



# Guida Installazione Arch

versione 1.2 : Talete

by PsykeDady

2019-08

## 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 [Manjaro](#), una distro su base arch completamente personalizzabile al momento dell'installazione, molto ma molto user-friendly. Segnalo anche [chakra](#).

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 Kde-Neon**.

Finalmente dopo un anno posso dire di aver concluso una prima *fase* di maturità di questa guida, che è stata letta, usata, commentata, criticata e corretta da tantissimi utenti. Ringrazio chiunque abbia partecipato a quella che oggi, ad Agosto 2019, ritengo essere la guida di arch versione 1.0

Detto questo: *buon divertimento e benvenuti nel fantastico mondo di Archlinux*.



*...Linux è sinonimo di libertà e rispetto...*

## Indice generale

1 Preparare il supporto di installazione.....	6
1.1 Metodo 1 da Linux: dd.....	6
1.2 Metodo 2 da Linux: copia su fat32 (Consigliato UEFI).....	7
1.3 Metodo 3 da Linux: varie GUI.....	9
1.4 Altri S.O.....	9
2 Hello world, I'm Archlinux Nice to meet you.....	10
2.1 preparazione del disco di installazione.....	11
2.2 Installazione dei pacchetti e connessione al mondo esterno.....	16
2.3 prime configurazioni.....	17
2.4 programma di installazione terminato.....	18
3 Configurazioni di sistema.....	19
3.1 Connettiamoci al mondo esterno e alcuni consigli iniziali.....	19
3.2 Server e driver.....	20
3.3 Dual boot con Windows.....	21
3.4 Aggiunta utente, creazione cartelle utente e cifratura home.....	22
3.5 configurare pacman e installare pakku.....	26
3.6 tmp file system.....	27
3.7 Swappiness.....	29
3.8 Swap file.....	29
3.9 Quando Tux congela: ibernazione.....	30
4 Desktop Environment, Display Manager NetworkManager e servizi systemd.....	33
4.1 Plasma D.E.....	35
4.2 Display Manager.....	37
4.3 NetworkManager e servizi di rete.....	38
4.4 netctl ed eduroam.....	39
4.5 Systemd: la bestia nera.....	41
5 Post-personalizzazioni di sistema by PsykeDady.....	43
5.1 I sette 'nano'.....	43
5.2 oh zsh, mio amore.....	45
5.3 fish, un'ulteriore avanzatissima shell.....	47
5.4 neofetch.....	47
5.5 Se avete installato su VirtualBox.....	50
5.6 Sotto effetto di LSD.....	52
5.7 IL COMANDONE.....	53
6 Contatti e tanti saluti.....	54
6.1 Template e legenda.....	54

pagina volutamente vuota

# Preparare il supporto di installazione

Prima di iniziare, se avete dubbi sulla simbologia del documento date un'occhiata all'ultima sezione di questa guida, Template e legenda, che vi chiarirà cosa si intende con un determinato colore-simbolo o blocco di ciò che leggerete. Buon inizio

## 1.1 Metodo 1 da Linux: dd

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

```
sudo 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 poterla nuovamente riutilizzare, dovrete azzerarla con lo stesso tool:

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

### **ATTENZIONE:**

**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 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 pennetta in fat32, supposto sia /dev/sdX il percorso della pennina, seguire queste istruzioni per formattarne il contenuto con fdisk (effettuare le operazioni con su o con sudo).

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

#si entrerà in modalità fdisk, scrivere i prossimi comandi e premere invio
fdisk /dev/sdX
o
p
1
2048
#(invio senza scrivere nulla oppure scegliere una dimensione)
t
5
6
b
w #si uscirà dalla modalità fdisk

mkfs.fat /devX1
mount /dev /sdX1 usb
#sostituendo ad AAAA e MM le stesse date che trovate sulla iso
dosfslabel /dev /sdX1 ARCH_AAAAMM
```

## Preparare il supporto di installazione

---

### SUGGERIMENTO:

È possibile, alternativamente a `fdisk`, usare `cfdisk` che è più 'user-friendly'

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 buffer del disco e della pennetta e smontare le cartelle, per evitare che le modifiche non vengano effettuate 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 del software citato e di **parted** ed eseguire i prossimi comandi (prima di smontare la `usb`):

```
mkdir usb/boot/syslinux  
extlinux --install usb/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 l'ho voluto riportare. Se qualcosa non dovesse funzionare vi invito a documentarvene personalmente sulla guida ufficiale

## 1.3 Metodo 3 da Linux: varie GUI

Se non amate molto sporcarvi 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 mi hanno consigliato i seguenti programmi:

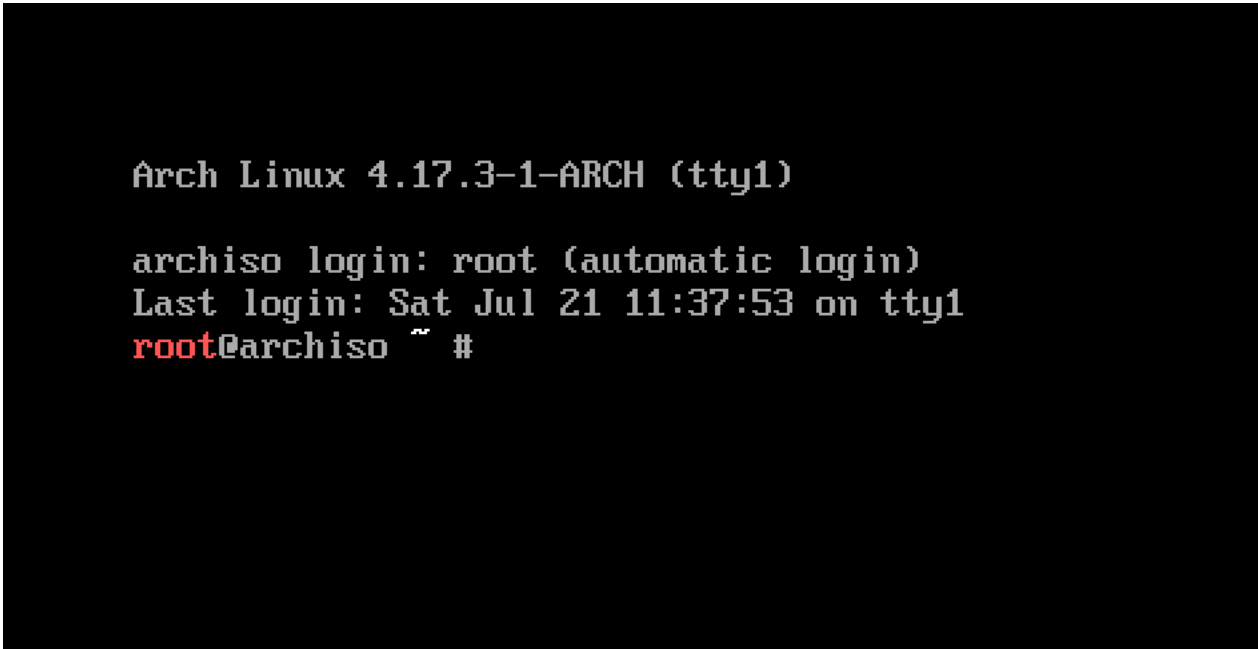
- Win32DiskImager
- USBWriter



# 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, possiamo passare alle prime fasi dell'installazione.

Avviando Arch, se tutto va bene, dovrete ritrovarvi davanti ad una schermata simile:



```
Arch Linux 4.17.3-1-ARCH (tty1)

archiso login: root (automatic login)
Last login: Sat Jul 21 11:37:53 on tty1
root@archiso ~ #
```

*Figura 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 massima leggibilità

## 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` sopriamo quindi le coordinate de lnostro 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 da ti 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 tenere.

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 di avere un sistema EFI, con tutto ciò 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 riservarne 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.

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

```
gdisk /dev/sda
```

in questo modo si entrerà nella 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 `cdisk`, e avere a che fare con partizioni di tipo tradizionale, cioè **MBR**.

Esiste anche `cgdisk`, l'alternativa user-friendly di `gdisk`, prendetela in considerazione se non volete seguire alla cieca le istruzioni qui sotto ma organizzare in maniera più pratica le partizioni secondo un' organizzazione personale.

Dunque procediamo con la supposizione di cui sopra:

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

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

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

n
4
(invio senza scrivere niente)
#sostituendo a ZZ il numero di Giga da dare alla swap
+ZZG
8200

w
```

Hello world, I'm Archlinux Nice to meet you

---

dopo essere usciti dalla modalità `gdisk`, possiamo accertarci della situazione usando il comando `gdisk -l` oppure con `blkid`.

Se non avete un sistema **UEFI** e volete usare `fdisk` la lista di parametri da inserire potrebbe essere questa:

```
n
p
1
(invio senza scrivere niente)
# sostituire a XXX il numero di giga da dare alla root
+XXXG

n
p
2
(invio senza scrivere niente)
# sostituire a YYY il numero di giga da dare alla home, qui mettere la
maggiorparte dello spazio
+YYYG

n
p
3
(invio senza scrivere niente)
# sostituire a ZZ il numero di Giga da dare alla swap
+ZZG

t
3
82
```

Ricordate comunque di controllare il significato delle opzioni tramite gli appositi help interattivi. Per usare le partizioni è comunque necessario formattarle, ritorniamo nella supposizione in cui siate UEFI:

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

Altrimenti avrete una cosa simile:

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

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
```

oppure (Non UEFI)

```
mount /dev/sda1 /mnt  
mkdir /mnt/home  
mount /dev/sda2 /mnt/home  
swapon /dev/sda3
```

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 sono stati trasmessi e ricevuti 3 pacchetti. Ora che siamo sicuri possiamo andare avanti.

Arch offre un modo davvero comodo per scaricare i pacchetti d'avvio sulla nostra nuova installazione, attraverso **pacstrap**:

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

NOTA BENE:

efibootmgr va installato solo se si ha sistema UEFI

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 con `passwd` e modifichiamo il file `fstab` (con `nano`, `vi` o qualunque altro editor ci piace usare) creato in precedenza sostituendo nella riga della swap il parametro `none` con la dicitura **swap**.

Supponendo che stiate usando `nano`, potete digitare:

```
nano /etc/fstab
```

fare la modifica al file e poi **ctrl-x** per uscire (**y** e poi **enter** per salvare il file).

Configuriamo quindi il grub:

```
grub-install /dev/sda
```

```
grub-mkconfig -o /boot/grub/grub.cfg
```

si può quindi dare un nome alla macchina in rete:

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

Si può ora impostare la lingua: andando ad editare il file `/etc/locale.gen` decommentando (togliendo il carattere `#`) le tre linee che iniziano con **it\_IT**, poi diamo il comando `locale-gen` per generare i file della lingua.

In seguito bisogna generare un buon `/etc/locale.conf`, usiamo il nostro editor preferito e scriviamo all'interno:

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



Hello world, I'm Archlinux Nice to meet you

---

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 se volete riavviare
```

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

# C

## onfigurazioni di sistema

Dopo aver riavviato e tolto la chiavetta dovremmo essere davanti la schermata di selezione dei sistemi del grub, dove la prima opzione dovrebbe proprio corrispondere a quella di avviare archlinux. Se così non fosse, reinserite la chiavetta, rimontate le partizioni e rifate il chroot, cercando la soluzione al problema usando anche la wiki. Da qui in poi sarà supposto che voi abbiate il sistema funzionante e possiate fare il login tramite account di root.

E quindi inserite come nome utente proprio *root* e come password quella impostata precedentemente. 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, esattamente come prima possiamo usare [dhcpcd](#) e nel caso della rete senza fili [wifi-menu](#).

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

```
pacman -Syuu
```

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:

- **linux-headers** sono una serie di interfacce che servono a compilare alcuni pacchetti, capirete a cosa vi serve quando arriveremo ai repository AUR
- **os-prober** serve a rilevare altri sistemi operativi all'avvio, digitate quindi [grub-mkconfig -o /boot/grub/grub.cfg](#) per aggiornare il grub
- **git** è un sistema di versioning di file e cartelle, usatissimo in ambito di programmazione e vi servirà per scaricare il codice da repository remoto

- **bash-completion** serve ad abilitare l'autocompletamento con tab nella famosissima shell bash

### 3.2 Server e driver

In ambienti linux, senza motore grafico, si può usare il pc al più come server. Oggi giorno la produttività dipende strettamente da ciò che si vede e come ci si può interagire. Per questo motivo verrà illustrato come installare il server grafico **XORG**. Esistono oggi alternative valide, ma rimane questo tuttavia il server più stabile e consolidato nel mondo del pinguino. È quindi consigliabile comunque installarlo a prescindere da ciò che vorrete provare in futuro.

Insieme al server grafico si consiglia di installare il suo sistemadi init, che nel caso in cui un D.M. (la grafica di accesso al sistema) non funzioni a dovere sarà un ottimo sostituto. Quindi:

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

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

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

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

altrimenti andiamo ad identificare la 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 installare i driver ati
pacman -S xf86-video-ati

#per installare i driver 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 Dual boot con Windows

Se avete il dual boot con Windows è probabile ci sarà tra un sistema e l'altro uno scarto di tempo di due ore, questo perchè Windows non usa il tempo universale ma il locale.

Se non volete quindi modificare il comportamento di windows ( la wiki di arch spiega molto bene come fare nel caso) potete modificare invece il comportamento di Archlinux

```
timedatectl set-local-rtc 1 --adjust-system-clock
```

NOTA BENE:

La saggia wiki in realtà consiglia di cambiare il comportamento di windows, e non di archlinux.

### 3.4 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 aggiungere 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 utente non amministratore basta eliminare il gruppo wheel dal comando 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) che il nostro utente fa parte del gruppo amministratore, questo tramite visudo:

```
#impostiamo prima un editor di testo a noi più amichevole, di default è 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 caso decommentare:

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

A questo punto configuriamo le cartelle utente, installiamo

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

Al nostro primo accesso con l'utente ricordiamoci di dare :

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

Se vogliamo possiamo accedere subito scrivendo `su <nome utente>`

#### NOTA BENE:

Ricordati che puoi cambiare il nome delle cartelle di base (Immagini, Video ...etc) modificando il corrispettivo in `/home/<nome utente>/.config/user-dirs.dirs` inserendo ad uno ad uno i nomi che desideriamo sostituire.  
I nomi predefiniti dovrebbero essere quelli della lingua impostata

#### ATTENZIONE:

**Il prossimo step è opzionale, fatto in fretta o male potrebbe farvi perdere dati, potreste dover riprendere la live per aggiustare le cose o reinstallare se non siete ingrado. Quindi procedete con cautela e se lo ritenete un passo fondamentale. Inoltre vorrei portarvi a conoscenza del fatto che questo metodo renderà la vostra home incompatibile con il client dropbox se lo usate, quindi dovrete installare dropbox in una cartella diversa da quella proposta.**

Potete decidere di cifrare il contenuto della vostra home, un po' come succede nei telefoni che possiedono metodi di sblocco biometrici. Per farlo la prima cosa è installare alcuni pacchetti:

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

Quindi caricate l'apposito modulo del kernel

```
modprobe ecryptfs
```

Potrebbe essere necessario apportare una modifica al file `/etc/mkinitcpio.conf` e scrivere all'interno della sezione **MODULES** il nome del modulo per forzarne il caricamento ad ogni avvio del pc. Successivamente ricompilate 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 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 criptare
```

Potete usare i due comandi ogni volta che volete cifrare o decifrare la cartella home manualmente, può accadere alla volta 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 fare l'accesso a mano ogni volta) impostare l'automounting della home già decriptata all'accesso. Facciamo un backup del file `/etc/pam.d/system-auth`

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

ed editiamo quindi il file *system-auth* con il nostro editor preferito

```
<editor> /etc/pam.d/system-auth
```

Da adesso facciamo MOLTA attenzione, sbagliando qualunque cosa potremmo non essere più in grado di accedere a nessun account ( dovremmo quindi aggiustare le cose in chroot dalla iso di arch). Andiamo quindi 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` ) e correggete l'errore nel file o, nel caso non riusciate, eliminate le modifiche e riprendete il file di backup

```
mv -f /etc/pam.d/system-auth.old /etc/pam.d/system-auth
```

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 un tty diverso e ricriptatela da virtual console con i comandi di umount, ripassate quindi al tty principale per continuare l'accesso.



## 3.5 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 gestore! 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 e non con root.

```
git clone https://aur.archlinux.org/pakku.git
cd pakku
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 di Archlinux, ci trovate davvero di tutto dentro (*perciò fate comunque attenzione*).

### NOTA BENE:

pakku, così come la maggiorparte degli AUR-helper, non vanno usati come utenti privilegiati (con **sudo** o **l'account root** per intenderci)

Tramite pakku potete visualizzare la differenza tra versione presente e quella aggiornata del software, oppure visionare e modificare il **PKGBUILD** (una sorta di file che contiene le istruzioni di compilazione del pacchetto, con dipendenze, siti da dove scaricare i binari ed eventualmente potete anche selezionare delle opzioni), questo lo rende uno strumento molto potente!

Pakku è scritto in **NIM**, *un linguaggio di programmazione ad alto livello multi piattaforma*, con aspetti molto molto interessanti, vi consiglio di approfondirlo se siete del settore.

### 3.6 tmp file system

Nel nostro file system una cartella speciale, */tmp*, 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 svuotata ogni qual volta il calcolatore viene spento.

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

#### **SUGGERIMENTO:**

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 particolarmente questo procedimento ai pc con meno di 8GB di RAM che installeranno quindi molti pacchetti da AUR

Per disabilitare il montaggio automatico basta dare:

```
sudo 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, va quindi eliminato manualmente.

Si può tornare al comportamento di default specificando manualmente il montaggio in `/etc/fstab`, aggiungendo questa riga 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 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 la SWAP.

In ogni caso questo aspetto del sistema si può benissimo controllare attraverso la swappiness, impostandola in modo permanente o temporaneo.

Per farlo in maniera definitiva possiamo modificare il file `/etc/sysctl.d/99-sysctl.conf`

```
vm.swappiness=1
```

Come funziona questo valore però? Perché **1** e non **0**?

Il valore di swappiness va da 0 a 100 e **indica il valore di ram in percentuale che si desidera tenere libero prima di attivare la zona di swap**. Ad esempio normalmente il valore di swappiness è *60*, questo significa che oltre il *40% di ram occupata*, la 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](#).

La fase iniziale di configurazione è finita, consiglio prima di proseguire un riavvio del sistema.

## 3.8 Swap file

Potrebbe essere più comodo in alcuni casi avere un file di swap anziché la partizione, un po' come fanno windows e mac.

In tal caso dovrete prima creare il file in questione:

```
sudo dd if=/dev/zero of=/swapfile bs=1M count=XXX status=progress
```

al posto di XXX inserite il quantitativo di megabyte che volete impostare, ad esempio per una ram di 2G potete impostare `count=2000`.

Impostiamo i permessi giusti al file

```
sudo chmod 600 /swapfile
```

e vediamo se va tutto bene attivandola

```
sudo swapon /swapfile
```

Se avete messo la partizione di swap potete disattivarla così

```
sudo swapoff /dev/sdXY # sostituendo le coordinate
```

Se tutto è ok, possiamo anche pensare di abilitare il file di swap all'avvio insieme al montaggio dei dischi.

Apriamo con un editor di testo il seguente file:

```
sudo nano /etc/fstab
```

e scriviamo una riga così:

```
/swapfile swap swap defaults 0 0
```

ora possiamo testare se abbiamo scritto bene il file. Prima di tutto disattiviamo la swap con `swapoff` se è attiva.

Poi digitiamo:

```
sudo swapon -a
```

Questo comando abiliterà tutte le swap presenti in `/etc/fstab`. Sì, se vi state chiedendo perché io abbia detto “*le swap*”, la risposta è che potete tenere anche più file di swap o più partizioni.

## 3.9 Quando Tux congela: ibernazione

Prima di tutto bisogna capire cos'è l'ibernazione, magari molti non lo sanno. Quando un sistema operativo entra in questo stato, la memoria con tanto di processi attivi vengono salvati nell'area di swap e il pc si spegne. Al riavvio, sarà come se avessimo sospeso il pc, ma non è stata consumata batteria (potete anche staccare l'alimentazione).

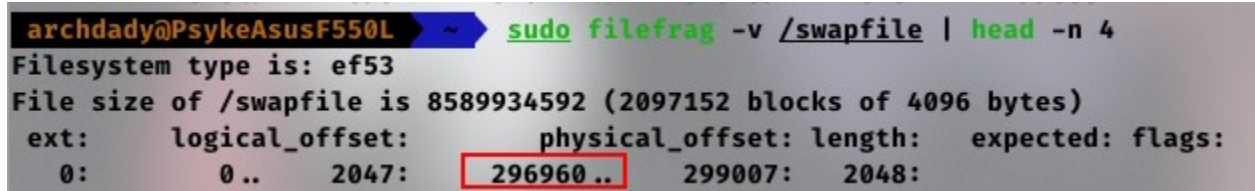
I prerequisiti per un'ibernazione corretta sono quindi legati alla swap: dovete averla attiva e funzionante, la parte di swap libera deve quantomeno eguagliare la memoria RAM utilizzata, in modo da poter ibernare senza complicazioni. Consiglio quindi di utilizzare una swappiness molto bassa, così da avere la swap tendente al vuoto.

Avendo questi prerequisiti, come si fa ad ibernare su Arch? I passi da effettuare sono principalmente questi:

Segnamoci innanzitutto l'UUID della partizione tramite il comando `blkid`. Se usiamo un file di ibernazione, segnamoci l'UUID del disco che contiene tale file, e teniamoci da parte anche il suo offset sul tale disco così:

```
sudo filefrag -v /swap | head -n 4
```

e andiamo a prendere il valore sotto il **physical\_offset**, prima dei tre puntini.



```
archdady@PsykeAsusF550L ~$ sudo filefrag -v /swapfile | head -n 4
Filesystem type is: ef53
File size of /swapfile is 8589934592 (2097152 blocks of 4096 bytes)
ext:      logical_offset:      physical_offset: length:  expected: flags:
0:        0 ..      2047:      296960 ..      299007:  2048:
```

Figura 2: se avete scelto il file di ibernazione, questo è il valore da prendere

Ora dobbiamo modificare il file di configurazione del grub:

```
sudo nano /etc/default/grub
```

Andiamo sulla riga `GRUB_CMD_LINUX_DEFAULT=" ... "` e aggiungiamo nelle virgolette questa stringa:

```
resume=UUID=<mettiamo qui l'UUID segnato prima>
```

Se siamo abbiamo scelto il file, aggiungiamo questa ulteriore parte:

```
resume_offset=<valore offset segnato prima>
```

Ancora, se usiamo l'ibernazione su file, potrebbe essere necessario dare `lsblk` e segnarsi i numero MAJ:MIN della partizione dove tenete il file. Una volta fatto date questi due comandi

```
echo <MAJ>:<MIN> /sys/power/resume
```

```
echo <valore offset> /sys/power/resume_offset
```

Andiamo ora a modificare l'avvio dei moduli di sistema:

```
sudo nano /etc/mkinitcpio.conf
```

e nella riga `HOOKS=(base udev...fsck)` andiamo ad aggiungere dopo filesystems il parametro `resume` (separato da spazi)

Ora aggiorniamo grub e sistemi di avvio:

```
sudo grub-mkconfig -o /boot/etc/grub
```

```
sudo mkinitcpio -p linux
```

Riavviamo e, teoricamente, l'ibernazione è pronta. Per testarla consiglio, una volta sistemato tutto l'ambiente, di aprire un paio di finestre a sistema appena avviato e ibernare. Se il vostro DE non ha un'opzione per farlo potete usare systemd da terminale

```
systemctl hibernate
```

Se al riavvio si riapre tutto come avete lasciato, l'esperimento ha avuto successo. Altrimenti consultate la wiki per capire cosa potrebbe essere andato storto.

# Desktop Environment, Display Manager NetworkManager e servizi systemd

Premessa, diceva il mio maestro di basso: ad ognuno la sua pistola. Su linux chiunque ha piena libertà di scegliere l'ambiente grafico (o di non sceglierlo proprio) che più gli garba, sia per leggerezza, per bellezza, per complessità d'uso o per la sua malleabilità. Ce n'è per tutti i gusti! La wiki in questo può essere molto più completa di qualunque cosa io possa scrivere qui sotto, vi elencherò in un primo momento i DE più conosciuti e le loro peculiarità, ma la mia guida tratterà solo come installare quello che preferisco: **plasma**.

Per sistemi poco prestanti consiglio **XFCE**, **LXDE** o (per chi ha la pazienza di configurarlo) **IBWM**, queste sono tre validissime alternative che con qualche tocco di eleganza hanno fatto la storia delle configurazioni e dei temi più belli e apprezzati del mondo Linux.

Per chi ha uno schermo HIDPI consiglio **plasma**, **Cinnamon**, **Mate**, **DDE** o **GNOME**. Andiamo con ordine:

- reputo il primo l'ambiente più avanzato dal punto di vista delle animazioni, delle personalizzazioni e dei servizi automatici, inoltre nel suo approccio minimale occupa meno delle sue alternative (intorno ai 300Mb per esperienza personale).
- Il secondo è un ambiente molto conosciuto dagli utenti di LinuxMint, infatti è la scelta predefinita ed è sviluppata e mantenuta dal team che sostiene la distribuzione. Considero cinnamon l'alternativa più avanzata ma anche più scattosa e tra le più pesanti.
- Mate invece è una buona via di mezzo, non molto avanzato con i tempi, da poco ha introdotto il supporto all'hidpi e comunque ho riscontrato qualche problemino nell'usarlo, ma è leggero ed è altamente modulare, permette senza problemi di usare componenti di altri **WM** o **DE**.
- Deepin Desktop Environment deriva dalla famosa deepin linux, qualche anno fa è stato il mio Desktop preferito in assoluto, molto elegante, effetti nella media e pochi problemi. Da qualche tempo perde colpi, hanno abbandonato lo sviluppo del wm appoggiandosi a kwin (quello di plasma) e in questa fase di transizione non nascondo che ci sono parecchi problemi fastidiosi.



- Gnome è un ambiente conosciuto per la sua pesantezza in termini sia di memoria occupata che di animazioni, nella versione di Fedora Linux è straordinariamente leggera e funzionante, ma si trasforma totalmente su altre distribuzioni. Resta uno degli ambienti più apprezzati per via delle estensioni (che comunque lo appesantiscono) e della “omogenità” dei suoi temi.

Ci sarebbe infine **Enlightment**, un DE con tiling manager che ho trovato difettoso in alcune applicazioni (come il noto I.M. **Telegram**). È comunque un'alternativa che cito poiché gestisce nativamente l'HIDPI.

### NOTA BENE:

D'ora in poi potrete operare tranquillamente con l'account root così come quello utente (consigliato). Quando agirete come root non sarà necessario specificare **sudo** all'inizio dei comandi.

Durante però l'esecuzione di **pkcu** ricordo che il root potrebbe dare problemi.

## 4.1 Plasma D.E.

In realtà installare un DE è spesso tempo di uno o due comandi, il più è quello di configurarlo a proprio piacere.

Potete installare plasma nella sua versione completa, con tanto di intero parco applicazioni kde così:

```
sudo pacman -S plasma kde-applications
```

Quando digiterete questo comando, vi verrà chiesto di selezionare quali software del gruppo volete. Per avere tutto basta premere invio, per chi conosce già i software e li riconosce dal nome basterà inserire i numeri corrispondenti divisi da spazio per avere solo quei software

```

:: There are 161 members in group kde-applications:
:: Repository extra
 1) akonadi-calendar-tools 2) akonadi-import-wizard 3) akonadiconsole 4) akregator 5) ark 6) artikulate 7) audiocd-kio 8) blinken 9) bomber 10) bovo
11) cantor 12) cervisia 13) dolphin 14) dolphin-plugins 15) dragon 16) ffmpegthumbs 17) filelight 18) granatier 19) grantlee-editor 20) gwenview
21) juk 22) k3b 23) kaccounts-integration 24) kaccounts-providers 25) kaddressbook 26) kajongg 27) kalarm 28) kalgebra 29) kalzium 30) kamera
31) kamoso 32) kanagram 33) kapman 34) kapptemplate 35) kate 36) katomic 37) kbackup 38) kblackbox 39) kblocks 40) kbounce 41) kbreakout
42) kbruch 43) kcachegrind 44) kcalc 45) kcharsselect 46) kcolorchooser 47) kcron 48) kde-dev-scripts 49) kde-dev-utils 50) kdebugsettings
51) kdeggraphics-mobipocket 52) kdeggraphics-thumbnails 53) kdenetwork-filesharing 54) kdenlive 55) kdepim-addons 56) kdesdk-kioslaves
57) kdesdk-thumbnails 58) kdf 59) kdialog 60) kdiamond 61) keditbookmarks 62) kfind 63) kfloppy 64) kfourinline 65) kgeography 66) kget
67) kgoldrunner 68) kgspace 69) khangman 70) khelplcenter 71) kig 72) kigo 73) killbots 74) kimagemapeditor 75) kio-extras 76) kirigami-gallery
77) kiriki 78) kiten 79) kjumpingcube 80) kleopatra 81) klettres 82) klickety 83) klines 84) kmag 85) kmahjongg 86) kmail 87) kmail-account-wizard
88) kmimes 89) kmix 90) kmousetool 91) kmouth 92) kmp3 93) knavalbattle 94) knetwalk 95) knights 96) knotes 97) kolf 98) kollision
99) kolourpaint 100) kompare 101) konqueror 102) konquest 103) konsola 104) kontakt 105) kopete 106) korganizer 107) kpatience 108) krdc
109) kreversi 110) krfb 111) kross-interpreters 112) kruler 113) kshisen 114) ksirk 115) ksnakeduel 116) kspaceduel 117) ksquares 118) ksudoku
119) ksystemlog 120) kteatime 121) ktimer 122) ktouch 123) ktuberling 124) kturtle 125) kubrick 126) kwalletmanager 127) kwave 128) kwordquiz
129) kwrite 130) lokalize 131) lskat 132) marble 133) mbox-importer 134) minuet 135) okular 136) palapeli 137) parley 138) picmi
139) pin-data-exporter 140) pin-sieve-editor 141) poxml 142) print-manager 143) rocs 144) signon-kwallet-extension 145) spectacle 146) step
147) sweeper 148) telepathy-kde-accounts-kcm 149) telepathy-kde-approver 150) telepathy-kde-auth-handler 151) telepathy-kde-call-ui
152) telepathy-kde-common-internals 153) telepathy-kde-contact-list 154) telepathy-kde-contact-runner 155) telepathy-kde-desktop-applets
156) telepathy-kde-filetransfer-handler 157) telepathy-kde-integration-module 158) telepathy-kde-send-file 159) telepathy-kde-text-ui 160) umbrello
161) zeroconf-ioslave

Digita una selezione (default=tutto): 5 13 14 17 20 22 27

```

*Figura 3: Questa selezione permetterà di installare solo alcuni software, come dolphin (13) o ark (5)*

Alternativamente potete avere un installazione minimale così:

```
sudo pacman -S plasma-desktop
```

Questo installerà solo l'ambiente, poi voi dovrete occuparvi di installare uno ad uno i software che userete (file manager, browser, pdf reader... etc)

Consiglio fortemente, se avete smartphone android, di installare kde-connect e associarlo all'omonima applicazione disponibile sul play store:

```
sudo pacman -S kdeconnect
```

Questo fantastico software vi permetterà di controllare alcune funzioni del pc a distanza, di controllare notifiche e messaggistica da pc e molto altro in realtà.

Consiglio anche di installare l'integrazione di plasma con i browser, consente di essere notificati quando finisce un download, integrare il pannello di sistema con l'avanzamento di download e musica da internet, un *must-have* insomma!

Se come me poi siete amanti delle dock, potete installare e personalizzare latte-dock, appositamente sviluppata per plasma:

```
sudo pacman -S latte-dock
```

### NOTA BENE:

Su molti Desktop Environment (plasma incluso) la tastiera è impostata di default con layout USA. Non è un problema di lingua, va proprio risolto dalle impostazioni della tastiera, togliendo la spunta al layout predefinito e selezionando manualmente la tastiera italiana

Se avete problemi con plasma, ma non volete riavviare il pc, potete tentare il riavvio del D.E. e del suo W.M.

Premete quindi `alt-f2` e digitare quanto segue:

```
killall plasma && kstart plasmashell && kwin_11 --replace
```

e premete quindi invio.

## 4.2 Display Manager

Insieme a plasma, nella sua versione completa, viene installato il suo Display Manager, o D.M., **SDDM**. Questo genere di software è banalmente quello che vi si presenta alla schermata di avvio chiedendovi la password per farvi accedere all'ambiente desktop avviando per altro il server X (o qualunque server grafico voi usiate). In genere è possibile selezionare anche un utente e un desktop diverso nel caso in cui ne abbiate più di uno installato.

Per utilizzare un DM dovreste abilitarlo come servizio di systemd, nel caso di sddm ad esempio avremo:

```
sudo systemctl enable sddm
```

al riavvio potrete godere della vostra interfaccia d'accesso.

Normalmente, se l'avete installato, anche xinit può avviare il vostro ambiente grafico, questa soluzione potrebbe accontentare quell'utenza che vuole fare del proprio portatile una vera freccia ad avviarsi. Per usufruire di xinit dobbiamo prima di tutto scrivere il comando di avvio del nostro DE all'interno del file `~/.xinitrc`, il file si trova nella home e varierà quindi per ogni utente.

Abbiamo dunque:

- per kde :

```
exec startplasma-x11
```

- per GNOME con X Server:

```
export GDK_BACKEND=x11
```

```
exec gnome-session
```

- per xfce

```
exec startxfce4
```

- per mate

```
exec mate-session
```

- per dde

```
exec startdde
```

- per cinnamon

```
exec cinnamon-session
```

Per altri DE consultare le relative wiki!

## 4.3 NetworkManager e servizi di rete

Dietro le quinte, quando nella guida si è fatto uso di [wifi-menu](#), il servizio che 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 notifichino ad esempio quando cade la connessione, vi mostrino la potenza di segnale continuamente e facilitino l'inserimento di ogni tipo di credenziali, proteggendone anche il contenuto se siete in pubblico.

Il servizio di connessione per eccellenza sui sistemi linux è **NetworkManager**.

### NOTA BENE:

Usando NetworkManager disabiliti netctl. Tentando di usare netctl quando nm è attivo causerai errori. Utilizza systemd per gestire il passaggio da un servizio all'altro se dovessi preferirne uno ad un altro in base alle situazioni

Quindi per installare networkmanager:

```
sudo pacman -S networkmanager
```

Se eventualmente vi servono particolari tipi di connessione, come vpn o point-to-point protocol, potrebbero servirvi pacchetti aggiuntivi:

```
sudo pacman -S networkmanager-pptp networkmanager-vpnc
```

Alcuni DE si portano nell'installazione completa un applet che interagisce con NetworkManager, tuttavia se non fosse così la wiki spiega quali pacchetti installare per connettersi comodamente con un click.

Nel più comune dei casi comunque viene usato quello di gnome:

```
sudo pacman -S network-manager-applet
```

è necessario abilitare NetworkManager con **systemd**, altrimenti sarà necessario ogni riavvio richiamare manualmente il servizio. Quindi

```
sudo systemctl enable NetworkManager
```

## 4.4 netctl ed eduroam

Spesso si è demoralizzati dall'usare **netctl** poichè file di configurazioni più complessi vanno scritti a mano.

Per mie esigenze personali ho dovuto cercare un modo di connettermi alla linea universitaria **eduroam**, ho quindi pensato di condividere la procedura che mi ha permesso di collegarmi.

Innanzitutto tenete una connessione con [wifi-menu](#), inserendo anche una password a caso, non importa. Quando vi dirà che la connessione è fallita e vi chiederà se volete comunque tenere il file di configurazione rispondete affermativamente.

Andate quindi a modificare manualmente il file generato. Si supporrà adesso che la scheda di rete si chiami per il sistema [wlp3s0](#), e il nome della linea semplicemente [eduroam](#) come nelle specifiche internazionali. Il nome del file dovrebbe essere quindi <nome interfaccia>-<nome rete> e essere nella cartella [/etc/netctl/](#)

```
sudo nano /etc/netctl/wlp3s0-eduroam
```

NOTA BENE:

per scoprire il nome dell'interfaccia potete usare **ip link** oppure **ifconfig**

Quindi scrivete all'interno del file quanto segue:

```
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 ormai 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 non è difficile da comprendere, quindi posso anche supporre che siate in grado di poterla modificare e metterci mano lì dove cambi qualcosa.

## 4.5 Systemd: la bestia nera

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

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



	comando	spiegazione
sudo	<b>systemctl enable &lt;servizio&gt;</b>	abilita il servizio all'avvio, che viene quindi attivato ogni qualvolta accedete
	<b>systemctl start &lt;servizio&gt;</b>	avvia immediatamente il servizio
	<b>systemctl restart &lt;servizio&gt;</b>	spegne e riavvia il servizio
	<b>systemctl stop &lt;servizio&gt;</b>	spegne il servizio, contrario di <i>start</i>
	<b>systemctl disable &lt;servizio&gt;</b>	disabilita il servizio, contrario di <i>enable</i>
sto fresco	systemctl status <servizio>	controlla lo stato del servizio, se è attivo, in errore o spento
	systemctl poweroff	spegne il sistema
	systemctl reboot	riavvia il sistema
	systemctl hibernate	iberna il sistema, da usare solo se avete attivato l'ibernazione in modo corretto
	systemctl suspend	sospende il sistema
	systemctl suspend-then-hibernate	sospende per un certo periodo di tempo. Poi iberna
	systemctl hybrid-sleep	Sospende e iberna il sistema. Così che se la batteria si scarica, il pc è comunque ibernato

Prendiamo ad esempio che vogliate usare nuovamente *netctl* anziché *NetworkManager*, la procedura corretta sarebbe:

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

Al contrario invece

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

#### NOTA BENE:

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**.

# Post-personalizzazioni di sistema by PsykeDady

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

## 5.1 I sette 'nano'

da linea di comando, l'avrete capito, 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, sviluppare e scrivere documenti complessi come potrebbe essere *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.

Creiamo con il nostro 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, senza però generare un nuovo fine linea nel testo, così non vi dovrete spostare per vedere cosa contengono quelle lunghe infinite linee che scrivete

**autoindent**: se programmate con nano, questo può essere davvero un aiuto importante. Abilitare questa opzione vi permette, una volta indentata una linea di 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 (in questo caso vi perdono, ma rimediate)

**linenumbers**: una volta che avete abilitato softwrap non capirete quando inizia una riga e quando invece sta continuando quella precedente. Questa opzione mostra i numeri riga, in termini di numeri cardinali, ogni suo inizio

**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 sistemi diversi da arch potrebbe non essere lo stesso path.

## 5.2 oh zsh, mio amore

*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 una priorità.

Allora ecco che entrano in gioco le *'alternative'* alla shell per eccellenza, sua maestà [bash](#).

La proposta più popolare è sicuramente [zsh](#). In realtà l'avete già usata e non lo sapete (?), infatti la live di archlinux ne è equipaggiata.

Quindi installiamo [zsh](#) e anche il suo famoso gestore di plugin: **oh-my-zsh**. Qui entra in gioco il famoso AUR con l'aur helper scelto, quindi non eseguite il prossimo comando da un account root.

```
pakku -S zsh oh-my-zsh zsh-theme-powerlevel9k
```

Il terzo pacchetto è uno dei temi più famosi di zsh, la configurazione che vi propongo è proprio quella che si basa su questo pacchetto.

In realtà solo oh-my-zsh si trova su AUR, gli altri si dovrebbero trovare sui repository standard, ma [pakku](#) come già detto, installerà anche i pacchetti normali (è come fosse un estensione di [pacman](#))

Copiamoci nella home innanzitutto il file rc di oh-my-zsh:

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

Modifichiamo quindi il file appena creato 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:

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

Potete comunque anche fare un link alla cartella con [ln -sf](#).

Adesso se abbiamo deciso che il nostro sistema ha la localizzazione italiana, dobbiamo leggermente modificare il tema. Allo scopo consiglio di usare un editor che permetta di modificare più parole alla volta tramite *cerca e sostituisci*. Supponendo di avervi fatto innamorare di [nano](#) potete usare **ctrl-w** per cercare una stringa (ripremetelo ogni volta che avrete fatto una modifica e cercate l'occorenza successiva).

Apriamo con i permessi di amministrazione il file `/usr/share/oh-my-zsh/temes/powerlevel9k/function/icon.zsh` ed eliminiamo o commentiamo tutte le righe che iniziano con **local LC\_**.

Il nostro `zsh` dovrebbe essere pronto, possiamo testarlo digitando `zsh` ed eventualmente impostarlo come shell predefinita tramite `chsh`:

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

### NOTA BENE:

La **powerline** (ovvero la barra che utilizza il tema **powerlevel**) potrebbe necessitare di un font in grado di visualizzare alcuni caratteri particolari. Consiglio di installare il pacchetto font: **ttf-fira-code**

## 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

```
sudo pacman -S fish
```

Fish è un po' particolare da tutti i punti di vista, ad esempio il suo file `rc` lo trovate in `~/.config/config.fish`.

Per personalizzare la shell dovrete usare `fish_config`, che aprirà un server web sul vostro browser dove potrete scegliere facilmente 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 nell'`rc` queste righe:

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

Per usarlo come shell predefinita si può digitare:

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

## 5.4 neofetch

Agli utenti linux piace avere tutto sotto controllo, e soprattutto piace che questo continuo monitorare sia esteticamente appagante.

Ecco perchè, chi per noi, ha creato dei tool che mostrano in modo sgargiante quello che tutte le informazioni di cui si ha bisogno. Uno di questi è `neofetch`, che possiamo semplicemente installare con

```
sudo pacman -S neofetch
```

Possiamo abilitare varie funzionalità installando i pacchetti aggiunti:

```
pacman -S catimg feh imagemagick jp2a libcacat nitrogen pacman-contrib w3m
```

Avviamolo almeno una volta scrivendo `neofetch` in modo da fargli creare i file di configurazione e poi andiamo a modificarli

```
nano ~/.config/neofetch/config.conf
```

Ecco la mia semplicissima configurazione della sezione **info**, che è quella poi che determina cosa uscirà stampato sul terminale:

```
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
}
```

Se studiate bene la wiki e le documentazioni sul web potrete sfruttare al meglio il vostro neofetch.

Poniamo di voler far partire neofetch con l'apertura del terminale (altra cosa che piace molto in genere) potete inserire `neofetch` nel file rc del vostro interprete preferito ( `~/.bashrc` per **bash**, `~/.zshrc` per **zsh**, etc...)



## 5.5 Se avete installato su VirtualBox

Se avete installato il tutto su virtual box, dopo aver installato le guest addiction (sezione Server e driver della guida) e il pacchetto estensione sul sistema operativo ospitante, potreste volere condividere delle cartelle tra sistema host e guest.

Supponiamo che la cartella si chiami Shared e la vogliamo montare nel percorso /media/vboxshare, le operazioni da eseguire saranno queste:

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

#quindi montiamola in Shared
sudo mount -t vboxsf Shared /media/vboxshare
```

È in genere 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: modifichiamo quindi il file in questione

```
sudo nano /etc/fstab
```

e aggiungiamo una linea siffatta:

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

Nel caso dell'esempio di cui sopra

```
Shared /media/vboxshared vboxsf defaults 0 0
```

Poi salviamo e, assicurandoci che non sia ancora montata la cartella, diamo un `sudo mount -a`

Se la cartella risulterà montata, allora possiamo stare certi che lo sarà ogni riavvio, altrimenti tentiamo di capire cosa c'è di sbagliato attraverso i messaggi di errore oppure cancelliamo la modifica.

### **ATTENZIONE:**

**la pena per un fstab mal fatto è l'impossibilità di avvio del sistema. Quindi assicuriamoci che sia tutto apposto prima di riavviare**

**NOTA BENE:**

La procedura sopra indicata per montare all'avvio una cartella condivisa, è simile per qualunque disco o dispositivo di cui volessimo imporre questo comportamento.;

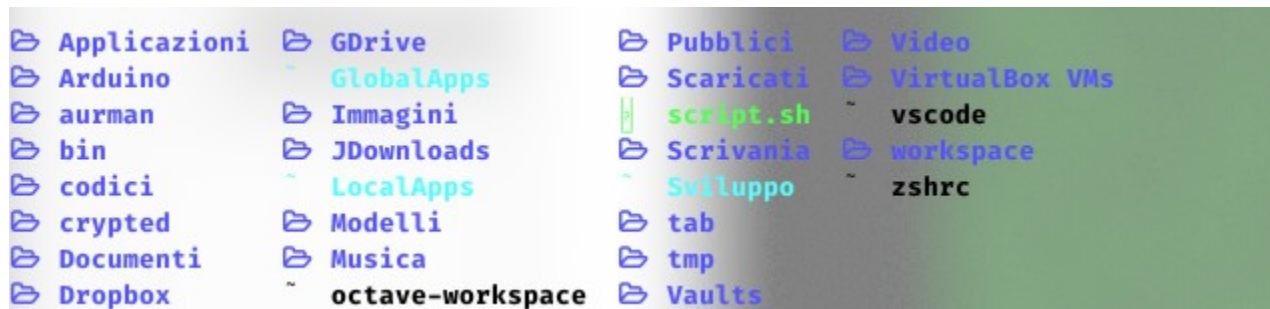
```
<NOME PUNTO DI MONTAGGIO> <PERCORSO PUNTO DI MONTAGGIO> <tipo partizione> defaults 0 0
```

In genere il nome del dispositivo di montaggio è un percorso del tipo `/dev/sdXY` dove X è una lettera rappresentante il disco o il dispositivo mentre Y è un numero rappresentate la partizione interna.

## 5.6 Sotto effetto di LSD

se avete aperto questa sezione sperando che si parli di erbe, sarete abbastanza delusi. Nonostante questo vi assicuro che `lsd` è un piccolo software che merita di essere approfondito.

Grazie ad esso quando vorrete vedere la lista dei file da terminale, saranno visualizzate anche delle icone simbolo (tipo quella della cartella, oppure la nota per i file musicali..etc)



*Figura 4: un esempio di output di lsd*

per installarlo basta dare:

```
sudo pacman -S lsd
```

Potete anche decidere di sostituirlo al normale `ls` ( solito strumento che da terminale fa vedere i file nella cartella) scrivendo nel file rc della shell questa riga:

```
alias "ls=lsd"
```

**NOTA BENE:**

Anche in questo caso potrebbe necessitare di un font in grado di visualizzare alcuni caratteri particolari. Consiglio sempre di installare il pacchetto font: **ttf-fira-code**

## 5.7 IL COMANDONE

Seguirà un elenco di comandi che include software o trick che normalmente applico io ad ogni installazione di archlinux. A voi il compito, se siete curiosi, di informarvi su cosa servono e perchè dovreste o non dovreste installarli!

```
#media tool
pakku -S clementine audacity vlc gimp 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

#work tool
pakku -S visual-studio-code-bin jdk8-openjdk jdk-openjdk jre-openjdk jre8-
openjdk openjdk-doc openjdk-src openjdk8-doc openjdk8-src terminator octave
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 textstudio python-pip mariadb

#net tool
pakku -S mailspring firefox firefox-i18n-it thunderbird thunderbird-i18n-it
deluge google-chrome

#misc tool
pakku -S ponysay lolcat redshift wine wine-mono winetricks wine_gecko
playonlinux steam steam-native-runtime ntfs-3g nitrogen xdotool rar zip
unzip p7zip sane hplip cups cups-pdf bluez-cups

sudo pip install youtube-dl

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

## 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](#)

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.

### 6.1 Template e legenda

Seguiranno dei template che sono stati usati spesso nella stesura del documento, con una piccola legenda introduttiva di come si vuole che siano intesi.

singole righe di codice sorgente

singola stringa di codice sorgente, in linea con altro testo

NOTA BENE:

Una nota basata sulle esperienze dell'autore

**SUGGERIMENTO:**

Un suggerimento orientato soprattutto ai meno esperti

```
blocco di codice sorgente o interno di un file
```

nei blocchi di codice l'introduzione di un cancelletto # a inizio riga è da intendersi per un commento

**ATTENZIONE:**

Un avvertimento. Il tratto appena spiegato non è stato testato oppure può fallire in alcuni casi ed è quindi sconsigliato da usare se non si sa esattamente quello che si fa. L'autore declina ogni responsabilità per danni che si possano verificare sul proprio sistema o dispositivo

tra parentesi angolate <> trovate il nome dei valori che dovrete sostituire voi.

Tutti i comandi dati con sudo è da sottointendere che necessitino dei permessi di amministratore, perciò se li operate da account root può essere evitato il sudo

