classe per la creazione di una gui (un Jform per esempio), editare graficamente la gui tramite gli strumenti automatici dell'IDE, personalizzandola pesantemente anche tramite codice scritto a mano, aggiunto alla classe dal menu grafico Code, in modo da non creare problemi mischiando codice pregenerato e codice hand-written. Si può anche scrivere codice direttamente nel file .java relativo alla Jform, nelle sezioni apposite, lasciando intatte quelle marcate come da non modificare da Netbeans.

NetBeans però è abbastanza legato all'ambiente di sviluppo della Sun e supporta solo swing.

Eclipse invece è una piattaforma di sviluppo comune utilizzata da molte aziende di grosso calibro come Adobe, IBM e Nokia, supporta molti compilatori java differenti, tra cui anche gcj. Inoltre tramite l'architettura a plugin permette di editare graficamente le GUI sia basate su Swing che su SWT, permettendone una più libera modifica manuale del codice rispetto a NetBeans. Inoltre anche Eclipse mette a disposizione tutti gli strumenti per lo sviluppo di web services tra cui http monitors, ecc.

QTDesigner e Kdevelop

Come si fa a gestire un progetto abbastanza complesso, con alcune parti che necessitano di una GUI grafica, utilizzando QT come framework di base?

Si parte da Kdevelop, l'IDE default di KDE, e si crea un Qmake project. Si può scegliere tra due alternative, la prima è "Main Application", e questo è il caso da scegliere se si intende sviluppare tutta la gui a mano, senza l'aiuto di strumenti visuali, infatti Kdevelop vi aiuterà a partire creando il .cpp e il .h relativi ad una semplice Main Application generale. Da questo punto in avanti non avrete bisogno di altro che Kdevelop e una buona guida di riferimento sulle qt, quale quella fornita con Kdevelop Assistant. L'altro caso, "a simple Hello World program" è quello da scegliere se si intende gestire il processo di creazione delle gui tramite lo strumento grafico principe fornito insieme alle QT: QTDesigner.

In questo caso infatti Kdevelop non creerà nulla se non un file vuoto di partenza, ma vi dà la possibilità di aggiungere al progetto dei file ".ui", ovvero i file xml relativi alle gui editate con qt designer. A questo punto basterà utilizzare quest'ultimo programma per editare la gui graficamente a piacere, creare gli slot e connetterli ai relativi signals sempre tramite semplici menu. Per l'editing degli slots, basta editare tramite kdevelop (ma anche qt designer ve ne dà la possibilità) il file

{nome}.ui.h che contiene le dichiarazioni di tutti gli slot e che viene incluso direttamente tramite una semplice #include, dal file {nome}.cpp generato da qt designer.

Se si necessita di includere degli headers, o dichiarare delle variabili di classe per la classe che rappresenta la gui, porre gli #include oppure le dichiarazioni delle variabili globali all'inizio del file {nome}.ui.h. Se si ha bisogno di aggiungere comandi all'inizializzazione o alla distruzione di tale classe, creare un metodo init e un metodo destroy sempre nel file {nome}.ui.h.

Anjuta (o Kdevelop) e Glade

Anjuta e Glade sono gli strumenti da utilizzare se si intende sviluppare un progetto con gui basata

su gtk (o anche KDevelop visto che nelle ultime versioni è stato aggiunto pieno supporto sia alle gtk che a gtkmm).

Si parte da Anjuta, si crea un progetto gtk basato sul linguaggio desiderato, dopodichè basta chiamare “Edit Application GUI” dal menu Project che viene chiamato Glade, lo strumento per l'editing visuale della GUI. Qui si può modificare la GUI a piacere, creando gli eventuali gestori di segnali che servono tramite menu grafici. Dopodichè si può tornare a lavorare con Anjuta, nel caso del linguaggio C modificando il file **callbacks.c**, l'unico file modificabile tra quelli creati (tutti gli altri vengono ogni volta sovrascritti da Glade), in tale file ci sono tutte le dichiarazioni degli “slot”, che aspettano solo di essere implementate. Se servono variabili globali o includes, basta aggiungerli all'inizio di tale file.

Se serve modificare o aggiungere componenti durante l'inizializzazione agire direttamente sul file **main.c**.

Se invece serve raggiungere un altro componente visuale da un componente visuale, per esempio modificare lo stato di un checkbox se ne viene cliccato un altro, si può fare sempre dal file **callbacks.c** all'interno dello slot, utilizzando la funzione **lookup_widget**.

Con Kdevelop l'unica differenza è che non c'è un collegamento esplicito dall'IDE a Glade, ma bisogna da Glade aprire a mano il file .glade creato da Kdevelop.

Per un'ottima guida rapida di consultazione sulle gtk, utilizzare devhelp o Kdevelop Assistant.

Se invece si desidera utilizzare il C++ come linguaggio, invece delle gtk direttamente si utilizzeranno le gtkmm, i bindings ufficiali in C++ delle gtk.

Naturalmente si può usare comunque Glade per l'editing dell'interfaccia solo che questa volta i file creati sono: **main_window_glade.hh**, **main_window_glade.cc**, **main_window.hh**, **main_window.cc** e **prova.cc**, dove **prova.cc** è il file con il main e che costruisce la finestra, editabile a piacere, in modo tale da poter aggiungere le altre inizializzazioni necessarie; i file ***_glade*** sono quelli relativi alla finestra costruita da Glade, e NON sono editabili.

Per poter implementare gli slots invece si editano gli altri due file: **main_window***, che sono relativi ad una classe derivata da quella relativa alla finestra creata da Glade, quindi se necessario si possono aggiungere senza problemi oggetti di classe, altri metodi, o qualunque altra cosa.

Importante ricordarsi di cambiare la visibilità da **private** a **protected** delle widget della finestra creata da Glade, se no non saranno accessibili dalla nostra classe.

Dalla versione 2.x Glade sconsiglia l'utilizzo del generatore di codice ed invece suggerisce l'utilizzo di **libglade**, una libreria disponibile in vari linguaggi di programmazione differenti, in grado di generare la gui al momento del lancio dell'applicazione a partire dal semplice file xml generato da Glade. In questo modo la gui è completamente gestita da Glade durante tutto il progetto, indipendentemente dal linguaggio utilizzato. Per fare questo basta aggiungere 2-3 linee di codice all'inizio del programma per caricare il file xml generato da Glade.

Postfix come relay con autenticazione TLS+SASL

Se volete installarvi un proprio server di posta ma per vari problemi la classe di ip di cui il vostro server fa parte si trova blacklistata, allora tutto ciò che vi serve è un secondo smtp server attraverso il quale inviare tutte le vostre lettere. Dovete quindi configurare il vostro postfix in modo tale che smisti tutta la posta in uscita al secondo smtp server.

Nel caso in cui quest'ultimo smtp necessiti di autenticazione criptata, allora dovrete configurare postfix in modo che anch'esso utilizzi tale autenticazione per mandare la posta al secondo smtp. Per fare questo prima è necessario installare postfix con supporto tls e cyrus-sasl.

Per installare cyrus da sorgente:

```
./configure --enable-login --enable-plain --enable-anon
make
make install
ln -s /usr/local/lib/sasl2 /usr/lib/sasl2
```

Per installare postfix da sorgente con il supporto per sasl2 e tls, prima è necessario scaricare sia i sorgenti di postfix che quelli della patch tls corrispondente, poi scompattare entrambi gli archivi. Per pathchare postfix bisogna eseguire il seguente comando dalla directory padre in cui è con tenuta la directory con i sorgenti di postfix:

```
path -p0 < pfixtls{versione}
```

a questo punto possiamo compilare postfix:

```
make makefiles CCARGS="-DUSE_SASL_AUTH -I/usr/local/include/sasl -
DUSE_SSL -I/usr/local/ssl/include" AUXLIBS="-L/usr/local/lib
-lsasl2 -L/usr/local/ssl/lib -lssl -lcrypto"
make install
```

Dopodiché iniziamo la configurazione editando /etc/postfix/mail.cf, aggiungiamo:

```
relayhost= foo.com
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
```

dove foo.com è l'host a cui dobbiamo mandare la mail in uscita. Aggiungiamo inoltre:

```
smtp_sasl_security_options =
```

se vogliamo un'autenticazione anonima in chiaro, invece:

```
smtp_sasl_security_options = noplaintext, noanonymous
```

se vogliamo un'autenticazione mai anonima con user e pass criptati. Se inoltre vogliamo che utilizzare smtp over ssl (e abbiamo configurato postfix con support tls), allora:

```
smtp_use_tls = yes
```

Editiamo poi il file /etc/postfix/sasl_passwd e aggiungiamo:

```
foo.com          username:password
```

dove foo.com è come al solito l'host a cui dobbiamo mandare la mail in uscita e username e password sono lo user e la pass necessari per loggarsi su quell'smtp. Per motivi di sicurezza siccome lo user e la pass sono scritti in chiaro in un file leggibile da tutti è meglio cambiare diritti a al file sasl_passwd:

```
chown root:root /etc/postfix/sasl_passwd && chmod  
600 /etc/postfix/sasl_passwd
```

A questo punto basta lanciare il comando:

```
postmap hash:/etc/postfix/sasl_passwd
```

e riavviare postfix.

Postfix come server di posta con autenticazione TLS+SASL

Prima di tutto abbiamo bisogno del pacchetto postfix e del pacchetto cyrus-sasl che possiamo trovare su Linuxpackages.net.

Una volta installati dobbiamo specificare quale metodo di autenticazione usare per SASL, in particolare se vogliamo che gli utenti di posta siano gli stessi utenti del sistema, allora dobbiamo creare il file `/usr/lib/sasl2/smtpd.conf` con la seguente linea:

```
pwcheck_method: saslauthd
```

e mettere all'avvio del sistema:

```
saslauthd -a shadow
```

Dopodichè dobbiamo configurare postfix, aggiungendo le seguenti linee a `/etc/postfix/main.cf`:

```
smtpd_sasl_auth_enable = yes
broken_sasl_auth_clients = yes
smtpd_sasl_security_options = noanonymous
smtpd_recipient_restrictions = permit_sasl_authenticated ,
permit_auth_destination, reject, check_relay_domains
smtpd_sasl_local_domain = $myhostname, stabellini.net, afga.it
```

dove `smtpd_sasl_security_options` specifica come vengono eseguiti i login SASL, in questo caso non vengono accettati login anonimi.

`smtpd_recipient_restrictions` invece pone restrizioni d'accesso a smtpd, in base al destinatario: in questo caso vengono permessi gli accessi autenticati tramite sasl, oppure quelli che hanno come destinatario un utente locale e rifiutati tutti gli altri.

In questo modo SASL e postfix sono configurati correttamente per l'autenticazione in chiaro, per terminare correttamente la configurazione basta abilitare il supporto tls in postfix, aggiungendo a `main.cf` le seguenti linee:

```
smtpd_use_tls = yes
smtpd_tls_key_file = /etc/postfix/newreq.pem
smtpd_tls_cert_file = /etc/postfix/newcert.pem
smtpd_tls_CAfile = /etc/postfix/cacert.pem
smtpd_tls_loglevel = 3
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
```

e generando tali certificati. Bisogna andare in `/etc/ssl/misc` ed eseguire:

```
./CA.sh -newca
./CA.sh -newreq
./CA.sh -sign
```

generati i certificati li copiamo in `/etc/postfix`:

```
cp newcert.pem /etc/postfix/
cp newreq.pem /etc/postfix/
cp demoCA/cacert.pem /etc/postfix/
```

GNU Java Tools

gij

gij è la Java Virtual Machine fornita da GNU, funziona esattamente come il comando “java” fornito da Sun, quindi ha le opzioni -cp e -classpath, ecc.

gcj

gcj è il compilatore java fornito da GNU, è in grado di generare sia bytecode, quindi i tipici file .class, che direttamente codice nativo.

Per esempio per generare normale bytecode:

```
gcj -C File.java
```

invece per generare codice nativo:

```
gcj File.java --main=File -o File
```

Da notare che gcj e gij hanno di default nel classpath solo i jar e i class nella directory corrente e quelli in /usr/share/java.

E' possibile integrare completamente gcj ed Eclipse in modo tale da sviluppare progetti java completamente basati sul compilatore open source e generare automaticamente dei makefiles che permettano di compilare le applicazioni scritte in java in codice nativo. Basta avere installati gcj, eclipse ed il plugin gcjbuilder.

Classpath

GNU classpath è il progetto GNU per la realizzazione di tutte le librerie java.* e javax.* che servono per eseguire e compilare programmi java. Al momento gcc 3.3.4 non utilizza ancora pienamente classpath ma ancora libgcj che a livello di compatibilità con la jdk ufficiale è inferiore, però dall'uscita di gcc 4.0 in poi i due progetti dovrebbero essere praticamente sincronizzati. In ogni caso è possibile utilizzare gcj con classpath scaricando classpath a parte e utilizzandolo nelle compilazioni specificando il jar a mano nel bootclasspath. Stessa identica cosa per gij, anch'esso usa di default libgcj che è inferiore, in ogni caso specificando a mano nel bootclasspath di utilizzare GNU classpath si risolvono tutti i problemi. Altre free jvm come Kaffe invece utilizzano direttamente GNU classpath al completo.

Swing e AWT

Al momento le uniche cose che mancano veramente alle implementazioni di java open source sono awt e soprattutto swing. Un programmatore java che intende sviluppare un software grafico in grado di girare sulle free jvm come Kaffe ha sostanzialmente 3 alternative:

- utilizzare swt, la libreria open source di IBM che chiama al di sotto direttamente le API del SO, molto veloce;
- utilizzare swingwt, una libreria che consente di scrivere programmi utilizzando normalmente le API swing, ma in realtà mappa le chiamate a javax.swing e java.awt sulle librerie swt. Quindi permette di utilizzare Netbeans per creare programmi grafici che in realtà hanno tutti i pregi di swt: look nativo e maggiore velocità. Per contro l'utente deve avere il package swingwt che è circa 10MB. Anche questa è open source.
- Utilizzare i bindings java-gnome o qtjava, al contrario delle due precedenti questa è una

soluzione solo linux. E' possibile però in questo modo sviluppare delle applicazioni tramite Glade per disegnare l'interfaccia grafica e Eclipse per tutto il resto, completamente integrate con Gnome\Gtk o KDE\Qt, e persino compilate in nativo.

Distribuzione di software java sotto linux

Mentre è normale nel mondo windows e macosx avere programmi tutti contenuti in una singola directory, nel mondo linux questo non accade praticamente mai.

Distribuire software scritto in java quindi in directory autocontenute sotto linux non è conforme agli standard e non è un metodo che può essere utilizzato dalle distribuzioni.

Bisogna invece porre tutti i jar in **/usr/share/java** con nel filename anche la versione, più un symlink senza versione:

```
nome-versione.jar  
ln -s nome-versione.jar nome.jar
```

Tutte le librerie native invece devono andare in **/usr/lib/jni** ed infine gli eseguibili e i files di configurazione normalmente in /etc e /usr/bin.

Solitamente gli eseguibili non sono altro che script sh che richiamano i jar, quindi un esempio di script d'avvio è il seguente:

```
#!/bin/sh  
  
if test $JAVA_HOME  
then  
    JAVA=$JAVA_HOME/bin/java  
else  
    JAVA=/usr/bin/java  
fi  
  
$JAVA  
-classpath /usr/share/java/nome.jar:/usr/share/java/dipendenza.jar  
-Djava.library.path=/usr/lib/jni classe.metodo
```

Il suono con linux

Dal lato kernel il suono sotto linux si basa su un layer chiamato ALSA, fino alle prime versioni del kernel 2.4 però veniva ancora utilizzato un altro layer più vecchio e con minori funzionalità chiamato OSS. Anche adesso quindi per mantenere la retrocompatibilità ALSA offre dei drivers di accesso OSS simulati, oltre ai propri, per permettere ai software meno aggiornati di funzionare comunque.

I device OSS sia con devfs che con udev sono i seguenti:

```
/dev/dsp*  
/dev/mixer
```

mentre quelli ALSA:

```
/dev/snd/pcmC0D0  
/dev/snd/*
```

Siccome utilizzare direttamente ALSA è difficile per i programmatori, ed inoltre ALSA di per sé non permette comunque a più programmi di avere accesso contemporaneo alla scheda sonora, sono stati realizzati dei sound servers, ovvero dei demoni che gestiscono il device relativo alla scheda sonora tramite ALSA, ma forniscono ai programmatori un modello semplificato di accesso ed inoltre sono in grado di gestire più programmi in contemporanea e persino il suono in remoto. Gnome utilizza il sound server ESD mentre KDE ARTS; quest'ultimo sta avendo rallentamenti nello sviluppo quindi forse nelle prossime versioni verrà sostituito.

Se per esempio si utilizza KDE con ARTS allora tutti i programmi che vogliono utilizzare la scheda sonora devono utilizzare arts per accedervi, se no avranno dei problemi, se invece non si utilizza ARTS allora i programmi dovranno accedere al suono tramite OSS o ALSA.

Generalmente tutti i software principali (mplayer, xine, xmms) hanno più di un plugin di supporto per i drivers audio, quindi possono essere configurati per utilizzare OSS, ALSA, ARTS o ESD.

Un altro modo molto usato per accedere alle funzionalità sonore è la libreria SDL, un'ottimo strumento per la realizzazione di applicazioni multimediali come giochi, ecc.. Essendo una libreria di più alto livello può essere compilata sia utilizzando OSS che ARTS o ESD.

Device rimovibili e Kernel 2.6

Finalmente con l'arrivo della versione 2.6 il kernel da tutti gli strumenti a programmatori e utenti per poter gestire al meglio l'hardware rimovibile. Così KDE e Gnome tramite freedesktops.org hanno stabilito una serie di standards per poter interoperare al meglio gestendo l'hardware in maniera uniforme. Gli strumenti a disposizione sono i seguenti:

Hotplug

Sistema molto potente per la gestione dell'hardware rimovibile. Quando viene aggiunto nuovo hardware hotplug si occupa di caricare i moduli kernel necessari ed eventualmente eseguire qualche script per la sua configurazione.

udev

Il kernel 2.6 utilizza come filesystem per l'accesso ai device udev anziché devfs. Questo è un cambiamento molto importante, perché mentre devfs nella directory /dev creava automaticamente tutti i dispositivi gestibili dal kernel, creando anche parecchia confusione ed una certa difficoltà di utilizzo da parte dell'utente, udev crea solamente quelli effettivamente presenti nel sistema. Se viene aggiunto un dispositivo, il kernel aggiunge al volo un nuovo device in /dev.

E' possibile configurare udev in modo da scegliere in nomi dei file in /dev per i singoli dispositivi, in particolare è possibile fare in modo che una penna usb sia sempre associata ad uno stesso file, ecc.

Per la configurazione di udev, bisogna modificare i files in **/etc/udev**.

Altra feature molto utile e potente di udev è il fatto che quando viene aggiunto un nuovo dispositivo e quindi creato un nuovo file in /dev, udev controlla anche in **/etc/dev.d** la presenza di una script con lo stesso nome del file appena aggiunto e se è presente lo esegue. In questo modo è possibile realizzare in maniera molto semplice il mount automatico dei dispositivi.

HAL

HAL sta per Hardware Abstraction Layer ed è una libreria ed un demone utilizzati sia da KDE che da Gnome per la gestione uniforme dei dispositivi: legge tutte le informazioni sullo stato del sistema tramite /sys (il sostituto a /proc nel kernel 2.6), e le rende disponibili ai programmi. In questo modo è possibile per esempio avere una lista di tutti i dispositivi presenti nel sistema, avere informazioni su di essi, sapere a che device sono associati in /dev, sapere tramite quali API sono accessibili, ecc.

D-BUS

D-Bus è un sistema per la comunicazione tra programmi realizzato tramite un demone che permette la realizzazione in modo molto semplice di message-passing tra processi differenti.

D-Bus inoltre notifica ai programmi eventi quali l'aggiunta o la rimozione di nuovo hardware. In questo modo un programma in esecuzione può accorgersi se è stata collegata una penna usb.

La visione d'insieme

Come vengono utilizzati tutti insieme questi strumenti da KDE o da Gnome per fornire un supporto uniforme per l'hardware rimovibile?

Quando viene collegata una penna usb al pc, il kernel carica i moduli necessari tramite hotplug, genera il file relativo in /dev tramite udev e aggiunge tutte le informazioni un suo possesso a /sys. Supponiamo di non aver manualmente configurato nulla quindi udev non esegue nessuno script e

non monta il dispositivo.

Il demone hald che monitora /sys si accorge della nuova penna presente, quindi notifica l'evento a tutti tramite d-bus, quindi kde se ne accorge e mounta automaticamente il dispositivo e magari crea anche l'icona sul desktop.

Come fare se vogliamo comunque mantenere la possibilità di montare e smontare a mano i dispositivi? I metodi sono 2: o si utilizza **fstab-sync**, un'utility di HAL che sincronizza fstab con i nuovi device presenti, oppure si può lasciare fstab sempre intonso e utilizzare **pmount** che permette di montare e smontare dispositivi tramite hald.

La nuova generazione del server X

Adesso come adesso il server X.org che si occupa di gestire il disegno a schermo delle finestre è implementato in 2D in stile framebuffer, con varie estensioni che gli permettono di interfacciare direttamente porzioni di schermo o finestre con opengl, e di renderizzare in modo più avanzato lettere e disegni. Al momento il codice non è ben modularizzato, e non sfrutta bene le nuove funzionalità offerte dal kernel 2.6, anzi in molti casi si sovrappone ad esso.

Con la prossima generazione si ha intenzione di scrivere un server X che si appoggia completamente su **opengl** per ogni operazione, e, nel caso la scheda video non supporti opengl, lo simula comunque via software tramite Mesa. Inoltre probabilmente verrà ridisegnato da capo pur mantenendo una certa compatibilità all'indietro, in modo da essere più facilmente mantenibile e sfruttare le nuove capacità del kernel.

In questo modo non solo sarà molto più veloce ed efficiente di quello odierno, ma offrirà ai livelli superiori nuove possibilità quali quella di realizzare windows managers tridimensionali come il progetto looking glass di sun. Al momento gli sforzi vanno in direzioni differenti con lo sviluppo contemporaneo di xgl e luminocity, ma con buona probabilità alla fine sarà solo il primo ad essere completamente portato a termine e globalmente accettato. **Xgl** è infatti una vera e propria implementazione di un X server tramite opengl, luminocity invece contiene in sé anche funzionalità da windows manager. Una volta che xgl sarà terminato allora KDE e Gnome avranno tutti gli strumenti per realizzare effetti grafici come quelli di Aqua sotto OSX e anche migliori. Inoltre anche Cairo, una libreria molto avanzata per la grafica 2D potrà essere portata in modo rapido sulla nuova piattaforma.

Backup di Sistema

Effettuare periodici backup dei propri server è un'attività di fondamentale importanza per tutti gli amministratori di reti. Sotto linux ci sono moltissimi tool a disposizione per questo scopo:

tar

tar è il metodo standard di unix per creare archivi, è in grado di mantenere gli stessi permessi per i files e le directory archiviate ed inoltre gestisce anche backup incrementali tramite l'opzione --incremental

rsync

rsync è un'utility molto avanzata che permette di seguire copie di files tra macchine differenti in maniera incrementale, ottimizzando lo sfruttamento della connessione e mantenendo tutti i diritti dei files originali.

Un esempio su come può essere utilizzato per eseguire un backup di un'intero sistema via rete è il seguente:

```
rsync -azvr -e ssh --delete / root@192.168.2.3:/mnt/backup
```

partimage

Utility testuale con interfaccia grafica ncurses per effettuare backup\restore di intere partizioni; è in grado di comprimere i backup, effettuare i salvataggi e i restore via rete, e salvare il MBR. Unico difetto: può salvare solo partizioni non montate.

mondorescue

Utility a linea di comando per effettuare il backup dell'intero sistema (tutte le partizioni più MBR) su cd; è in grado di funzionare anche con le partizioni montate e salvare file iso su partizioni NFS. Il suo scopo è creare un insieme di cd autobootanti tramite i quali è possibile ripristinare il sistema tramite un semplice click su un menu grafico.

Virtualizzazione ed Emulazione

Negli ultimi mesi (inizio 2005) hanno raggiunto la maturità un paio di progetti open source gratuiti di grande utilità ed interesse sia in abito server che desktop: qemu e xen.

QEMU

Qemu è un software di emulazione, che permette di emulare un intero sistema, anche di architettura differente, sul proprio sistema operativo. Permette inoltre di emulare anche un solo processo alla volta. Parlando di emulazione si ha sempre a che fare con grossi rallentamenti rispetto alle prestazioni native, ma qui sta la grande forza di qemu: grazie ad un modulo del kernel opzionale gratuito ma non open source, è possibile incrementare moltissimo le prestazioni del sistema emulato, naturalmente solo se l'architettura emulata è la stessa della vostra macchina hardware; si possono raggiungere prestazioni complessive pari al 50% di quelle native (simile a vmware). In poche parole si crea un file di immagine che rappresenta l'hd del sistema emulato, si fa partire qemu con quel file come argomento e voilà, parte un'intera nuova macchina (virtuale) dentro il nostro sistema operativo.

Per configurare qemu in modo tale che host e guest possano comunicare via rete, per esempio via ssh o ftp, allora bisogna aggiungere nell'`rc.local` del guest:

```
#modprobe ne2k-pci (inutile nelle ultime versioni)
dhcpcd
```

dopodichè dobbiamo lanciare qemu in questo modo:

```
qemu -k it -boot c -net nic -net user -redir tcp:2222:10.0.2.15:22
{/mnt/slamd64/slamd64.img}
```

in questo modo aprendo delle connessioni ssh su localhost con porta 2222 nel sistema host, queste vengono ridirezionate sulla porta 22 del guest.

Se il vostro sistema guest emula qualche architettura diversa da quella della vostra macchina e quindi è troppo lento per utilizzare ssh si può utilizzare ftp. In questo caso bisogna lanciare qemu in questo modo:

```
qemu -k it -boot c -net nic -net user -redir tcp:21:10.0.2.15:21 -
redir udp:21:10.0.2.15:21 -redir tcp:20:10.0.2.15:20 -redir udp:
20:10.0.2.15:20 {/mnt/winarchive/slack10/linux.img}
```

poi dall'host aprire connessioni ftp in passive mode su localhost direttamente sulla 21.

Fare attenzione al fatto che i ping tra guest os e host non funzionano nelle nuove versioni di qemu.

XEN

Completamente differente da qemu c'è **Xen**, che invece è un software di virtualizzazione e in quanto tale non emula nulla, permette semplicemente di eseguire più istanze contemporaneamente di sistemi operativi con una performance paragonabile a quella nativa, sulla vostra macchina hardware. Tuttavia i sistemi operativi supportati finora sono solo Linux e NetBSD. In questo caso bisogna bootare all'avvio xen anziché il kernel "normale", dopodichè possiamo creare tramite xm più macchine virtuali e configurarle per avere come harddisk i file di immagine che vogliamo, ecc.

Postfix + Cyrus-Sasl come server di posta su Slackware con autenticazione

Per utilizzare postfix come server di posta con autenticazione su Slackware, sono necessari i seguenti passi:

-scaricate da linuxpackages.net i pacchetti relativi alla vostra versione di Slackware o Slamd64 di cyrus-sasl e postfix e successivamente installateli.

-create la directory `/var/state/saslauthd` e cambiategli il proprietario con `chown` in modo che sia lo stesso utente che poi farà girare il demone di autenticazione `saslauthd`.

-create il file `/usr/lib/sasl2/smtp.conf` e scrivete al suo interno la configurazione relativa a `saslauthd`. Siccome può funzionare in molti modi differenti, riporto di seguito un esempio molto semplice, in cui utilizziamo un login in chiaro e come utenti quelli di sistema:

```
pwcheck_method: saslauthd
mech_list: plain login
```

-modificate `/etc/postfix/main.cf` in modo che la configurazione sia adeguata alle vostre esigenze, poi aggiungete in fondo le righe relative alla configurazione dell'autenticazione, anche in questo caso sono possibili molte scelte differenti, continuando con le scelte effettuate in precedenza si ha:

```
smtpd_sasl_auth_enable = yes
broken_sasl_auth_clients = yes
smtpd_sasl_security_options = noanonymous
smtpd_recipient_restrictions = permit_sasl_authenticated ,
permit_auth_destination, reject, check_relay_domains
smtpd_sasl_local_domain = $myhostname
```

-a questo punto non dovete fare altro che avviare il demone di autenticazione con l'utente che avete scelto in precedenza:

```
saslauthd -a shadow
```

e successivamente avviare postfix:

```
postfix start
```

ssh senza password

Se vogliamo utilizzare ssh all'interno di script da mettere in cron può diventare difficoltoso l'inserimento della password per l'autenticazione, per risolvere il problema basta cambiare metodo di autenticazione, passando al sistema di chiavi pubbliche.

Prima di tutto bisogna generare la coppia di chiavi sulla macchina client:

```
ssh-keygen -t dsa
```

dopodichè bisogna copiare la chiave pubblica del client sul server su cui si vuole autenticarsi:

```
scp .ssh/id_dsa.pub server:/home/utente/.ssh/authorized_keys
```

a questo punto ogni volta che ci si vuole autenticare dal client sul server verrà richiesta la passphrase relativa alla propria chiave private anziché la password. Non è un gran migliramento. Per evitare il problema si può scegliere di non utilizzare passphrase e cercare di tamponare i problemi di sicurezza aggiungendo sul server limitazioni, per esempio è possibile aggiungere:

```
command="command"
```

subito prima della chiave pubblica nel file `authorized_keys` sul server, in modo tale che l'utente che si logga con quella chiave pubblica può eseguire solo quel comando.

Altri esempi di limitazioni sono:

```
from="hostA,hostB",no-port-forwarding,no-X11-forwarding,no-agent-  
forwading,no-pty
```

Il metodo migliore invece per evitare il problema della richiesta della passphrase invece è utilizzare **ssh-agent**, ovvero un sistema che permette all'utente sul client di inserire la passphrase solo una volta al logging e basta. Funziona in questo modo: si mette in esecuzione `ssh-agent` in **.bash_profile** in modo tale che venga eseguito appena l'utente entra nel sistema, controllando che non ve ne sia uno già in esecuzione, poi si aggiungono all'agente le chiavi che si intendono utilizzare con **ssh-add**, che di default aggiunge tutte le chiavi dell'utente appena entrato nel sistema. Per esempio si può aggiungere a `.bash_profile`:

```
if test -z $SSH_AGENT_PID  
then  
    eval `ssh-agent`  
    ssh-add  
fi
```

`eval `ssh-agent`` serve per esportare alcune variabili globali utili a tutti i programmi che vogliono utilizzare l'agent.

Spegnere la Slackware

Di default la Slackware quando viene spenta non “spegne del tutto” il pc, semplicemente al termine della procedura scrive a schermo power off, poi sta all'utente togliere del tutto la corrente al pc. Per evitare questo basta abilitare **apm**, ovvero

```
modprobe apm
```

dopodichè lanciando **halt** come al solito il pc verrà spento del tutto. Il posto migliore dove aggiungere modprobe apm è **/etc/rc.d/rc.modules**, dove in realtà basta appunto scommentare una linea.

Kppp da utente

Per eseguire kppp da utente senza problemi bisogna effettuare un paio di piccole operazioni:

- 1) assicurarsi che in /etc/ppp/options ci sia l'opzione **auth** abilitata
- 2) scommentare l'opzione **noauth** in /etc/ppp/peers/kppp-options
- 3) aggiungere **route add default ppp0** in fondo a /etc/ppp/ip-up

Synergy

Se state lavorando con più computer contemporaneamente sulla stessa scrivania probabilmente trovare comodo poterli controllare tutti con gli stessi mouse e tastiera.

Questo è possibile utilizzando un software chiamato synergy, che deve essere installato su tutti i pc. Sul computer al quale sono collegati il mouse e la tastiera che volete usare dovete eseguire il server, mentre sugli altri il client.

Il server viene invocato in questo modo:

```
synergys --daemon --config synergy.conf
```

il file di configurazione synergy.conf è di questo tipo:

```
section: screens
    desktop:
    laptop:
end
section: links
    desktop:
        left = laptop
    laptop:
        right = desktop
end
section: options
    keystroke(control+alt+left) = switchInDirection(left)
    keystrokeexport EDITOR=vike(control+alt+right) =
switchInDirection(right)
```

dove desktop e laptop sono gli host dei due computer in questione, section links specifica la posizione degli schermi e options permette di aggiungere degli shortcuts per passare da uno schermo all'altro.

Il client invece si invoca in questo modo:

```
synergyc ip -n laptop
```

dove ip è l'indirizzo del server, mentre laptop è il proprio hostname come specificato nel file di configurazione sul server.

Purtroppo tutto il traffico passa in chiaro quindi è consigliabile far passare tutto attraverso un bel tunnel ssh, basta eseguire sul client:

```
ssh -f -N -L 24800:IP:24800 IP
```

dove IP è l'indirizzo del server, poi eseguire synergyc con indirizzo 127.0.0.1.

Salvare pagine web in firefox

Un problema molto sentito è quello di salvare una pagina web in modo da poterla consultare anche offline oppure in modo da poter sempre accedere al suo presente contenuto anche se questo in futuro dovesse cambiare.

Firefox è perfettamente in grado di salvare tutto il contenuto di una pagina web, ma lo fa creando un file html e una directory con le immagini, i css, ecc.

In questo modo diventa difficile gestire un gran numero di pagine salvate in sottocartelle dato che ognuna crea a sua volta una nuova sottocartella.

Per salvare le pagine in un unico file o in modo da poterle poi gestire come bookmark esiste un'estensione chiamata scrapbook, che di default salva tutti i le pagine web nel proprio profilo e permette di organizzarle in folders.

Unico problema: questa estensione potrebbe presentare incompatibilità con alcuni temi, se questo dovesse accadere semplicemente passate al tema di default.

Aggiungere il cestino sul desktop in Kubuntu

L'icona del cestino di default in Kubuntu è presente sulla barra delle applicazioni. Per chi volesse aggiungerla sul proprio desktop, in stile classico, la procedura è la seguente:

- create sul desktop un file di testo chiamato Trask semplicemente cliccando con destro
- editatelo inserendo il seguente testo:

```
[Desktop Entry]
Comment=Contains removed files
EmptyIcon=trashcan_empty
Encoding=UTF-8
Icon=trashcan_full
Name=Trash
Type=Link
URL=trash:/
```

salvate e riavviate.

Alltray

Vi sarà sicuramente capitato più di una volta di voler mettere una applicazione in systray che non lo supporta. Alcuni esempi di applicazioni molto popolari che non supportano la systray sono thunderbird, xmms e linphone.

In realtà c'è una soluzione molto semplice a questo problema: alltray.

Alltray è un piccolo programmino che permette di mettere direttamente in systray qualsiasi applicazione passata come argomento, per esempio:

```
alltray linphone
```

Purtroppo c'è un piccolo problema: kde fa partire ogni sessione esattamente come avete lasciato la precedente, ma non riesce a riconoscere se una applicazione è stata lanciata tramite alltray, quindi al prossimo avvio della sessione in questo caso linphone verrebbe fatto partire normalmente.

Per questo motivo è necessario scrivere un piccolo script da mettere in .kde/Autostart di questo tipo:

```
killall linphone  
alltray linphone
```

questo per ogni applicazione che volete lanciare automaticamente tramite alltray.

Non una bellissima soluzione ma certamente ben funzionante.

Un'alternativa e' disabilitare la modalita' "Restore session" tramite Kcontrol, sotto "Session Manager", e selezionare invece la modalita' "Start with an empty session".

In questo modo si puo' evitare di aggiungere il killall nello script di Autostart.

Ubuntu: avvio in modalita' console

Da quanto Ubuntu ha sostituito il vecchio **init** con **upstart** come sistema di avvio iniziale dei processi non e' piu' possibile semplicemente modificare **/etc/inittab** per fare in modo che il sistema si avvii in modalita' grafica o testuale.

Se gdm o kdm e' installato sul sistema allora Ubuntu parte di default in maniera grafica, se si vuole fare in modo che parta in maniera testuale senza disinstallare kdm, basta fare in modo che kdm non venga piu' fatto partire all'avvio:

```
update-rc.d -f kdm remove
```

se non si fa altro il sistema all'avvio si blocca perche' ancora si aspetta che un qualche manager grafico entri in funzione. Bisogna quindi editare **/boot/grub/menu.lst** e cambiare le opzioni relative al kernel in modo che grub rimanga in modalita' testuale e non faccia partire lo splash grafico, per esempio:

```
/boot/vmlinuz-2.6.17-10-386  
root=UUID=e0658c23-5649-42ac-8ef2-24dc64999fd7 ro quiet splash
```

va cambiato in :

```
/boot/vmlinuz-2.6.17-10-386  
root=UUID=e0658c23-5649-42ac-8ef2-24dc64999fd7 ro quiet
```

Se invece si e' installato ubuntu-server quindi il sistema parte di default in maniera testuale e si vuole cambiare la cosa, basta installare il pacchetto virtuale relativo a kubuntu, ubuntu o xubuntu che installa tutto il sistema grafico con le applicazione di base come se si fosse installato kubuntu/ubuntu/xubuntu direttamente da cd.

```
apt-get install kubuntu-desktop
```

questa operazione si occupa anche di far partire kdm direttamente all'avvio del sistema.

Vi col comportamento di VIM su Ubuntu

Subito dopo aver installato Ubuntu se si prova ad usare **vi** ci si puo' subito accorgere che il comportamento di default e' quello "compatibile" ovvero quello molto scomodo eredita' delle vecchissime versioni di vi.

Per risolvere il problema prima di tutto e' necessario installare **vim**:

```
apt-get install vim
```

dopodiche' bisogna fare in modo che **/usr/bin/vi** punti alla versione "avanzata" di vi, questo lo si fa utilizzando il sistema degli alternatives di Ubuntu, ovvero lanciando il comando:

```
update-alternatives --config vi
```

e scegliendo tra le opzioni disponibili **/usr/bin/vim**.

Risolvere i problemi di apt-get

Purtroppo anche se **apt-get** e' un tool molto comodo che generalmente funziona molto bene, a volte puo' incappare in quale (stupido) problema dovuto a qualche piccolo errore nei pacchetti che si sta cercando di installare.

Un errore comune che mi e' capitato piu' di una volta e' questo:

```
dpkg: error processing /var/cache/apt/archives/libpthread-  
dev_2.0.7-2ubuntu2_i386.deb (--unpack):  
  trying to overwrite `/usr/include/pthread.h', which is also in  
package libc6-dev
```

che e' dovuto al fatto che il nuovo pacchetto (libpthread-dev_2.0.7-2ubuntu2_i386.deb) che si sta cercando di installare contiene uno o piu' files che vanno a sovrascrivere altri file presenti nel sistema (/usr/include/pthread.h) che appartengono a uno o piu' altri pacchetti gia' installati (libc6-dev).

Normalmente non e' grave, il file da sovrascrivere non e' importante, per andare oltre basterebbe poter dire ad apt-get di procedere con l'installazione ignorando il problema e quindi sovrascrivendo quei file.

Per fare questo bisogna usare direttamente **dpkg**, forzando l'installazione del pacchetto col file duplicato, che in questo caso e' libpthread-dev_2.0.7-2ubuntu2_i386.deb:

```
dpkg --force-all --install /var/cache/apt/archives/libpthread-  
dev_2.0.7-2ubuntu2_i386.deb
```

notare che l'argomento principale di dpkg e' il nome assoluto del file del pacchetto non solo libpthread-dev_2.0.7-2ubuntu2_i386.deb. Tale nome assoluto e' stampato a video insieme all'errore da apt-get.

Configurazione di Mldonkey

Mldonkey è un software peer to peer multiplatforma che supporta più reti p2p contemporaneamente, per esempio supporta sia Edonkey che Bittorrent.

Inoltre ha una architettura client-server, quindi permette l'esecuzione del demone su un piccolo server casalingo e del client comodamente sul proprio laptop.

Infine fornisce di default un client testuale molto comodo da usare da remoto tramite ssh.

Mldonkey può facilmente saturare la banda o la tabella NAT di un normale router casalingo, quindi va configurato attentamente.

Le opzioni “avanzate” a cui bisogna prestare attenzione in modo che la banda non venga saturata sono le seguenti:

```
max_hard_upload_rate [max_hard_upload_rate]= 10  
max_hard_download_rate [max_hard_download_rate]= 50
```

che sono nel file di configurazione downloads.ini. Possono anche essere settate da console tramite il comando **set**. I valori numerici rappresentano la massima banda concessa a Mldonkey in upload e download espressa in kBytes/s.

Inoltre è necessario configurare Mldonkey in modo che non ecceda in connessioni tcp contemporanee, per non saturare la tabella NAT del router. In questo caso le opzioni sono:

```
max_opened_connections [max_opened_connections]= 100  
max_indirect_connections [max_indirect_connections]= 30  
max_upload_slots [max_upload_slots]= 3  
max_connections_per_second [max_connections_per_second]= 3
```

anche queste sono nel file downloads.ini.

Infine è consigliabile non connettersi a più di 3 Edonkey servers contemporaneamente:

```
max_connected_servers [ED2K-max_connected_servers]= 3
```

quest'ultimo parametro è nel file donkey.ini.

Dual Monitor con schede NVidia

Se avete un portatile con una scheda grafica NVidia e volete aggiungere un proiettore o un secondo monitor, la configurazione del server X per fare funzionare sia il secondo monitor che quello del portatile non potrebbe essere piu' semplice.

Basta eseguire **nvidia-settings** con sudo (nvidia-settings e' disponibile sui repository di Ubuntu) e premere su "detect display". A questo punto il secondo monitor verra' rilevato e rimane solo da configurarlo. Basta scegliere "Twin View" come modalita' di configurazione e il gioco e' fatto!

E' possibile utilizzare il secondo monitor come un clone del primo, oppure aggiungerlo come spazio virtuale al proprio desktop e sceglierne perfino la posizione.

Vim e spellchecking

Per abilitare lo spellchecking su Vim basta eseguire il comando:

```
setlocal spell spelllang=en_us
```

invece per uscire dalla modalita' spellchecking:

```
setlocal nospell
```

Per rendere piu' veloce l'attivazione della modalita' spellchecking si possono inserire le seguenti due righe di configurazione nel file .vimrc:

```
map <F10> <Esc>:setlocal spell spelllang=en_us<CR>  
map <F11> <Esc>:setlocal nospell<CR>
```

in questo modo si associa l'attivazione al tasto F10 e la disattivazione al tasto F11.

Una volta attivata la modalita' spellcheck, si possono usare le combinazioni]s e [s per spostarsi alla successiva o alla precedente parola sbagliata. Inoltre con z= si ottiene l'elenco dei suggerimenti per la correzione.

Sudo e redirectione

Se volete eseguire un comando come root usando sudo e questo comando contiene una qualche redirectione su file, del tipo `> /path/file`, allora avrete dei problemi. Per esempio:

```
sudo cat file | sed s/parola1/parola2/g > /path/nuovofile
```

non funziona come ci si aspetta, sudo ritorna con un errore.

Per eseguire correttamente quel comando infatti bisogna usare la seguente sintassi:

```
sudo sh -c "cat file | sed s/parola1/parola2/g > /path/nuovofile"
```

in questo modo sudo crea correttamente una nuova shell e gli fa eseguire il comando tra virgolette, che la nuova shell e' perfettamente in grado di gestire.

Criptare partizioni

Il tool migliore per criptare partizioni di harddisk e' **truecrypt**. Permette di creare sia normali partizioni criptate, che richiedono una password per accedere al filesystem, che partizioni nascoste. Queste ultime si nascondono all'interno di una normale partizione criptata (sempre con truecrypt) e solo il creatore di tali partizioni puo' davvero sapere della loro esistenza.

Supponiamo che abbiate una penna usb, riconosciuta come /dev/sdb, con una sola partizione al suo interno, che volete criptare con filesystem ext3.

Quello che dovete fare e' prima di tutto eseguire:

```
truecrypt --type normal -c /dev/sdb1
```

scegliendo come filesystem **None**, quando viene richiesto, perche' ext3 non e' supportato automaticamente.

A questo punto la partizione criptata e' stata creata, ora bisogna creare il filesystem.

Eseguite:

```
truecrypt -N 1 /dev/sdb1
```

questo crea un nuovo mapping della partizione criptata, eseguite:

```
truecrypt -l
```

per sapere il percorso del volume. Ora potete creare il filesystem nel volume mappato, per esempio:

```
mkfs.ext3 /dev/mapper/truecrypt0
```

Ora non vi rimane che togliere il mapping e montare normalmente la partizione criptata:

```
truecrypt -d /dev/sdb1  
truecrypt /dev/sdb1 /mnt/tc
```

Le cose sono piu' complicate se volete creare una partizione nascosta.

In questo caso procediamo da capo creando prima la partizione esterna, che deve essere creata con filesystem **FAT**:

```
truecrypt --type normal -c /dev/sdb1
```

di seguito create quella criptata scegliendo questa volta **None** come filesystem se volete ext3:

```
truecrypt --type hidden -c /dev/sdb1
```

ora montiamo la partizione esterna in modo "sicuro" e poi copiamoci qualche file all'interno che sembri importante:

```
truecrypt -P /dev/sdb1 /mnt/tc  
cp file /mnt/tc  
truecrypt -d /dev/sdb1
```

ora creiamo il filesystem ext3 nella partizione nascosta. Eseguite i seguenti comandi **inserendo la password del volume nascosto**:

```
truecrypt -N 1 /dev/sdb1  
mkfs.ext3 /dev/mapper/truecrypt0  
truecrypt -d /dev/sdb1
```

d'ora in poi se volete montare la partizione nascosta lo potete fare normalmente con:

```
truecrypt /dev/sdb1 /mnt/tc
```

ed inserendo la password della partizione nascosta anziche' quella di quella esterna.