

# 1 Arch Linux

## 1.1 Maintenance

```
#check file size  
du -sh .cache/  
#remove a file  
rm -rt .cache/  
#delete what you don't need in .config file
```

specific maintenance:

```
#check the failed systems  
systemctl --failed  
#check the systemd journal  
sudo journalctl -p 3-xb  
#if the system doesn't boots then ctrl+alt+shift then timeshift -restore  
#then update mirrors  
#clar chache  
  
#then to update the whole system use:  
sudo pacman -Syyu  
#to check system updates  
sudo pacman -Syu  
#if you wan't to remove all packages in the drive use  
sudo pacman -Scc  
#remove all unwanted dependencies  
paru -Yc  
#remove orphan packages  
sudo pacman -Rns \$(pacman -Qdtq)  
#sudo pacman -Syyy Synchronise data use "mirror1"
```

## 1.2 Print in arch linux

install packages: usbutils, lusb, cups  
use this to make cups usable

```
sudo systemctl enable cups  
sudo systemctl start cups  
localhost:631  
  
lp -d HP_Officejey_Pro_8600]
```

## 1.3 configure date and time

```
hwclock --set --date = "04/32/2021 19:00:00"  
hwclock -hctosys
```

## 1.4 Configure wireless

```
#when entering an iso  
iwctl  
#then in the ui  
  
#to list all available devices  
device list
```

```

#to scan networks
station <device> scan

#to get networks
station <device> get-network

#to connect to a network
station <device> connect "<name of network>"

#to check if the connection is stable
ping -c s 8.8.8.8

#don't forget before rebooting the iso run
pacman nmtui

```

from Arch Water Linux

```

# to acces the gui for the internet
nmtui
# solve temporary failure in name resolution
# change the /etc/resolve.conf file to nameserver 8.8.8.8

# restart the resolved daemon
sudo systemctl restart systemd-resolved.service
# check that the daemon is running and active
sudo systemctl status systemd-resolved.service

```

dwm basic configuration

```

#MODKEY + shift + q to restart X server
startx # to start the X server

```

## 1.5 mount devices

mount usb sticks:

```

#to mount a usb stick
mount /dev/sdb1 /mnt/<destination folder>
#to unmount a sub stick
umount /dev/sdb1

```

mount an android device:

```

#to mount and android device
simple-mtpfs --device 1 tablet/

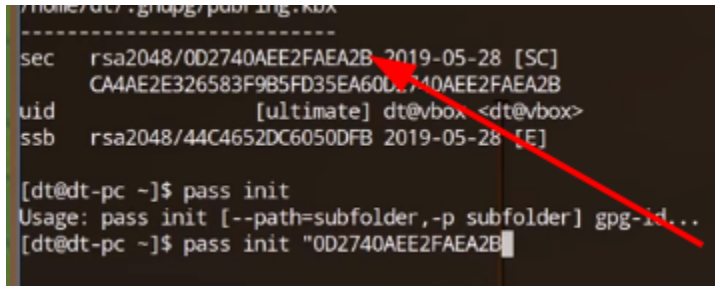
#to unmount an android device
fusermount -u /tablet

```

## 1.6 import export passwords from pass

export passwords:

```
# to list first the gpg keys  
gpg --list-secret-keys --keyid-format LONG
```

A terminal window with a dark background. The top part shows the output of 'gpg --list-secret-keys --keyid-format LONG', listing a secret key (sec), a user ID (uid), and a subkey (ssb). A red arrow points from the key ID '0D2740AEE2FAEA2B' in the 'uid' line to the 'pass init' command below. The bottom part shows the user running 'pass init' and seeing the usage instructions, then running 'pass init "0D2740AEE2FAEA2B"'.

```
-----  
sec  rsa2048/0D2740AEE2FAEA2B 2019-05-28 [SC]  
    CA4AE2E326583F985FD35EA60D2740AEE2FAEA2B  
uid      [ultimate] dt@vbox <dt@vbox>  
ssb  rsa2048/44C4652DC6050DFB 2019-05-28 [E]  
  
[dt@dt-pc ~]$ pass init  
Usage: pass init [--path=subfolder,-p subfolder] gpg-id...  
[dt@dt-pc ~]$ pass init "0D2740AEE2FAEA2B"
```

public key

```
# to create the export files  
# save this files in a usb and use it later  
gpg --output MY_FILENAME_public.gpg --armor --export GPG_PUB_KEY  
gpg --output MY_FILENAME_secret.gpg --armor --export-secret-key GPG_PUB_KEY  
# in other pc import the gpg keys  
gpg --import MY_FILENAME_pub.gpg  
gpg --allow-secret-key-import --import MY_FILENAME_sec.gpg  
# now copy the .password-store folder from the main machine and paste it into t
```