

# Guida Installazione Arch

by PsykeDady

AAAA-MM-DD

## Prefazione: a chi è indirizzata questa guida?

questa guida si prefigge lo scopo di essere generale un po' orientata a chiunque si avvicini la prima volta nel mondo di Archlinux. Tuttavia è stata seguita seguendo le mie esigenze e le mie esperienze acquisite nel campo. Al tempo in cui sto scrivendo questa prefazione (estate 2018) ho alle spalle soli 3 anni di abilità acquisite su archlinux, installando tuttavia più e più volte la distribuzione su vari calcolatori (pc fissi, portatili, macbook etc..)

Se c'è comunque una consapevolezza di cui il tempo, le mie e le esperienze altrui mi hanno fatto dono è che, inevitabilmente, **ogni calcolatore gode di un'esperienza unica in termini di prestazioni, estetica e praticità della configurazione distribuzione/kernel/parametri installati da colui che si appresta ad utilizzarci GNU/Linux sopra**. Questa affermazione, se pur posso assicurare abbia un alto grado di verità, tende ad entrare difficilmente nelle mentalità di chi da tempo, ormai, tende ad avere atteggiamenti da stadio anche nei confronti della più assoluta libertà che dovrebbe invece professare la community di Linux.

Inoltre consiglio a **tutti** coloro che si avventurano nella piccola impresa di installare archlinux, di tenere sempre sottomano la guida *ultima* di tutti noi arch-user (e non solo), la [wiki](#): un'enorme fonte di conoscenza sul mondo linux che risolve problemi in qualsiasi ambito, o quanto meno vi aiuta a risolverli indirettamente. Tutto ciò che troverete in questa guida altro non è che un estratto di piccole sezioni della wiki che io uso sempre per installare archlinux. La guida è inoltre disponibile in moltissime lingue, tra cui l'italiano.

Ricordo inoltre che per chi volesse provare un Archlinux con installer user-friendly, esiste [Antergos](#), una distro su base arch completamente personalizzabile al momento dell'installazione, molto ma molto user-friendly. Rimane comunque consigliato, a mio avviso, non scegliere archlinux come distribuzione per approcciarsi la prima volta con il mondo GNU/Linux, ma scegliere distro più "alla mano" come **Ubuntu**, **Fedora**, **Linux Mint** o **Deepin**.

Un'ultima considerazione: questo file è in continuo aggiornamento, motivo per il quale la data, in pagina principale, non è ancora specificata. Detto questo: buon divertimento e benvenuti nel fantastico mondo di Archlinux



# Indice

<b>1</b>	<b>Preparare il supporto di installazione</b>	<b>5</b>
1.1	Metodo 1 da Linux : dd . . . . .	5
1.2	Metodo 2 da Linux: copia su fat32 (Consigliato UEFI) . . . . .	5
1.3	Metodo 3 da Linux : varie GUI . . . . .	6
1.4	Altri S.O. . . . .	7
<b>2</b>	<b>Hello world, i'm Archlinux! Nice to meet you</b>	<b>9</b>
2.1	preparazione del disco di installazione . . . . .	10
2.2	Installazione dei pacchetti e connessione al mondo esterno . . . . .	11
2.3	prime configurazioni . . . . .	11
2.4	programma di installazione terminato . . . . .	12
<b>3</b>	<b>Configurazioni di sistema</b>	<b>13</b>
3.1	Connettiamoci al mondo esterno e alcuni consigli iniziali . . . . .	13
3.2	Server e driver . . . . .	13
3.3	Aggiunta utente, creazione cartelle utente e cifratura home . . . . .	14
3.4	configurare pacman e installare pakku . . . . .	16
3.5	Sincronizzare orologio di sistema e Hardware . . . . .	16
3.6	tmp file system . . . . .	16
3.7	Swappiness . . . . .	17
<b>4</b>	<b>Desktop Environment, Display Manager, NetworkManager e servizi systemd</b>	<b>19</b>
4.1	Plasma D.E. . . . .	19
4.2	Display Manager . . . . .	20
4.3	NetworkManager e servizi di rete . . . . .	21
4.4	netctl ed eduroam . . . . .	21
4.5	Systemd: 'la nera bestia della morte' . . . . .	22
<b>5</b>	<b>Post-Personalizzazioni di sistema by PsykeDady</b>	<b>25</b>
5.1	I sette 'nano' . . . . .	25
5.2	oh zsh, mio amore <3 . . . . .	25
5.3	fish, un ulteriore avanzatissima shell . . . . .	26
5.4	neofetch . . . . .	27
5.5	Se avete installato su VirtualBox . . . . .	28
5.6	Il COMANDONE da due milioni di dollari . . . . .	28
<b>6</b>	<b>Contatti e tanti saluti</b>	<b>31</b>



## Capitolo 1

# Preparare il supporto di installazione

### 1.1 Metodo 1 da Linux : dd

Il primo semplice metodo consiste nel preparare la pennina con il famoso tool `dd`:

```
# dd if=/percorso/iso of=/dev/sdX bs=4M status=progress
```

sostituire a X la lettera del mezzo di installazione, inserire la pennina e digitare `# fdisk -l` e quindi leggere l'output fino ad arrivare a quella che sembra essere la vostra pennina, leggendone le coordinate in seguito all'installazione, per poter riutilizzare la pennina nuovamente, dovrete azzerarla con lo stesso tool:

```
# dd if=/dev/zero of=/dev/sdXY bs=4M status=progress
```

#### NOTA BENE:

un uso intensivo di `dd` potrebbe rovinare la pennina, in quanto ne sovrascrive il contenuto bit a bit. Fare quindi attenzione, meglio utilizzare pennine a basso costo e soprattutto non molto capienti

### 1.2 Metodo 2 da Linux: copia su fat32 (Consigliato UEFI)

un altro metodo consiste nel copiare il contenuto della ISO in un file system FAT32, questo metodo presenta diversi vantaggi, ma funziona solo con macchine UEFI

creare quindi due directory dove montare la ISO e la USB di installazione:

```
mkdir {usb,iso}
```

montare la iso utilizzando:

```
# mount -o loop /percorso/iso iso
```

se non lo si è ancora fatto, formattare la pennina in fat32, supposto sia `/dev/sdX` il percorso della pennina, seguire queste istruzioni per formattarne il contenuto (effettuare le operazioni si `su` o con `sudo`):

```
dd if=/dev/zero of=/dev/sdX #da eseguire solo se si vuole
completamente resettare la pennina

fdisk /dev/sdX #si entrerà in modalità fdisk, scrivere e
premere invio i prossimi comandi
o
p
1
2048
#(invio senza scrivere nulla oppure scegliere una dimensione
)
t
```

```

b
w #si uscirà dalla modalità fdisk

#è possibile alternativamente a fdisk, usare cfdisk che è
più user-friendly

mkfs.fat /devX1

mount /dev/sdX1 usb

dosfslabel /dev/sdX1 ARCH_AAAAMM #sostituendo ad AAAA ed MM
le stesse date che trovate sulla iso

```

In questo momento, si ha nelle cartelle usb e iso, montati rispettivamente la usb e il contenuto della iso. Ci apprestiamo dunque a copiare il contenuto della iso nella usb

```
cp -r iso/* usb
```

Consiglio a questo punto di sincronizzare i dischi e smontare le cartelle, per evitare che le modifiche non vengano attuate correttamente:

```

sync
umount {usb,iso}

```

La pennina è pronta per i sistemi UEFI. Questo metodo è più sicuro di dd ma funziona su meno sistemi, eventualmente si può pensare di usare syslinux per ampliare ulteriormente il bacino di pc che vedranno la pennetta come avviabile, scaricare quindi dal proprio gestore di pacchetti l'ultima versione di **syslinux** e di **parted** ed eseguire i prossimi comandi (prima di smontare la usb):

```

extlinux --install usb/arch/boot/syslinux
dd bs=440 conv=notrunc count=1 if=/usr/lib/syslinux/bios/
mbr.bin of=/dev/sdX
parted /dev/sdX toggle 1 boot

```

#### ATTENZIONE:

quest'ultimo passo non l'ho mai applicato personalmente, ma l'ho letto sulla wiki e ve l'ho voluto riportare. Se qualcosa non dovesse funzionare, vi invito a documentarvene personalmente dalla guida ufficiale

## 1.3 Metodo 3 da Linux : varie GUI

Se non amate molto sporcarvi personalmente le mani durante queste operazioni sono disponibili moltissimi programmi che lo fanno per voi. Personalmente (ma anche la guida ufficiale) sconsiglio fortemente l'utilizzo del noto programma **uNETbootin**, in quanto tende a funzionare solo con ubuntu e derivate. Comunque sia elencherò una serie di software che ho usato io e che *spesso* funzionano:

- etcher
- suse image writer
- deepin usb maker
- Fedora media writer

## 1.4 Altri S.O.

Su sistemi operativi OSX consiglio di usare l'**utility Disco** di sistema.

Su sistemi operativi Windows invece consiglio l'uso di **Rufus** o **LiLi USB** (meno consigliato su sistemi UEFI comunque). Se vi doveste trovare male con i primi, altri utenti archlinux mi hanno consigliato i seguenti programmi:

- Win32DiskImager
- USBWriter





## Capitolo 2

# Hello world, i'm Archlinux! Nice to meet you

Supponendo ora che siate riusciti da soli ad avviare il supporto attraverso le impostazioni del vostro BIOS o le impostazioni EFI del vostro sistema, o che ancora l'abbiate avviata attraverso virtualbox e che vogliate farvi una VM con sopra archlinux, possiamo quindi passare alle prime fasi dell'installazione. Avviando arch, se tutto va bene, dovrete ritrovarvi davanti ad una schermata simile:

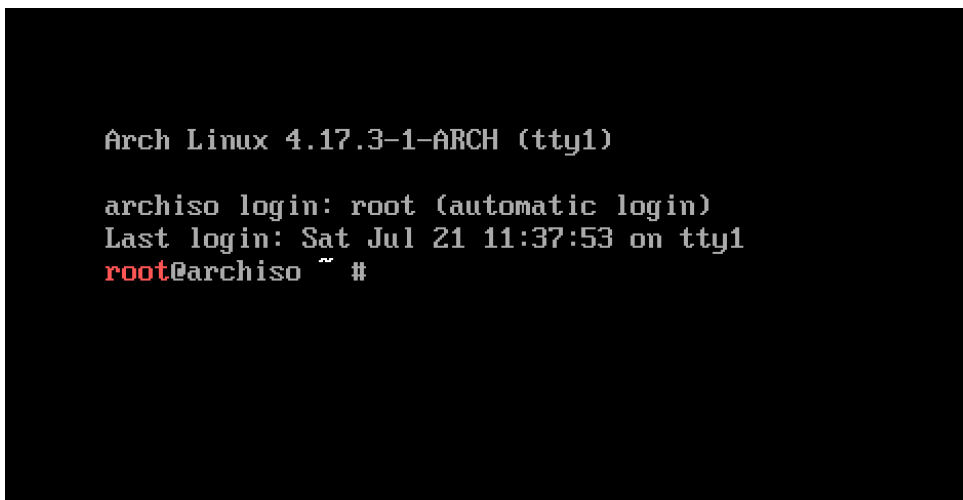
A screenshot of a terminal window with a black background and white text. The text shows the Arch Linux boot process: 'Arch Linux 4.17.3-1-ARCH (tty1)', followed by the login prompt 'archiso login: root (automatic login)', the last login information 'Last login: Sat Jul 21 11:37:53 on tty1', and the root prompt 'root@archiso ~ #'.

Figura 2.1: ecco a voi la schermata da cui tutto avrà inizio...

L'utente un po' inesperto o pratico solo di installazioni Ubuntu/OSX/Windows sarà spaesato, ma nessuna paura, è tutto molto più semplice di ciò che si pensa. Ma prima di tutto, se avete una tastiera italiana, digitate:

```
loadkeys it
```

se avete uno schermo hidpi digitate anche:

```
setfont /usr/share/kbd/consolefonts/sun12x22.psfu.gz
```

così vedrete meno madonnine volare in cielo...

### NOTA BENE:

nella cartella `/usr/share/kbd/consolefonts/` trovate molti altri font, scegliete solo quelli 12x22 per la leggibilità massima

## 2.1 preparazione del disco di installazione

La prima cosa da fare è preparare il disco di installazione, attraverso i comandi `blkid` o `fdisk -l` scopriamo quindi le coordinate del nostro disco ed eventualmente della partizione se preparata in anticipo attraverso altri sistemi operativi (può essere utile spesso preparare tutto attraverso una live di ubuntu con `gparted` se si è alle prime armi e si hanno dati da preservare).

Ci vuole comunque un po' di organizzazione, bisogna sapere in anticipo in quante partizioni si vuole suddividere la propria installazione, se si è su un sistema UEFI, se si hanno più dischi e se si hanno altre installazioni da preservare.

La guida supporrà che il disco sia inizialmente non inizializzato, sia l'unico disco e che non si hanno altri sistemi operativi presenti. Supporrò inoltre di voler suddividere l'installazione in: root, home e swap. Un'altra condizione supposta sarà quella di avere un sistema EFI con tutto quello che ne deriva.

### SUGGERIMENTO

Lo spazio di Swap è uno spazio utilizzato dal sistema per sopperire alla mancanza di memoria RAM sufficiente a far funzionare correttamente tutti i programmi. In genere si usa riservare spazio uguale alla RAM per usufruire anche della funzione di ibernazione, che consente di spegnere il computer senza perdere la sessione di lavoro corrente (diversamente dalla sospensione non consuma batteria). Per pc con RAM maggiore di 4Gb non consiglio di usare la swap a meno di usare anche l'ibernazione, un'alternativa può essere anche quella di usare il **file di swap** anziché la partizione. Maggiori informazioni si troveranno nella sezione che riguarda le configurazioni di sistema.

Abbiamo quindi il nostro disco su `/dev/sda`, vuoto e non inizializzato in alcun modo (un disco vergine per intenderci, come quello che potremmo trovarci in una macchina virtuale). Alternativamente si può pensare che abbiamo un disco di cui il contenuto non ci interessa, quindi le seguenti operazioni lo formatteranno completamente:

```
gdisk /dev/sda
```

in questo modo si entrerà in modalità `gdisk`, che installerà uno schema di partizioni di tipo GPT. se non si ha a che fare con UEFI, è consigliato usare `fdisk` o `cfdisk`, e avere a che fare con schema di partizioni tradizionale.

Come per `fdisk` e `cfdisk`, anche `gdisk` ha un'alternativa user-friendly che è `cgdisk`. Prendetela in considerazione se non volete seguire le istruzioni che seguiranno ma avere accesso ad un'interfaccia più pratica. Se avete scelto per `gdisk`, premere quindi in sequenza:

```
o
n
1
(invio senza scrivere niente)
+200M
ef00

n
2
(invio senza scrivere niente)
+XXXG
# (sostituendo a XXX il numero di Giga che volete dare alla vostra
  root)
(invio senza scrivere niente)

n
3
(invio senza scrivere niente)
+YYYG
#sostituendo a YYY il numero di Giga che volete dare alla home, in
  genere qua si mette la maggiorparte dello spazio
(invio senza scrivere niente)

n
4
```

```
(invio senza scrivere niente)
+ZZG
#sostituendo a ZZ il numero di Giga che volete dare alla swap.
8200

w
```

dopo essere usciti dalla modalità `gdisk`, possiamo accertarci della situazione usando il comando: `gdisk -l` oppure con `blkid`. Per usare le partizioni comunque è necessario formattarle

```
mkfs.fat /dev/sda1
mkfs.ext4 /dev/sda2
mkfs.ext4 /dev/sda3
mkswap /dev/sda4
```

dunque possiamo iniziare a montarle:

```
mount /dev/sda2 /mnt
mkdir -p /mnt/boot/efi
mkdir /mnt/home
mount /dev/sda1 /mnt/boot/efi
mount /dev/sda3 /mnt/home
swapon /dev/sda4
```

La sezione riguardante la configurazione dei dischi finisce qua.

## 2.2 Installazione dei pacchetti e connessione al mondo esterno

Stop. senza internet non si va da nessuna parte.

Se quindi siete connessi via cavo, basta dare `dhcpcd`, altrimenti archlinux fornisce una comodissima interfaccia di rete wireless, a cui potete accedere così:

```
wifi-menu
```

Scegliete quindi il vostro SSID di fiducia, scrivete la password (se ne avete una) e date `dhcpcd` per forzare il router a rilasciarvi un indirizzo ip. Ben fatto, siete connessi! Sicuri? Per accertarcene possiamo dare

```
ping -c 3 www.google.com
```

se tutto va bene, vi risponderà che son stati trasmessi e ricevuti 3 pacchetti. Ora che siamo sicuri possiamo andare avanti.

Arch offre un modo davvero comodo di installare i pacchetti iniziali sulle nuove installazioni, attraverso lo script `pacstrap`:

```
pacstrap /mnt base base-devel net-tools dialog netctl wpa_supplicant grub efibootmgr
```

se tutto va per il meglio (spero per voi) il vostro `/mnt` sarà adesso abbastanza popolato.

## 2.3 prime configurazioni

creiamo quindi il vostro `fstab` che permetterà di montare le cartelle nel giusto ordine all'avvio:

```
genfstab -pU /mnt > /mnt/etc/fstab
```

ed entriamo quindi in `chroot` attraverso un comodissimo script `arch`:

```
arch-chroot /mnt
```

settiamo la password di root:

```
passwd
```

e modifichiamo il file `fstab` creato in precedenza sostituendo nella riga della swap, il parametro `none` scrivendo **swap**  
configuriamo quindi il grub:

```
grub-mkconfig -o /boot/grub/grub.cfg
grub-install /dev/sda
```

possiamo quindi dare un nome alla nostra macchina in rete:

```
echo "NOMEPC" > /etc/hostname
```

Impostiamo quindi la lingua: andando ad editare (con `nano`, `vi` o qualunque altro editor ci piace usare) il file `/etc/locale.gen` decommentando le tre linee che iniziano con **it\_IT**, successivamente diamo il comando

```
locale-gen
```

successivamente generiamo un buon `/etc/locale.conf`, usiamo il nostro editor di testo preferito e scriviamo:

```
LANG=it_IT.UTF-8
LC_COLLATE="C"
LC_TIME="it_IT.UTF-8"
LANGUAGE="it_IT:en_GB:en"
```

impostiamo la lingua del tty con:

```
echo "KEYMAP=it" > /etc/vconsole.conf
```

e impostiamo il fuso orario di sistema con:

```
ln -sf /usr/share/zoneinfo/Europe/Rome /etc/localtime
```

## 2.4 programma di installazione terminato

Il sistema è installato correttamente, adesso andiamo per smontare le partizioni:

```
exit
umount /mnt/boot/efi
umount /mnt/home
umount /mnt
swapoff
sync
poweroff # o reboot per riavviare
```

possiamo quindi riavviare e continuare con le configurazioni tramite l'utente `root` e non in ambiente di `chroot`

## Capitolo 3

# Configurazioni di sistema

Dopo aver riavviato e tolto la chiavetta, dovremmo trovare la possibilità di avviare archlinux tramite grub, se così non fosse reinserite la chiavetta, rimontate le partizioni e rifate il chroot, cercando la soluzione al problema tramite la guida wiki. Da qui in poi sarà supposto che voi abbiate il sistema funzionante e possiate fare il login tramite account di root.

Inserite quindi come nome *root* e come password quella impostata durante le configurazioni precedenti. È quindi tempo di fare un po' di configurazioni a sistema appena installato!

### 3.1 Connettiamoci al mondo esterno e alcuni consigli iniziali

Come sempre la prima cosa da fare è connettersi. lo si può fare esattamente come prima tramite `dhcpcd` e nel caso di una wifi `wifi-menu`.

Il primo consiglio che innanzitutto do è quello di eseguire subito un upgrade del sistema e dei repository:

```
pacman -Syu
```

Consiglio poi di installare alcuni pacchetti che nel 90% dei casi vi saranno utili

```
pacman -S linux-headers os-prober git bash-completion
```

Nello specifico, os-prober vi serve nel caso in cui pianificate ( o abbiate ) una macchina con più sistemi operativi installati.

### 3.2 Server e driver

In ambienti linux, senza motore grafico, possiamo usare il pc al più come server. Oggi giorno la produttività dipende strettamente da ciò che vediamo e come interagiamo. Per questo motivo installiamo il server grafico, oggi Xorg, nonostante ci siano valide alternative, risulta ancora la realtà più consolidata, vi consiglio dunque di installarlo a prescindere da ciò che poi proverete. Insieme a Xorg, è consigliato installare il suo sistema di init, utile nel caso in cui non vogliate un DM o il vostro non funzioni a dovere. Quindi:

```
pacman -S xorg-server xorg-xinit
```

Potete quindi impostare nel file `~/xinitrc` il comando per utilizzare il vostro DE preferito (quando ne avrete uno).

Installare i driver è anche abbastanza semplice, se avete installato arch su una macchina virtuale virtualbox, vi basterà digitare:

```
pacman -S virtualbox-guest-utils
```

altrimenti andiamo ad identificare la vostra scheda video con `lspci`

```
lspci | grep VGA
```

l'output riporterà al suo interno: *intel*, *ati* o *nvidia*. In base a cosa riporta andiamo ad installare i driver *open* relativi:

```
#per installare i driver intel
pacman -S xf86-video-intel

#per i driver ati
pacman -S xf86-video-ati

#per nvidia
pacman -S xf86-video-nouveau
```

per installare i driver proprietari vi invito invece a visitare la wiki relativa.

Alcune volte può essere necessario installare i vecchi driver synaptics per il touchpad, tanto meglio nel caso averli già pronti:

```
pacman -S xf86-input-synaptics
```

### 3.3 Aggiunta utente, creazione cartelle utente e cifratura home

È sempre bene utilizzare l'account root solo se strettamente necessario, altrimenti è meglio impostare un utente amministratore o semplice (ancora meglio).

per aggiunta di un utente amministratore digitare:

```
useradd -m -g users -G wheel,video,audio,sys,lp,storage,scanner,games,network,disk,input -s /bin/bash <nome utente>
```

per un amministratore non utente basta eliminare il gruppo **wheel** dal codice precedente.

Impostiamo quindi una password per l'utente appena creato:

```
passwd <nome utente>
```

e indichiamo al programma sudo ( che ci permette di effettuare operazioni in modalità amministratore) tramite *visudo*:

```
#impostiamo prima un editor di testo a noi piu' amichevole, di
  default e' vi
export EDITOR=<nome editor>
visudo
```

a questo punto esistono due modi di impostare i permessi di amministratore, con richiesta della password (consigliato) e senza richiesta.

nel primo caso decommentare la riga:

```
wheel ALL=(ALL) ALL
```

nel secondo decommentare:

```
wheel ALL=(ALL) NOPASSWD: ALL
```

A questo punto configuriamo le cartelle utente, installiamo

```
pacman -S xdg-user-dirs
```

. Al nostro accesso con l'utente digitiamo

```
xdg-user-dirs-update
```

e ci dovremmo trovare nella home tutte le cartelle utente. Nel caso non siano in italiano si può editare il file */home/<nome utente>/.config/user-dirs.dirs* inserendo ad uno ad uno i nomi che desideriamo sostituire.

---

**Il prossimo step è opzionale, se non volete eseguirlo passate direttamente alla prossima sezione:**

potete decidere di cifrare il contenuto della vostra home, un po' come succede nei telefoni che possiedono metodo di sblocco con impronta digitale. Per farlo la prima cosa è installare alcuni pacchetti

```
pacman -S ecryptfs-utils keyutils rsync lsof
```

Quindi caricare l'apposito modulo del kernel:

```
modprobe ecryptfs
```

. Potrebbe essere necessario apportare una modifica al file `/etc/mkinitcpio.conf` e scriverci all'interno, nella sezione **MODULES**, il nome del modulo per forzarne il caricamento ad ogni avvio del pc. Successivamente ricompilare il servizio di avvio

```
mkinitcpio -p linux
```

Usare quindi il tool per la migrazione della home, per avviare questa fase è necessario che voi non abbiate alcun processo aperto con l'utente di cui volete cifrare la home:

```
ecryptfs-migrate-home -u <nome utente>
```

Seguire le istruzioni indicate. Per completare la procedura, uscite dal vostro account root con `exit` ed entrate con quello dell'utente. Verificate quindi con `ls` che siano state criptate tutte le cartelle (dovrebbe apparire `Access-your-data.desktop` e un altro file di testo)

Quindi decriptate e ri-criptate voi stessi la home usando i due tool

```
ecryptfs-mount-private #per decriptare
ecryptfs-umount-private #per ricriptare
```

Potete usare i due comandi ogni volta che volete cifrare o decifrare la cartella home manualmente, può accadere alle volte che la cifratura non avvenga per processi aperti su file all'interno della home. Si può quindi forzare il procedimento con questo comando

```
umount.ecryptfs_private
```

uscite dall'account user (dopo aver smontato la cartella) e rientrate con root per maggiore comodità.

Ora è necessario (a meno che non vogliate farlo a mano ogni accesso) impostare l'auto-mounting della home criptata all'accesso. Facciamo un backup del file `/etc/pam.d/system-auth`

```
cp /etc/pam.d/system-auth /etc/pam.d/system-auth.old
```

e apriamo `/etc/pam.d/system-auth` con il nostro editor preferito.

Da adesso facciamo **MOLTA** attenzione, sbagliando qualunque cosa potremmo non poter più accedere a nessun account (a meno di aggiustare poi le cose con l'iso di arch), andiamo ad aggiungere dopo la stringa che contiene **auth required pam\_unix.so** la seguente linea:

```
auth required pam_ecryptfs.so unwrap
```

Dopo la linea che contiene **password required pam\_unix.so** aggiungiamo:

```
password optional pam_ecryptfs.so
```

E infine dopo la linea che contiene **session required pam\_unix.so** aggiungiamo:

```
session required pam_ecryptfs.so unwrap
```

Usciamo dall'editor salvando. Per essere sicuri di aver fatto le cose a modo, apriamo un altro tty ( `ctrl-alt-f2` ) e facciamo l'accesso con l'utente. Se l'accesso avviene correttamente, e se la cartella viene correttamente decriptata, allora è tutto ok. Altrimenti ritornate immediatamente nel primo tty ( `ctrl-alt-f1` ) correggete l'errore nel file o, nel caso non ci riusciate, eliminate tutte le modifiche facendo tornare il file allo stato originale attraverso il backup. Se tutto è andato a buon fine, ricordate di far uscire con `exit` l'utente. Se la cartella non viene ricriptata potreste avere problemi di accesso d'ora in poi, nel caso entrate con il tty e ricriptatela con il comando di `umount`. Ripassate quindi al tty per continuare con l'installazione.

---

### 3.4 configurare pacman e installare pakku

Perché Archlinux? Perché complicarsi la vita con questa tortura che ti porta a perdere una giornata per l'installazione di un sistema operativo? Le risposte sono tante, ma la prima in assoluto è il gestore di pacchetti **pacman** e tutto ciò che ne deriva, compreso il famoso **AUR: Arch User Repository**.

Prima di tutto, abbelliamo il nostro pacman! Sempre con il nostro editor preferito (a proposito, il mio è nano, vi insegnerò anche a renderlo carino) modifichiamo il file `/etc/pacman.conf`: andiamo a decommentare la riga con scritto **Color** e aggiungiamo sotto **ILoveCandy**. Poi decommentiamo dove c'è scritto **multilib** e la riga di sotto se vogliamo abilitare il supporto alle librerie a 32 bit (necessario per alcuni programmi). Se volessimo aggiungere un nuovo repository lo possiamo fare seguendo il template alla fine del file, ma difficilmente ne avrete bisogno dopo che installerete un **aur-helper**.

Installiamone quindi uno: **pakku**! Per lo step successivo, è fortemente consigliato l'accesso con l'utente amministratore e non con root.

```
git clone https://aur.archlinux.org/pakku.git
makepkg -si
```

**pakku** usa la stessa sintassi di **pacman**, e potete sostituirlo in tutto e per tutto al package manager, l'unica differenza sta nel fatto che cerca pacchetti anche su **AUR**, un immenso repository di pacchetti offerti dalla comunità, ci trovate davvero di tutto dentro. **Ma non va eseguito come utente amministratore (con sudo per intenderci) né come utente root**. Tramite pakku potete visualizzare la differenza tra versione presente e quella aggiornata del software oppure visionare e modificare il PKGBUILD, è uno strumento molto potente!

Pakku è scritto in NIM, un linguaggio di programmazione ad alto livello multi piattaforma, con aspetti molto molto interessante :) se siete del settore vi consiglio di leggerne qualcosa sul web

La fase iniziale di installazione è terminata, consiglio a questo punto di riavviare prima di continuare.

### 3.5 Sincronizzare orologio di sistema e Hardware

Per sincronizzare l'orologio di sistema con quello del calcolatore si può digitare:

```
hwclock --systohc (--utc)
```

### 3.6 tmp file system

nel nostro file system una cartella speciale, `/tmp`, che contiene file e cartelle temporaneamente creati dai programmi per il loro corretto funzionamento. Questa cartella viene montata da *systemd* automaticamente in memoria **RAM** in modo da essere resettata ad ogni arresto del calcolatore.

È comunque possibile, per quei pc che non hanno un gran quantitativo di memoria disponibile, montare la cartella nello spazio di archiviazione piuttosto che in memoria volatile

#### ATTENZIONE:

Se installate frequentemente programmi da **AUR** con un **AUR-helper** è possibile che quest'ultimo, facendo tante scritture su `/tmp`, vi saturi facilmente la memoria, consiglio quindi questo procedimento a tutti i pc con meno di 8 GB di RAM.

Per disabilitare il montaggio automatico basta dare:

```
# systemctl mask tmp.mount
```

Bisogna comunque tener conto che disabilitando questo comportamento, il contenuto della cartella dei file temporanei **non verrà più cancellato allo spegnimento del dispositivo**.

Si può ritornare al comportamento di default specificando manualmente il montaggio in `/etc/fstab` aggiungendo questa linea di testo al file:

```
tmpfs /tmp tmpfs nodev,nosuid 0 0
```

Si può anche specificare una size massima per evitare che la ram venga monopolizzata dalla sola partizione di file temporanei, facendo un esempio con il limite di **2GB** avremmo:



```
tmpfs /tmp tmpfs nodev,nosuid,size=2G 0 0
```

Altre interessanti informazioni su **tmpfs** si trovano sulla guida di arch.

## 3.7 Swappiness

Se avete installato un area di swap la vostra maggior paura sarà appunto quella che il vostro pc possa usarla senza che ce ne sia la reale necessità; non è infatti detto che il sistema operativo debba aspettare di riempire la RAM prima di usare l'area di SWAP.

In ogni caso questo aspetto del sistema si può benissimo controllare attraverso la [swappiness](#)

Si può impostare la swappiness temporaneamente o permanentemente, nel secondo caso basta modificare il file [/etc/sysctl.d/99-sysctl.conf](#) e scrivere:

```
vm.swappiness=1
```

Come funziona questo valore però? perchè 1 e non 0?

Il valore di swappiness va da [0](#) a [100](#) e indica il **valore di ram libera che si desidera tenere libero prima di attivare la zona SWAP**. Ad esempio, normalmente il valore di swappiness è **60**, questo significa che oltre il 40% di RAM occupata, l'area di SWAP si attiva ( non finisce tutto in swap ovviamente ma il sistema tende ad assestarsi su quel valore di RAM libera). Nelle versioni antecedenti a linux 3.5, un valore di swappiness [0](#) significava "*usa la swap solo in caso di stretta necessità*", oggi ne disattiva invece le funzioni. Maggiori informazioni [qui](#) e [qui](#).



## Capitolo 4

# Desktop Environment, Display Manager, NetworkManager e servizi systemd

Premessa: ad oggi l'unico Desktop Environment che **non** mi sento di consigliare, è **GNOME**. Molti prenderanno questa come una bestemmia e chiuderanno subito la guida, fatti loro. A mio parere **GNOME** ha ancora molte mancanze che mi fanno sempre desistere dal tenerlo installato sui miei sistemi, è inoltre l'unico sistema che, appena aperto, raggiunge il Gigabyte di ram occupata (più o meno). In ogni caso la bellissima guida wiki ha una guida per ogni singolo DE che vogliate installare (e installare **GNOME** completo è davvero cosa di due comandi).

La qui presente guida invece vi insegnerà a installare **Plasma**, un DE molto avanzato con consumi ram/cpu nella media.

Per sistemi poco prestanti consiglio **XFCE**, **LXDE** o **I3WM**, che sono tre validissime alternative che con qualche tocco di eleganza hanno fatto la storia delle configurazioni e dei temi più belli del mondo Linux.

Per chi ha uno schermo HIDPI consiglio ancora **PlasmaDE**, **Cinnamon** o **GNOME** (*per i più impavidi che non hanno desistito dopo la mia precedente descrizione*) in quanto sono gli unici tre ad offrire un supporto nativo. In realtà ci sarebbe anche **Enlightenment**, ma lo reputo un DE con qualche difetto di troppo a gestire alcuni tipi di applicazioni (come ad esempio il noto I.M. **Telegram**).

### NOTA BENE:

D'ora in poi potrete operare tranquillamente con l'account root così come dall'account utente, usando **sudo** lì dove viene richiesta la modalità di amministrazione.

Ogni qualvolta in un codice della guida vedrete il simbolo **#**, dovrete scrivere quel comando come amministratori.

## 4.1 Plasma D.E.

In realtà installare un DE è spesso tempo di uno o due comandi, ad esempio per installare plasma completo, con tanto di applicazioni kde, basta digitare:

```
# pacman -S plasma kde-applications
```

per un installazione minimale invece digitare:

```
# pacman -S plasma-desktop
```

Questo installerà solo l'ambiente, poi dovrete selezionare voi ad uno ad uno i software che volete (file manager, browser, ecc).

Consiglio fortemente, se avete uno smartphone android, di installare kde-connect e associarlo all'omonima applicazione smartphone:

```
# pacman -S kdeconnect
```

Un altro consiglio è quello di installare l'integrazione browser di plasma, che vi consente di essere notificati quando un download finisce o di integrare il gestore multimediale di plasma al browser, il che vi permetterà di fermare la musica direttamente dalle notifiche ad esempio. Per installarlo:

```
# pacman -S plasma-browser-integration
```

Se usate la lingua italiana potete installarne il supporto con:

```
# pacman -S kde-l10n-it
```

Se, come me, siete amanti delle dock, potete installare e personalizzare latte-dock, fatta apposta per plasma:

```
# pacman -S latte-dock
```

#### NOTA BENE:

Su molti Desktop Environment (plasma incluso) la tastiera è impostata di default con il layout USA. Non è un problema di localizzazione, ma va risolto proprio dalle impostazioni della tastiera del DE.

Se avete problemi con plasma, ma non volete riavviare il pc per risolverli, potete tentare il riavvio del solo D.E. premendo `alt-f2` e digitando quanto segue:

```
killall plasma && killall kwin_x11 && kstart plasmashell && kstart kwin_x11
```

e premendo quindi invio.

## 4.2 Display Manager

Insieme a plasma, nella sua versione completa, verrà installato il suo *D.M. (Display Manager)* **SDDM**. Un D.M. banalmente vi presenta quella schermata che vi permette facilmente l'accesso al vostro ambiente, avviando anche il server X (o qualunque altro server grafico usiate). Ciò avviene facendovi selezionare il vostro utente da GUI, insieme ad un ambiente desktop e inserendo la password. Alcuni D.M. vi impongono di scrivere anche il nome utente. In quel caso diventano banalmente delle alternative a xinit.

Sì, anche xinit può essere visto come un Display Manager da questo punto di vista, anche se xinit formalmente è un servizio che vi permette di avviare manualmente il server X, seguito da un comando che consente di avviare un DE a vostra scelta. Il comando in questione deve essere inserito nel file nascosto `~/.xinitrc` inserito poi nella **home dell'utente**.

Per utilizzare xinit è necessario effettuare l'accesso da tty con l'utente scelto, e digitare `startx`.

Per utilizzare invece un D.M. completo, dovrete abilitarlo come servizio di systemd.

Tornando a SDDM ad esempio, per abilitarlo dovrete scrivere:

```
# systemctl enable sddm
```

E riavviare.

In generale, al posto di sddm, dovrete scrivere il DM da voi scelto (ad es. *gdm*, *lightdm*, *slimdm*, ecc..)

Un'alternativa potrebbe essere quella di usare **xinit**, cioè un servizio che avvia il server x e, in seguito, un DE da voi scelto. Questa procedura è consigliata a chi ha poche risorse grafiche, a chi vuole ridurre al minimo i tempi di accesso o chi deve semplificare al più possibile l'accesso per isolare e capire dei problemi.

Per accedere tramite servizio **xinit** dovete assicurarvi di avere installato il pacchetto `xorg-xinit`. In seguito dovete scrivere file `~/.xinitrc` (varia per ogni utente dunque):

Per KDE:

```
exec startkde
```

Per XFCE:

```
exec startxfce4
```

Per GNOME con X Server:

```
export GDK_BACKEND=x11
exec gnome-session
```

Per altri DE, consultare le relative wiki!

## 4.3 NetworkManager e servizi di rete

Dietro le quinte, quando nella guida si è visto l'uso di *wifi-menu*, il software che vi permetteva di accedere ad internet nient'altro era che **netctl**. Se avete seguito la guida dall'inizio, sia **netctl** che *wifi-menu* continueranno a funzionare egregiamente. Tuttavia spesso, è utile avere a che fare con servizi più user-friendly, che vi notificino ad esempio quando cade la connessione, vi mostrino costantemente il segnale e vi facilitino quando dovete inserire le credenziali, proteggendone anche il contenuto se siete in pubblico. Il servizio per eccellenza è **NetworkManager**.

### NOTA BENE:

Usando NetworkManager, disabiliterai **netctl**. Tentando di usare **netctl** quando **nm** è attivo, causerai degli errori. Utilizza **systemd** per gestire il passaggio da un servizio all'altro se dovessi necessitare di preferirne uno in particolari situazioni.

Quindi per installare **networkmanager**:

```
# pacman -S networkmanager
```

Se eventualmente navigate sotto reti d'azienda o sotto reti istituzionali come **eduroam**, vi potrebbero servire dei pacchetti aggiuntivi, in tal caso:

```
# pacman -S networkmanager-pptp networkmanager-vpnc
```

Alcuni tra i DE più avanzati si porta dietro, nella sua installazione completa, l'applet che consente di monitorare e connettersi attraverso il pannello delle notifiche. Se così non fosse la pagina wiki-arch di NetworkManager spiega come interlacciare il servizio al DE. Nei casi più comuni comunque, viene usato quello di **gnome**

```
# pacman -S network-manager-applet
```

NetworkManager installa un apposito servizio **systemd**, che va abilitato all'avvio e gestito tramite **systemctl**

## 4.4 netctl ed eduroam

Quello che spesso fa desistere dall'usare **netctl**, che sfrutta molto meglio i driver delle periferiche di rete e causa molti meno errori, è il fatto che per connettersi a reti aziendali bisogna personalizzare i file di configurazione a mano.

Per mie esigenze personali ho dovuto cercare un modo di connettere **netctl** ad **eduroam**, e ho creato quindi una procedura che consente facilmente di connettervi ad esso:

Innanzitutto tentate una connessione con **wifi-menu**, inserendo una password anche a caso. Quando vi dirà che la connessione è fallita, dite di voler tenere il file di configurazione.

Andate quindi a modificare manualmente il file generato:

```
# <editor di testo preferito> /etc/netctl/wlp3s0-eduroam
```

## NOTA BENE:

Al posto di `wlp3s0` potrebbe esserci scritto altro, questo dipende da come il sistema chiama la vostra interfaccia di rete. Per scoprirlo si può usare il tool `ip link` o `ifconfig`.

Quindi scrivere all'interno del file:

```
Description='Automatically generated profile by wifi-menu'
Interface=wlp3s0
Connection=wireless
Security=wpa-configsection
ESSID=eduroam
IP=dhcp
WPAConfigSection=(
'ssid="eduroam" '
'proto=RSN '
'key_mgmt=WPA-EAP '
'eap=PEAP '
'identity="SCRIVERE QUI IL PROPRIO NOME UTENTE DI EDUROAM" '
'password="SCRIVERE QUI LA PASSWORD DEL PROPRIO EDUROAM" '
'phase2="auth=MSCHAPV2" '
)
```

**eduroam** è una realtà abbastanza consolidata con direttive precise, motivo per cui non dovrebbero esserci differenze tra le configurazioni qui scritte e la vostra. Tuttavia la struttura del file è semplice da capire, quindi posso supporre che possiate anche immaginare dove mettere mano se qualcosa dovesse essere diverso.

## 4.5 Systemd: ‘la nera bestia della morte’

Systemd è un insieme di tool che gestiscono servizi e avvio del vostro sistema operativo su base Linux. Non entrerò nel dettaglio spiegando perché molti lo odiano, perché altri lo amano e perché invece altri ancora, come il sottoscritto, se ne fregano altamente, basta che funzionino.

Vi spiegherò invece come interfacciarvi al gestore facilmente, elencando una serie di comandi e spiegandone l'uso.

comando	spiegazione
<code># systemctl enable &lt;nome&gt;</code>	abilita il servizio all'avvio, che viene quindi attivato ogni qualvolta accendete il vostro calcolatore
<code># systemctl start &lt;nome&gt;</code>	avvia immediatamente il servizio
<code># systemctl restart &lt;nome&gt;</code>	spegne e riavvia il servizio, utile se si stanno sperimentando problemi
<code># systemctl stop &lt;nome&gt;</code>	spegne il servizio, il contrario di <i>start</i>
<code># systemctl disable &lt;nome&gt;</code>	disabilita il servizio che non viene più avviato all'accensione del calcolatore, il contrario di <i>enable</i>
<code>systemctl status &lt;nome&gt;</code>	controlla lo stato del servizio, se è attivo, in errore o fermo.
<code>systemctl poweroff</code>	spegne il sistema
<code>systemctl reboot</code>	riavvia il sistema
<code>systemctl hibernate</code>	iberna il sistema, da usare solo se avete attivato l'ibernazione in modo corretto.
<code>systemctl suspend</code>	sospende il sistema.
<code>systemctl suspend-then-hibernate</code>	sospende per un periodo di tempo, successivamente iberna
<code>systemctl hybrid-sleep</code>	sospende il sistema, se la batteria arriva a livello critico durante la sospensione, iberna.

Prendiamo ad esempio che vogliate usare nuovamente *netctl* anziché *NetworkManager*. La procedura corretta sarebbe:

```
systemctl stop NetworkManager
systemctl start netctl
wifi-menu
dhcpcd
```

Al contrario invece:

```
dhcpcd -x  
systemctl stop netctl  
systemctl start NetworkManager
```

**ATTENZIONE:**

Spesso NetworkManager, anche a servizio spento, prende il sopravvento perchè il plugin ( in background tramite il DE ) resta attivo. In tal caso bisogna ripetere la procedura più volte se si vuole usare netctl.





## Capitolo 5

# Post-Personalizzazioni di sistema by PsykeDady

Seguiranno alcune personalizzazioni tipiche dei miei sistemi. Questo capitolo è assolutamente opzionale e non necessaria al funzionamento del sistema.

### 5.1 I sette ‘nano’

da linea di comando, il mio editor preferito è nano. Premesso che io penso che la linea di comando serva giusto per piccole modifiche, non uso editor che consentono, anche da terminale, di progettare grandi sistemi (si veda VIM).

Detto ciò, piccole modifiche non significa che non si debba stare comodi no? quindi vediamo come fare a rendere più piacevole l'uso di nano.

Creare con il proprio editor preferito il file `~/nanorc`:

```
set softwrap
set autoindent

set linenumbers

include "/usr/share/nano/*.nanorc"
```

**softwrap:** fa andare a capo le righe che superano la lunghezza del terminale, così non vi dovrete spostare per vedere cosa contengono quelle lunghe infinite linee che scriverete

**autoindent:** se programmate con nano, questo può essere un aiuto davvero importante. Abilita l'auto-indentazione, che vi permette, una volta che avete indentato il codice, di mantenere sulle righe di sotto la stessa tabulazione precedente. Se programmate e non indentate, o siete figli di satana o vi volete davvero male o ancora, siete alle prime armi (e in questo caso vi perdono).

**linenumbers:** una volta che avete abilitato softwrap non capirete quando inizia una riga e quando invece sta continuando quella precedente. Questo parametro evidenzia quindi i numeri di riga, cioè ad ogni riga vi dice quale riga è in termini di numeri cardinali

**include /bla/bla/bla:** permette l'evidenziazione della sintassi dei linguaggi di programmazione supportati da nano. Il percorso indicato è dove normalmente sono salvati i template che descrivono come evidenziare la sintassi di quel linguaggio. Su alcuni sistemi potrebbe essere diverso

### 5.2 oh zsh, mio amore <3

*il terminale è vita, il terminale è amore.* Spesso per l'utilizzatore linux, il terminale è davvero tutto. Ecco perchè abbellirlo e renderlo funzionare dovrebbe essere la prima cosa da fare per tutti coloro che si avventurano nell'utilizzo dei sistemi operativi del pinguino. Allora ecco che entrano in gioco le 'alternative'

alla shell per eccellenza, sua maestà *bash*. La prima alternativa, nonché la più utilizzata credo, è *zsh*. In realtà avete già usato *zsh* ma non lo sapete(?). Infatti la live di archlinux la usa. Quindi installiamo *zsh* e anche il famosissimo *oh-my-zsh*, gestore dei plugin di *zsh*. Qui entra finalmente in gioco *AUR*, quindi dovrete fare queste operazioni da account utente e non come amministratori:

```
pacman -S zsh oh-my-zsh-git zsh-theme-powerlevel9k
```

Il primo pacchetto indicato è ovviamente *zsh*, il secondo è *oh-my-zsh* e il terzo è uno dei temi più famosi per *zsh*. In realtà solo il secondo pacchetto si trova su *AUR*, gli altri dovrete trovarli nei repo standard.

Una volta installato tutto dobbiamo apportare alcune modifiche...

Copiamo innanzitutto il file rc di *oh-my-zsh*:

```
cp /usr/share/oh-my-zsh/zshrc ~/.zshrc
```

Modifichiamo quindi il file appena copiato con il nostro editor preferito. Andiamo alla linea `ZSH_THEME` e scriviamo

```
ZSH_THEME="powerlevel9k/powerlevel9k"
```

per funzionare comunque bisogna anche copiare il tema nella cartella temi di *zsh*:

```
# cp -r /usr/share/zsh-theme-powerlevel9k/ /usr/share/oh-my-zsh/themes/powerlevel9k
```

Potete comunque anche fare un link alla cartella con `ln`.

Adesso, se abbiamo deciso che il nostro sistema ha localizzazione italiana, dobbiamo leggermente modificare il tema. Allo scopo consiglio di usare un editor di testo che permetta di modificare più linee alla volta con la funzione cerca e sostituisci. Apriamo con permessi di amministrazione il file `/usr/share/oh-my-zsh/themes/powerlevel9k/function/icons.zsh` ed eliminiamo o commentiamo tutte le righe che iniziano con `local LC_`.

Il nostro *zsh* dovrebbe essere pronto, possiamo testarlo digitando *zsh* e, eventualmente, impostarlo come shell predefinita con

```
chsh -s /usr/bin/zsh
```

### 5.3 fish, un ulteriore avanzatissima shell

Ancora più avanzata è la shell **fish**, che possiede per altro un proprio linguaggio di scripting, diverso da quello di *bash*.

Lo possiamo installare digitando

```
# pacman -S fish
```

*Fish* è un po' particolare da tanti punti di vista, ad esempio il suo file rc lo trovate in `$USER/.config/config.fish`, e per personalizzarlo dovrete usare **fish\_config**, che aprirà un server web sul vostro browser dove potrete scegliere il tema, il prompt e altre funzionalità.

Un'altra particolare funzione di *fish* è il suo saluto di benvenuto, che può piacere come no. Per ridefinirlo scrivere nel file rc di *fish*:

```
function fish_greeting
    #...scrivere qui il codice o lasciare vuoto se si vuole
    disattivare
end
```

per cambiare shell predefinita si può digitare

```
chsh -s /usr/bin/fish
```

## 5.4 neofetch

Agli utenti linux piace avere tutto sottocontrollo, e soprattutto piace che questo continuo monitorare sia esteticamente appagante. Ecco perché, chi per noi, ha creato dei tool che mostrano in modo sgargiante quello tutte le informazioni di cui abbiamo bisogno. Uno di questi è **neofetch**, che possiamo semplicemente installare con

```
# pacman -S neofetch
```

Avviamolo almeno una volta scrivendo **neofetch** e poi andiamo a modificare `~/.config/neofetch/config.conf`. Il mio ad esempio ha nella sezione print queste informazioni:

```
print_info() {
    info title
    info underline

    info "OS" distro
    info "Host" model
    info "Kernel" kernel
    info "Uptime" uptime
    info "Packages" packages
    info "Shell" shell
    info "Resolution" resolution
    info "DE" de
    info "WM" wm
    info "WM Theme" wm_theme
    info "Theme" theme
    info "Icons" icons
    info "Terminal" term
    info "Terminal Font" term_font
    info "CPU" cpu
    info "GPU" gpu
    info "Memory" memory
    info underline
    # info "GPU Driver" gpu_driver # Linux/macOS only
    # info "CPU Usage" cpu_usage
    info "Disk" disk
    info "Battery" battery
    # info "Font" font
    info "Song" song
    # info "Local IP" local_ip
    # info "Public IP" public_ip
    # info "Users" users
    # info "Install Date" install_date
    # info "Locale" locale # This only works on glibc systems.

    info line_break
    info cols
    info line_break
}
```

Ma i veri avventori sanno che neofetch può fare molto di più... sulla wiki potrete trovare qualcosa. Per far partire il programma ogni qual volta aprite il terminale (altra cosa che piace molto in genere) potete inserire **neofetch** nell'rc del vostro terminale preferito ( `~/.bashrc`, `~/.zshrc`, `~/.config/fish/config.fish`, etc... )

Funzioni aggiuntive di neofetch vengono attivate installando questi pacchetti:

```
pacman -S catimg feh imagemagick jp2a libcaca nitrogen pacman-contrib w3m xdotool
```

## 5.5 Se avete installato su VirtualBox

Se avete installato il tutto su virtual box sicuramente avrete anche qualche cartella da condividere tra sistema e ospite e sistema ospite, quindi potete montarlo così:

```
#creiamo una cartella dove montare i dati condivisi
sudo mkdir -p /media/vboxshare

#quindi montiamola, supponiamo di aver chiamato dalle impostazioni
di VirtualBox la cartella SHARED
sudo mount -t vboxsf SHARED /media/vboxshare
```

È in generale possibile, attraverso la modifica del file `/etc/fstab`, aggiungere un punto nuovo di montaggio da caricare all'avvio del sistema. Nel particolare possiamo farlo anche con la cartella di virtual box.

Apriamo quindi con il nostro editor preferito il file `/etc/fstab` e aggiungiamo una linea si fatta:

```
<NOME PUNTO DI MONTAGGIO> <PERCORSO PUNTO DI MONTAGGIO> vboxsf defaults 0 0
```

Nel caso di SHARED e vboxshared:

```
SHARED /media/vboxshared vboxsf defaults 0 0
```

Poi salviamo e, assicurandoci che non sia ancora montata la cartella, diamo un

```
# mount -a
```

Se la cartella risulta montata in seguito al comando, allora possiamo stare certi che lo sarà ogni riavvio, altrimenti tentiamo di capire cosa c'è di sbagliato attraverso i messaggi di errore nella sintassi, oppure azzeriamo le operazioni, pena: **potremmo riuscire a non avviare più correttamente il nostro sistema operativo.**

### NOTA BENE:

la procedura sopra indicata per montare all'avvio una cartella virtualbox è simile per ogni dispositivo che volessimo montare ad avvio del nostro sistema, bisogna giusto apportare qualche modifica al tipo di partizione e sostituire al **nome del punto di montaggio** il file che identifica il nostro dispositivo, in genere si tratta del file `/dev/sdXY`, dove **X** è una lettera che identifica il dispositivo e **Y** un numero che identifica la partizione all'interno del dispositivo. (es. `/dev/sdc3`, cioè disco c partizione 3)

## 5.6 Il COMANDONE da due milioni di dollari

segue un elenco di comandi che include tutti i software che normalmente installo nel mio sistema ed eventuali configurazioni (*sezione in aggiornamento costante*), a voi il compito di informarvi su a cosa servono e perché dovrete o non dovrete installarli!

```
aurman -S clementine audacity visual-studio-code-bin mailspring
firefox firefox-i18n-it thunderbird thunderbird-i18n-it vlc
ponysay lolcat redshift jdk jdk-docs atom l3afpad deluge
terminator octave gimp xterm libreoffice-fresh libreoffice-fresh-
it texlive-bin texlive-core texlive-bibtexextra texlive-
fontsextra texlive-formatsextra texlive-games texlive-humanities
texlive-latexextra texlive-music texlive-pictures texlive-
pstricks texlive-publishers texlive-science texstudio wine wine-
mono winetricks wine_gecko playonlinux steam steam-native-
runtime ntfs-3g python-pip mariadb nitrogen xdotool gst-plugins-
good gst-plugins-bad gst-plugins-ugly gst-plugins-base gst-libav
gvfs alsa-firmware alsa-lib alsa-oss alsa-utils pulseaudio-alsa
pavucontrol rar unrar zip unzip p7zip sane hplip

sudo pip install youtube-dl
```

```
echo "export EDITOR=nano" >> .zshrc
```



## Capitolo 6

# Contatti e tanti saluti

Ringrazio chiunque diffonderà questa guida, chiunque mi aiuterà a correggerla e chiunque mi aiuterà ad ampliarla. Se avete qualche cosa da chiedermi relativa alla guida potete contattarmi per email: [psdady@msn.com](mailto:psdady@msn.com) o su Telegram al nickname [@PsykeDady](#) o sui gruppi linux-oriented:

[@gentedilinux](#)  
[@linuxandtech](#)

Sperando che il mio lavoro sia utile per voi, come lo sarà per me ogni qualvolta mi dimenticherò di installare qualcosa, mi congedo e vi auguro ancora una volta una buona permanenza nel mondo di Archlinux.

