



GUIDA INSTALLAZIONE ARCHLINUX

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**. Se volete comunque un feeling Arch, una distribuzione userfriendly ma comunque rolling e basata sulla struttura di ArchLinux è senza dubbio **Manjaro**. 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.



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

Convenzioni

I comandi da shell verranno identificati con il simbolo \$, che indica l'inizio della shell utente.

Acronimi

AUR	Arch User Repository
CLI	Command Line Interface
DE	Desktop Environment
DM	Display Manager
GPT	GUID Partition Table
GRUB	GRand Unified Bootloader
GUI	Graphical User Interface
GUID	Globally Unique Identifier
TTY	TeleTYpewriter
UEFI	Unified Extensible Firmware Interface

Indice

1	Preparare il supporto di installazione	7
1.1	Metodo 1 da Linux : dd	7
1.2	Metodo 2 da Linux: copia su Fat32 (Consigliato UEFI)	7
1.3	Metodo 3 da Linux: GUI	8
1.4	Altri S.O.	9
2	Hello world, I'm ArchLinux! Nice to meet you	11
2.1	Preparazione del disco di installazione	11
2.2	Connessione al mondo esterno e installazione dei primi pacchetti	13
2.3	Prime configurazioni	13
2.4	Programma di installazione terminato	14
3	Configurazioni di sistema	15
3.1	Connettiamoci al mondo esterno e alcuni consigli iniziali	15
3.2	Server e driver	15
3.3	Aggiunta utente, creazione cartelle utente e cifratura home	16
3.3.1	Cifratura cartella home	16
3.4	Configurare pacman e installare aurman	17
3.5	Sincronizzare orologio di sistema e Hardware	18
4	DE e altri servizi	19
4.1	Plasma Desktop Environment (DE)	19
4.2	Display Manager	20
4.3	NetworkManager e servizi di rete	20
4.3.1	netctl ed eduroam	21
4.4	Systemd: <i>la nera bestia della morte</i>	21
5	Post-Personalizzazioni di sistema by PsykeDady	23
5.1	I sette 'nano'	23
5.2	Oh zsh, mio amore <3	24
5.3	Fish, un ulteriore avanzatissima shell	25
5.4	neofetch	25
5.5	Il COMANDONE da due milioni di dollari	26

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:

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

sostituendo a X la lettera del mezzo di installazione. Per ottenere il valore di X, inserire la pennina e digitare

```
$ fdisk -l
```

cercando nell'output quella che sembra essere la vostra pennina (in base a dimensioni e nome) e leggendone quindi le coordinate. Importante, sostituite a X soltanto la lettera che identifica la pennina, non anche il numero che la segue e che tipicamente indica la partizione.

In seguito all'installazione, per poter riutilizzare la pennina nuovamente, dovrete azzerarla con lo stesso tool:

```
$ sudo dd if=/dev/zero of=/dev/sdX 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 Unified Extensible Firmware Interface (UEFI).

Create quindi due directory dove montare la Iso e la USB di installazione:

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

e montate quindi la Iso utilizzando:

```
$ sudo mount -o loop </percorso/iso> iso
```

A questo punto, se non lo si è ancora fatto, formattate la pennetta in Fat32, supponendo che /dev/sdX sia il percorso della pennina, seguire queste istruzioni per formattarne il contenuto (effettuare le operazioni in ambiente su o aggiungere sudo prima di ogni comando):

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

```
# entra in fdisk, scrivere i prossimi comandi e premere invio
# dopo ciascuno
o
p
1
2048
# invio senza scrivere nulla oppure scegliere una dimensione
t
b
w
# Ora si esce da fdisk
$ mkfs.fat /devX1
$ mount /dev/sdX1 usb \nlcode
$ dosfslabel /dev/sdX1 ARCH\_AAAAMM \nlcode
$ # 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 documentarvi personalmente sulla guida ufficiale.

1.3: Metodo 3 da Linux: GUI

Se non amate molto sporcarvi personalmente le mani durante queste operazioni sono disponibili moltissimi programmi Graphical User Interface (GUI) 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.

Anche su sistemi Windows o MacOS esiste e funziona bene [Etcher](#). Altrimenti, potete usare alcuni dei programmi appositi.

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:

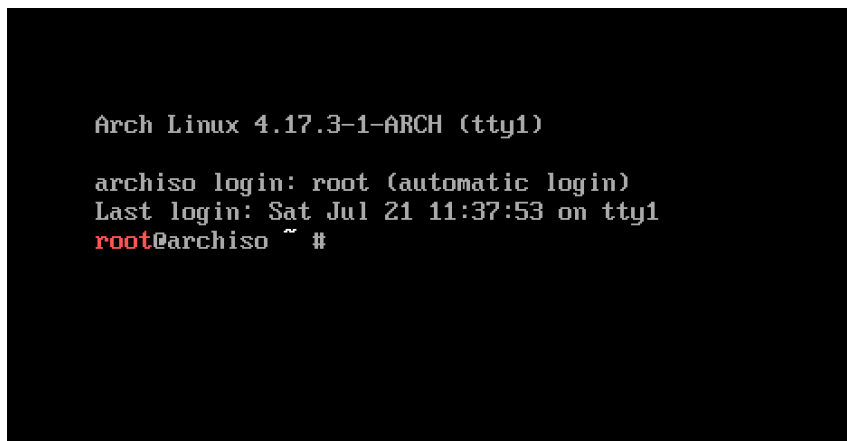
- Win32DiskImager
- USBWriter

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:



```
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 ~ #
```

Fig. 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
```

così vedrete meno madonnine volare in cielo...

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 ci siano altri sistemi operativi presenti. Supporrà inoltre di voler suddividere l'installazione in: `root`, `home` e `swap`. Un'altra condizione supposta sarà quella di avere un sistema UEFI con tutto quello che ne deriva.

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. Cominciamo dunque le operazioni dando

```
$ gdisk /dev/sda
```

In questo modo si entrerà in modalità `gdisk`, che installerà uno schema di partizioni di tipo GUID Partition Table (GPT). Se non si ha a che fare con UEFI, è consigliato usare `fdisk` o `cdisk`, e avere a che fare con schema di partizioni tradizionale.

Come per `fdisk` e `cdisk`, 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,
  in genere lo si dà uguale alla RAM per usufruire della
  funzione di ibernazione. Per pc con RAM maggiore di 4Gb non
  consiglio di usare la swap a meno di voler usare anche l'
  ibernazione
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
```

e poi 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: Connessione al mondo esterno e installazione dei primi pacchetti

Stop. senza internet non si va da nessuna parte.

Se quindi siete connessi via cavo, basta dare

```
$ dhcpcd
```

Se non avete il cavo di rete, archlinux fornisce una comodissima interfaccia di rete wireless, a cui potete accedere con:

```
$ 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
```

e, se la connessione effettivamente c'è, otterrete in output l'informazione 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.

A questo punto dobbiamo configurare e installare GRand Unified Bootloader (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 piaccia usare, se non lo conoscete sconsiglio vi il cui funzionamento non è immediato) 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
```

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 attraverso 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 aggiungere 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 `~/ .config/user-dirs.dirs` inserendo uno ad uno i nomi che desideriamo sostituire.

3.3.1: Cifratura cartella home

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 scrivervi 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 (dovrebbero 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 ArchLinux). 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 TeleTYpewriter (TTY) (utilizzando la combinazione di tasti `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 `umount`. Ripassate quindi al TTY per continuare con l'installazione.

3.4: Configurare pacman e installare aurman

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 Arch User Repository (AUR).

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: *aurman*¹ Per lo step successivo, è **fortemente consigliato** l'accesso con l'utente amministratore e non con root.

```
$ git clone https://aur.archlinux.org/aurman.git
$ sudo pacman -S expac python-regex
$ # se siete entrati come utenti, altrimenti senza sudo
$ cd aurman
$ # la prossima istruzione tende a bloccarsi se la linea non e'
  stabile, nel caso bloccate l'esecuzione con ctrl-c e
  rieseguitela finche' non avra' successo
$ gpg --recv-key 465022E743D71E39
$ makepkg -si
$ cd ..
$ rm -rf aurman
```

aurman (e anche *yay*) 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.

***aurman* e gli altri AUR helper non vanno mai eseguiti come utente amministratore con privilegi di amministratore (con `sudo` per intenderci) né come utente root.**

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)
```

¹jschiavon consiglia *yay* al posto di *aurman*, dato che quest'ultimo non è più supportato. La procedura cambia leggermente, ma è facile trovarla alla pagina GitHub di *yay*.

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)¹. 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. Mentre i primi due sono due DE completi, seppur leggeri, I3 è un Window Manager, quindi molto più leggero ma anche con meno funzionalità e una estetica *mooolto* più minimale.

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 applicativo di Instant Messaging 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.

4.1: Plasma DE

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:

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

per un installazione minimale invece digitare:

```
$ sudo 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:

```
$ sudo pacman -S kdeconnect
```

¹Ultimamente anche le prestazioni di GNOME stanno migliorando molto

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:

```
$ sudo pacman -S plasma-browser-integration
```

Se usate la lingua italiana potete installarne il supporto con:

```
$ sudo pacman -S kde-l10n-it
```

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

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

NOTA BENE:

Su molti DE (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.

4.2: Display Manager

Insieme a plasma, nella sua versione completa, verrà installato il suo Display Manager (DM) SDDM. Un DM 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 richiedendo l'inserimento della password. Alcuni DM vi impongono di scrivere anche il nome utente. In quel caso diventano banalmente delle alternative a xinit.

Si, anche xinit può essere visto come un DM 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 DM completo, dovrete abilitarlo come servizio di systemd. Tornando a SDDM ad esempio, per abilitarlo dovrete scrivere:

```
$ sudo systemctl enable sddm
```

Per poi riavviare. In generale, al posto di sddm, dovrete scrivere il dm da voi scelto (alcuni esempi sono: gdm, lightdm, slimdm, ...).

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 notificchino 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 non 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:

```
$ sudo 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:

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

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

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

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

4.3.1: 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:

```
$ sudo <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
WPACfgSection=(
    'ssid="eduroam"'
    'proto=RSN'
    'key_mgmt=WPA-EAP'
    'eap=PEAP'
    'identity="UsernameEduroam"'
    'password="PasswordEduroam"'
    '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.

4.4: Systemd: la nera bestia della morte

Systemd è un insieme di tool che gestiscono servizi e avvio del vostro sistema operativo su base GNU/Linux. Non entrerà 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 funzioni.

Vi spiegherò invece come interfacciarvi al gestore facilmente, elencando una serie di comandi e spiegandone l'uso nella tabella 4.1.

Comando	Spiegazione
<code>systemctl enable <nome></code>	abilita il servizio all'avvio, che viene quindi attivato ogni qualvolta accendete la vostra macchina
<code>systemctl start <nome></code>	avvia immediatamente il servizio
<code>systemctl restart <nome></code>	spegne e riavvia il servizio, utile se si stanno sperimentando problemi
<code>systemctl stop <nome></code>	spegne il servizio, il contrario di start
<code>systemctl disable <nome></code>	disabilita il servizio che non viene più avviato all'accensione del calcolatore, il contrario di enable
<code>systemctl status <nome></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.

Tab. 4.1. Alcuni utili e diffusi comandi per interagire con systemd

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

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

Al contrario invece:

```
$ sudo dhcpcd -x
$ sudo systemctl stop netctl
$ sudo 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 smarthome
set smooth

set linenumbers
set tabsize 4
set tabstospaces

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, ma imparate a indentare subito).

smarthome: dal momento che utilizzate l'indentazione (perché la usate), avere il tasto home che vi manda al punto giusto della riga è un gran bel plus.

smooth: normalmente nano permette di scorrere il testo a *blocchi*, mentre con questa opzione potrete farlo riga per riga.

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

tabsize: questa è la dimensione delle tabulazioni, utile soprattutto quando abbiamo un terminale piccolo per non trovarci continuamente con righe spezzate a causa dell'indentazione.

tabstospaces: converte i tab in insiemi di spazi.

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. Includendo i file *.nanorc vengono aggiunti tutti i linguaggi presenti di default, per aggiungerne uno voi trovate facilmente delle guide online.

In generale, consiglio di guardare il file `nanorc.sample` che di solito si trova in `/usr/share/doc/nano/`, copiandolo nella propria home

```
$ cp /usr/share/doc/nano/nanorc.sample ~/.nanorc
```

e poi modificando la propria versione locale con le indicazioni viste sopra.

5.2: Oh zsh, mio amore <3

Il terminale è vita, il terminale è amore.

Spesso per l'utilizzatore GNU/Linux, il terminale è davvero tutto. Ecco perchè abbellirlo e renderlo funzionale 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, uno dei gestori dei plugin di zsh. Qui entra finalmente in gioco AUR, quindi dovrete fare queste operazioni da account utente e non come amministratori:

```
$ aurman -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 repository 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"
```

perché il temi funzioni, comunque, bisogna anche copiare il tema stesso nella cartella temi di zsh:

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

Alternativamente potete creare un link alla cartella di origine del tema utilizzando `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 (ad esempio, se vi trovate in ambiente KDE Plasma, potete usare kate). 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 direttamente zsh da terminale 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

```
$ sudo pacman -S fish
```

Fish è un po' particolare da tanti punti di vista, ad esempio il suo file rc lo potete trovare in `~/.config/config.fish`, e per personalizzarlo dovreste 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 GNU/Linux piace avere tutto sotto controllo, piace che questo continuo monitorare sia esteticamente appagante e, soprattutto, piace far vedere agli altri la propria configurazione desktop. Ecco perché, chi per noi, ha creato dei tool che mostrano in modo sgargiante tutte le informazioni di cui abbiamo bisogno.

Uno di questi è neofetch, che possiamo semplicemente installare con

```
$ sudo pacman -S neofetch
```

Avviamolo almeno una volta scrivendo `neofetch`, ma se vogliamo possiamo poi modificare il file di configurazione `~/.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...)

5.5: 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 code 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 textstudio wine wine-mono
winetricks wine_gecko playonlinux steam steam-native-runtime
ntfs-3g python-pip mariadb catimg feh imagemagick jp2a
libcaca nitrogen w3m xdotool gst-plugins-good gst-plugins-bad
gst-plugins-ugly gst-plugins-base gst-libav gvfs

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 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.

