

Atelier StarinuX Raspberry Pi

Pascal Bessonneau

12/2016

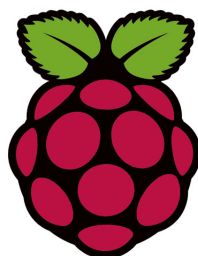


Table des matières

1	Petit historique du Raspberry Pi	4
2	Les caractéristiques du Pi 3	5
3	Formatage et écriture de la carte SD	6
3.1	Téléchargement de l'image	6
3.2	Sauvegarde de la carte SD	8
4	Personnalisation de la carte avant le gravage	10
4.1	Activer SSH	10
4.2	Édition des paramètres réseaux	10
4.3	Ajout d'un partition /home	12
4.4	Préserver la carte SD	14
4.4.1	Mettre la carte SD en lecture seule	14
4.4.2	Ajuster les paramètres pour préserver la carte SD	15
4.5	Démarrage	16
5	Connexion SSH à la Pi	18
5.1	Se connecter à la Pi	18
5.2	Créer une clef pour la connexion sans mot de passe	18
6	Les utilitaires de configuration de la Pi	20
6.1	raspi-config	20
6.2	rpi-update	20
7	Sécurisation de votre Raspberry Pi	21
7.1	Changer le mot de passe	21
7.2	Pare-feu	21
7.3	Mise à jour	24
7.4	Bloquer les connexions SSH non basées sur les clefs	24
8	Installer Privoxy	25
8.1	Installation	25
8.2	Configuration	26
9	Installer un serveur VPN	27
9.1	Installation	27
9.2	Ajouter la possibilité de se connecter en réseau local	31
9.3	Ajouter un nouveau client	33
9.4	Côté client	33
10	Application avec Python sur les ports GPIO	34
10.1	Qu'est ce que le GPIO ?	34
10.2	Eclairer une diode	34
10.3	PiFaceCAD	37

10.4 Sense HAT	38
11 Application avec la caméra	40
11.1 Description des caméras	40
11.2 Capturer des images	40
11.3 Capturer des vidéos	41
11.4 Timelapse video	41



1 Petit historique du Raspberry Pi

Le Raspberry a été conçu car si en 90 les adolescents faisaient leur arme sur des Amiga, Commodore, ... et passaient souvent des heures à programmer, pour leur homologues des années 2000 se n'était plus le cas. En effet ils étaient formés pour la bureautique et les sites web éventuellement. Avec le PC familial on ne peut plus prendre de "risque".

C'est pour contourner ce problème que les concepteurs du Raspberry Pi ont voulu concevoir un ordinateur qui, vu le prix, pourrait être bidouiller à loisir et pouvait servir de support pour la programmation.

De 2006 à 2008, Eben Upton a réalisé plusieurs prototypes de ce qui allait devenir le Raspberry Pi. A partir de 2008 les processeurs boostés par les applications mobiles rendaient possible de faire une carte pour laquelle la programmation et la diffusion de vidéo devenaient possibles pour un prix raisonnable.

Eben Upton, Robert Mullins, Jack Lang et Alan Mycroft en équipe avec Pete Lomas et David Braben ont créé la Fondation Raspberry Pi est l'équivalent d'une association de loi 1901 en France en 2009.

Il faudra attendre trois ans plus tard pour que la fabrication en série commence.

Février 2012 Disponibilité du Raspberry Pi A, le premier lot est vendu en quelques minutes

Mai 2012 La production atteint son rythme de croisière, le nombre de Pi par personne n'est plus de 1.

Septembre 2012 La fondation annonce que le Raspberry Pi sera désormais fabriqué en Angleterre

Octobre 2012 Le Raspberry Pi B est livré avec 512Mo

Janvier 2013 Un million de Raspberry Pi ont été vendus

Novembre 2013 on passe à deux millions

Février 2015 Sortie du Raspberry Pi 2

Avril 2015 5 millions de Raspberry Pi ont été vendus dont 500000 de Raspberry Pi 2

Novembre 2015 Le Raspberry Pi Zéro est en vente. En rupture de stock en moins de 24 heures

Janvier 2016 7 millions de Raspberry Pi vendus

Février 2016 Pour le quatrième anniversaire, le Raspberry Pi 3 apparaît

Septembre 2016 10 millions de Raspberry Pi vendus

La famille des Pis est représentée sur cette image (figure 1) faite par RaspberryTV.

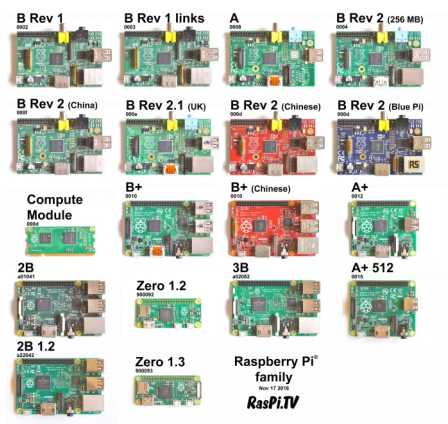


FIGURE 1 – Les différents modèles de Pi

2 Les caractéristiques du Pi 3

Décrire toutes les versions du Pi serait assez long, alors nous n'évoquerons ici que le Raspberry Pi 3.

Broadcomm est censé avoir libéré toutes les spécifications des circuits depuis 2014.

Le CPU est de la famille ARMv8 et est donc un processeur RISC. Vous retrouverez des ARM un peu partout maintenant puisque c'est le processeur le plus fréquent pour les smartphones, tablettes et autres joujous. Celui du Raspberry Pi 3 a quatre coeurs Cortex A53.

Il est 64 bits et compatible 32 bits. Toutefois il est bridé pour l'instant car le support 64 bits n'est apparu que dans une des dernières versions du kernel Linux. Pour l'instant avec la Raspbian, il est encore bridé en 32 bit.

Le GPU est quant à lui un VideoCore IV double coeur. Il est compatible OpenGL et supporte l'accélération matérielle : certains calculs 2D peuvent être déportés sur le GPU par le CPU.

Il possède une accélération matérielle pour ce qui est codé en H.264. Il supporte aussi le MPEG-2 et le VC-1 mais pour les utiliser il faut acheter une licence. Licence qui est associée au numéro de processeur du Pi.

Un composant spécifique gère à la fois le LAN et l'USB ce qui peut éventuellement générer un engorgement.

Les composants WiFi et Bluetooth sont pratiques mais ils n'ont une portée que d'une dizaine de mètres contrairement aux clefs USB qui ont une portée plus proche des 30m pour le WiFi.

Un détail qui a son importance, le Raspberry Pi n'a pas d'horloge interne. Elle se met à l'heure grâce à un serveur sur Internet. C'est utile de le savoir pour certaines applications.

Pour une somme modique des horloges internes sont disponibles à brancher sur le port GPIO.

3 Formatage et écriture de la carte SD

3.1 Téléchargement de l'image

Les différentes images proposées par la Raspberry Pi Foundation sont à cette adresse <https://www.raspberrypi.org/downloads/>.

Différents systèmes d'exploitation sont disponibles :

- NOOBS
- Raspbian
- Ubuntu Mate
- Snappy Ubuntu Core
- Windows 10
- OSMC
- LibreElec
- PINET
- RISC OS
- Weather Station

La NOOBS est un particulière. NOOBS signifie *New Out The Box Software* et a été développée spécifiquement pour le Pi par la Raspberry Pi Foundation. NOOBS n'est pas vraiment un système d'exploitation mais un gestionnaire de systèmes qui va simplifier l'installation de l'OS désiré mais également permettre le multiboot entre plusieurs systèmes présents sur la même carte SD. Solution idéale pour débiter ou pour tester les différents systèmes existants.

Les distributions OSMC et LibreELEC utilisent les bonnes capacités vidéos des Pi pour en faire des serveurs multimedia. Pour les audiophiles il existe des amplificateurs ou des DACs (Digital-to-Analogic Converter) de grande qualité pour le prix de 35 à 45\$.

Les images sont disponibles sous forme de fichiers compressés au format zip. Pour télécharger l'image :

```
wget https://downloads.raspberrypi.org/raspbian_lite_latest
```

La première étape consiste à vérifier qu'il n'y a pas eu de problème lors du téléchargement en vérifiant la somme de contrôle (SHA-1).

Après téléchargement, il suffit de taper :

```
$sha1sum 2016-11-25-raspbian-jessie-lite.zip
6741a30d674d39246302a791f1b7b2b0c50ef9b7 2016-11-25-raspbian-jessie-lite.zip
```

Il suffit de comparer alors le checksum du site web à celui affiché dans la console.

Pour décompresser l'image, il suffit de faire :

```
$unzip 2016-09-23-raspbian-jessie-lite.zip
```

Pour écrire l'image, vous pouvez vous aider de l'aide pour Linux et vous reporter à l'aide sur l'utilisation de "dd" ci-dessous.

Mais il ne faut pas faire d'erreur en spécifiant le disque. Si vous êtes frileux, il y a le logiciels etcher. Ce logiciel fonctionne aussi bien sur Linux que Windows et Mac OS.

Les images représentent les données brutes à écrire sur une carte SD. Le format « *img* » est aussi valable pour faire une image disque, etc.

L'utilitaire pour graver cette image est *dd* sous Linux.

Il y a trois arguments :

bs ou *block-size*. C'est la taille des blocs que va écrire l'utilitaire. Plus la valeur est grande plus l'écriture est rapide. Mais une taille trop grande peut générer des erreurs d'écriture. Le mieux est de choisir 1M ou 4M respectivement 1 méga-octet ou 4 méga-octets.

if *input file*, c'est la source. Ici cela va être notre image mais ça peut être un disque comme on le verra plus tard.

of *output file*, c'est la destination. Il faut faire extrêmement attention à la destination. En effet si vous écrivez sur un disque vous devez le faire généralement en tant que root et donc si vous vous trompez de disque vous allez PERDRE toutes les données du disque. Il convient de vérifier que l'on ne se trompe pas de destination.

Pour repérer le disque à écrire, la première possibilité est d'insérer la carte et de taper juste après la commande *dmesg* :

```
[ 6515.152466] usb 2-3.4: new high-speed USB device number 9 using xhci_hcd
[ 6515.242745] usb 2-3.4: New USB device found, idVendor=058f, idProduct=6362
[ 6515.242748] usb 2-3.4: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 6515.242750] usb 2-3.4: Product: Mass Storage Device
[ 6515.242751] usb 2-3.4: Manufacturer: Generic
[ 6515.242752] usb 2-3.4: SerialNumber: 058F312D81B
[ 6515.243201] usb-storage 2-3.4:1.0: USB Mass Storage device detected
[ 6515.244066] scsi host5: usb-storage 2-3.4:1.0
[ 6516.241274] scsi 5:0:0:0: Direct-Access          Generic  USB SD Reader      1.00 PQ: 0 ANSI: 0
[ 6516.241872] scsi 5:0:0:1: Direct-Access          Generic  USB CF Reader      1.01 PQ: 0 ANSI: 0
[ 6516.242416] scsi 5:0:0:2: Direct-Access          Generic  USB SM Reader      1.02 PQ: 0 ANSI: 0
[ 6516.242941] scsi 5:0:0:3: Direct-Access          Generic  USB MS Reader      1.03 PQ: 0 ANSI: 0
[ 6516.243594] sd 5:0:0:0: Attached scsi generic sg1 type 0
[ 6516.243889] sd 5:0:0:1: Attached scsi generic sg2 type 0
[ 6516.244143] sd 5:0:0:2: Attached scsi generic sg3 type 0
[ 6516.248541] sd 5:0:0:3: Attached scsi generic sg4 type 0
[ 6516.764148] sd 5:0:0:0: [sdb] 62333952 512-byte logical blocks: (31.9 GB/29.7 GiB)
[ 6516.765711] sd 5:0:0:0: [sdb] Write Protect is off
[ 6516.765713] sd 5:0:0:0: [sdb] Mode Sense: 03 00 00 00
[ 6516.767313] sd 5:0:0:0: [sdb] No Caching mode page found
[ 6516.767316] sd 5:0:0:0: [sdb] Assuming drive cache: write through
[ 6516.769552] sd 5:0:0:2: [sdd] Attached SCSI removable disk
[ 6516.772199] sd 5:0:0:1: [sdc] Attached SCSI removable disk
[ 6516.772727] sd 5:0:0:3: [sde] Attached SCSI removable disk
```

```
[ 6516.785521] sdb: sdb1 sdb2 sdb3
[ 6516.791350] sd 5:0:0:0: [sdb] Attached SCSI removable disk
[ 6517.205895] EXT4-fs (sdb2): mounted filesystem with ordered data mode. Opts: (null)
[ 6517.224103] EXT4-fs (sdb3): mounted filesystem with ordered data mode. Opts: (null)
```

Sur ce listing, on voit que la carte SD a comme nom de périphérique assigné `/dev/sdb`.

Pour vérifier que c'est ça, il faut utiliser *fdisk* :

```
$sudo fdisk /dev/sdb
```

Puis appuyer sur *p* (pour print, m pour l'aide).

Vous allez avoir quelque chose comme ça :

```
Disque /dev/sdb : 29,7 GiB, 31914983424 octets, 62333952 secteurs
Unités : sectors of 1 * 512 = 512 octets
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x99176dab
```

```
Périphérique Amorçage Start      Fin Secteurs  Size Id Type
/dev/sdb1          2048 62332927 62330880 29,7G  b W95 FAT32
```

On voit que la carte fait environ 30Go avec une partition formatée en fat32, c'est typiquement une carte quand vous la sortez de la boîte...

Vous pouvez aussi utiliser *blkid* qui est moins informatif ou bien, si la partition est montée *df -h*.

3.2 Sauvegarde de la carte SD

Avec "sdb" comme identifiant de disque, il faut utiliser la commande suivante pour écrire l'image :

```
$dd bs=4M if=2016-11-25-raspbian-jessie-lite.img of=/dev/sdb
$sudo sync
```

L'instruction "sudo sync" force le cache écriture à se vider pour pouvoir éjecter la carte en toute sécurité. Il faut pour retirer la carte attendre que l'ordinateur vous "rende" la main.

L'image une fois écrite ne remplit pas la carte... généralement. Sur la carte après écriture, il y a trois partitions :

1. une partition FAT32 (Windows/Linux) qui contient les fichiers de démarrage de la Pi
2. une partition ext4 (Linux) qui contient tous les fichiers de la distribution Raspbian
3. une zone vide non partitionnée et non formatée

Apparemment il y a un dispositif automatique pour étendre la partition ext4 avec l'espace disque non partitionnée sur les dernières versions de Raspbian. Donc si vous voulez faire une troisième partition comme un “/home”, il faudra partitionner AVANT de démarrer le Pi avec la carte sur un ordinateur GNU/Linux.

Après démarrage si l'espace non partitionnée subsiste vous pouvez utiliser “raspi-config” pour étendre la deuxième partition ou avec fdisk.

Dans le chapitre “personnalisation”, est abordé le cas où on veut faire une partition “home” séparée. Ce partitionnement est à faire sur un ordinateur GNU/Linux après l'écriture de la carte.

4 Personnalisation de la carte avant le gravage

4.1 Activer SSH

Maintenant pour des raisons de sécurité, sur Raspbian, SSH est désactivé par défaut. Ceci pour éviter, à cause des gens qui ne modifient pas les mots de passe par défaut, que les Pi ne constituent un réservoir de bots.

<http://www.framboise314.fr/une-mise-a-jour-de-securite-pour-raspbian/>
<https://www.raspberrypi.org/blog/a-security-update-for-raspbian-pixel/>

Pour activer le SSH, il faut créer un fichier s'appelant "ssh" dans la partition fat32 de démarrage ou passer par "raspi-config" en utilisant un clavier et un écran.

Sous Linux, rien de plus simple, il suffit de faire :

```
touch /media/pascal/boot/ssh
```

4.2 Édition des paramètres réseaux

Une fois branchée la Pi va récupérer une adresse IP auprès du serveur DHCP du réseau. Le plus souvent c'est votre box internet.

Il vous faudra donc regarder les logs (ou l'interface) de votre serveur DHCP (généralement votre box internet) pour voir quel IP à été attribué à la Pi pour réussir à vous connecter.

Si vous êtes sous Linux, vous pouvez trouver le Pi en scannant le réseau à la recherche des ordinateurs avec un serveur SSH accessible :

```
nmap -p 22 192.168.0.0/16
```

Sous Linux, vous avez la possibilité de fixer l'IP avant le démarrage, il suffit de faire :

```
cd ~  
mkdir pi  
sudo mount /dev/sdb2 pi
```

Ensuite le fichier */home/pascal/pi/etc/network/interfaces* est à éditer pour le personnaliser et donc donner une adresse fixe au Pi.

On retrouve les noms d'interface de debian, assez simples, pour les interfaces réseaux :

- eth0 pour le LAN
- wlan0 pour le wifi que ce soit pour le Raspberry Pi 3 ou une clef Wifi additionnelle sur une Raspberry plus ancienne.

Sur une Raspberry Pi 3 si vous rajoutez une clef USB WiFi, vous aurez un wlan0 et un wlan1 qui sont normalement la carte interne et la clef USB respectivement.

Dans le fichier "interfaces", on remplace la ligne de "eth0" par ce contenu :

```
allow-hotplug eth0
iface eth0 inet static
address 192.168.0.40
netmask 255.255.255.0
gateway 192.168.0.1
```

Idem pour le WiFi avec une modification pour accéder aux identifiants et mots de passe du réseau WiFi auquel on se connecte :

```
allow-hotplug wlan0
iface wlan0 inet static
address 192.168.0.41
netmask 255.255.255.0
gateway 192.168.0.1
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
```

Attention au cours de ces manipulations, il faut ne pas se tromper entre votre machine (*/etc/network/interfaces*) et le Pi (*/home/pascal/pi/etc/network/interfaces*). Sinon c'est un peu la catastrophe ;)

Et le fichier de configuration "wpa_supplicant.conf" est à modifier selon la configuration de votre réseau sans fil :

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

# à adapter selon la configuration de votre réseau
network={
    ssid="freebox_AERFTRE"
    proto=WPA RSN
    key_mgmt=WPA-PSK
    psk="maphrasedepasse"
}
```

Par la suite si vous voulez que l'IP ne soit pas fixée par la Pi, vous pourrez le remettre en automatique et fixer un bail DHCP en récupérant l'(es) adresse(s) MAC de la Raspberry Pi.

```
iface eth0 inet dhcp
```

ou pour wlan0

```
allow-hotplug wlan0
iface wlan0 inet static
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

```
$ifconfig
eth0      Link encap:Ethernet  HWaddr b8:27:eb:1a:34:64
```

```

            inet adr:192.168.0.50 Bcast:192.168.0.255 Masque:255.255.255.0
            adr inet6: fe80::ba27:ebff:fe1a:4048/64 Scope:Lien
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:3630835 errors:0 dropped:13 overruns:0 frame:0
            TX packets:3365928 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 lg file transmission:1000
            RX bytes:2510486112 (2.3 GiB) TX bytes:2571670733 (2.3 GiB)

lo        Link encap:Boucle locale
            inet adr:127.0.0.1 Masque:255.0.0.0
            adr inet6: ::1/128 Scope:Hôte
            UP LOOPBACK RUNNING MTU:65536 Metric:1
            RX packets:284 errors:0 dropped:0 overruns:0 frame:0
            TX packets:284 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 lg file transmission:1
            RX bytes:28401 (27.7 KiB) TX bytes:28401 (27.7 KiB)

wlan0     Link encap:Ethernet HWaddr 74:da:38:1a:b4:b8
            UP BROADCAST MULTICAST MTU:1500 Metric:1
            RX packets:0 errors:0 dropped:2 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 lg file transmission:1000
            RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

```

Par exemple ici l'adresse MAC pour eth0 est b8 :27 :eb :1a :34 :64 et pour wlan0 l'adresse MAC est 74 :da :38 :1a :b4 :b8.

4.3 Ajout d'un partition /home

Le plus simple est d'utiliser gparted ou parted. Ceci est à faire avant l'écriture de la carte depuis un ordinateur GNU/Linux.

Ici la carte est le périphérique "sdb" mais vous devez le remplacer par l'identifiant de votre disque.

Avec parted en ligne de commande, on va agrandir un peu la partition système :

```
$parted /dev/sdb
$print
```

```

Modèle: Generic USB SD Reader (scsi)
Disque /dev/sdb : 31,9GB
Taille des secteurs (logiques/physiques): 512B/512B
Table de partitions : msdos
Disk Flags:

```

Numéro	Début	Fin	Taille	Type	Système de fichiers	Fanions
--------	-------	-----	--------	------	---------------------	---------

1	4194kB	70,3MB	66,1MB	primary	fat16	lba
2	70,3MB	1390MB	1320MB	primary	ext4	

```
$resizepart 2
$2000MB
$print
```

```
Modèle: Generic USB SD Reader (scsi)
Disque /dev/sdb : 31,9GB
Taille des secteurs (logiques/physiques): 512B/512B
Table de partitions : msdos
Disk Flags:
```

Numéro	Début	Fin	Taille	Type	Système de fichiers	Fanions
1	4194kB	70,3MB	66,1MB	primary	fat16	lba
2	70,3MB	2000MB	1930MB	primary	ext4	

Pour ajouter la partition “/home” :

```
$mkpart
$primaire
$ext4
$2001MB
$31900MB
$print
```

```
Modèle: Generic USB SD Reader (scsi)
Disque /dev/sdb : 31,9GB
Taille des secteurs (logiques/physiques): 512B/512B
Table de partitions : msdos
Disk Flags:
```

Numéro	Début	Fin	Taille	Type	Système de fichiers	Fanions
1	4194kB	70,3MB	66,1MB	primary	fat16	lba
2	70,3MB	2000MB	1930MB	primary	ext4	
3	2001MB	31,9GB	29,9GB	primary	ext4	lba

```
$quit
```

```
$sudo mkfs.ext4 /dev/sdb3
```

Ensuite il faut éditer le fichier ‘/etc/fstab’ sur la carte.

proc			/proc	proc	defaults			0	0
/dev/mmcblk0p1	/boot	vfat	defaults			0	2		
/dev/mmcblk0p2	/		ext4	defaults,noatime	0		1		
/dev/mmcblk0p3	/home	ext4	defaults,noatime	0		1			

Enfin il faut copier le contenu du “home” de la pi sur la nouvelle partition :

```
$cd /media/pascal/0aed834e-8c8f-412d-a276-a265dc676112/home/  
$sudo cp -Ra * /media/pascal/f59b2c6c-b34c-45ec-b945-e01823d08bf5/
```

4.4 Préserver la carte SD

4.4.1 Mettre la carte SD en lecture seule

Certains, pour augmenter la durée de vie de la carte SD, monte la racine sur la clef en lecture seule. Ceci pour éviter les écritures des logs et autres qui sont fréquents et qui “abiment” la carte SD.

Avec la diminution du prix des cartes SD, cela devient moins nécessaire. D’expérience une carte SD utilisée sur un Pi peut tenir plusieurs années. En outre cela a un grave inconvénient si on utilise le Pi comme routeur ou comme serveur c’est qu’on perd les logs qui sont stockés en mémoire vive. Et les logs, c’est plutôt utile...

Un moyen de contourner serait d’écrire les logs sur une clef usb mais ça a grosso modo la même durée de vie qu’une carte SD (voire moins). Le mieux est sûrement de prendre une carte minimum genre 8Go qui n’est pas trop chère.

La manipulation pour passer la carte en lecture seule est décrite ci-dessous.

Tout d’abord il faut mettre en mémoire vive les fichiers d’ajustement du temps (la Raspberry n’a pas d’horloge interne).

```
$sudo ln -s /var/run/adjtime /etc/adjtime  
$sudo nano +61 /etc/init.d/hwclock.sh
```

Dans le fichier qui vient d’ouvrir à la ligne 61, il faut changer le -f (vrai si le fichier existe et est un fichier régulier) en -L (vrai si le fichier existe et est un lien symbolique).

Ce qui donne :

```
if [ -w /etc ] && [ ! -L /etc/adjtime ] && [ ! -e /etc/adjtime ]; then
```

Puis on crée un fichier cache dans “/etc/blkid/blkid.tab”. Nous allons le déplacer sur un ramdisk.

```
$sudo nano /etc/environment  
$BLKID_FILE="/var/run/blkid.tab"
```

Il faut éditer le fichier “/etc/fstab” :

```
proc /proc proc defaults 0 0  
tmpfs /tmp tmpfs nodev,nosuid,size=30M,mode=1777 0 0  
tmpfs /var/log tmpfs nodev,nosuid,size=30M,mode=1777 0 0  
/dev/mmcblk0p1 /boot vfat defaults,noatime,ro 0 2  
/dev/mmcblk0p2 / ext4 defaults,noatime,ro,errors=remount-ro 0 1
```

Puis il faut dans le fichier “/etc/default/rcS” changer la valeur de RAMTMP :

```
$sudo nano /etc/default/rcS
RAMTMP=yes
```

Enfin dans “/etc/profile” :

```
mount | grep ' on / ' | grep '(ro' && echo "Montage en lecture-écriture" &&
sudo mount -o remount,rw /
```

Et on reboote.

4.4.2 Ajuster les paramètres pour préserver la carte SD

Pour ménager la chèvre et le chou, on utilise les recommandations faites pour préserver les lecteurs SSD sur les PCs fixes.

Dans le fichier “/etc/fstab”, on change quelques petites choses :

```
proc /proc proc defaults 0 0
tmpfs /tmp tmpfs nodev,nosuid,size=30M,mode=1777 0 0
/dev/mmcblk0p1 /boot vfat defaults,noatime 0 2
/dev/mmcblk0p2 / ext4 defaults,noatime,errors=remount-ro 0 1
```

L’argument “noatime” est un moyen de supprimer l’écriture de la date de la dernière lecture ce qui minimise les écritures sur le disque.

De plus on crée une partition en mémoire pour /var/tmp dans lequel le système écrit beaucoup.

Puis il faut dans le fichier “/etc/default/rcS” changer la valeur de RAMTP :

```
$sudo nano /etc/default/rcS
RAMTMP=yes
```

Pour que les changements prennent effet, redémarrez votre Pi.

4.5 Démarrage

Le Raspberry Pi démarre de la façon suivante.

1. il va chercher un premier fichier bootcode sur la partition FAT32 qui est exécuté par le GPU
2. il va chercher un second fichier `start.elf` sur la partition FAT32 qui est exécuté par le GPU
3. le GPU va sonner le CPU pour qu'il démarre en passant le noyau avec les arguments qui vont bien
4. Le CPU se lance avec le kernel (noyau) et le démarrage devient celui de Linux :
 - chargement en mémoire du kernel
 - montage en lecture de la partition /
 - ...

Il est donc important de noter que contrairement à un ordinateur normal, le démarrage se fait à l'aide du GPU et non du CPU.

Les fichiers de démarrage du GPU *bootcode* et *start.elf* sont propriétaires et sont distribués sous la forme de binaire.

Le démarrage ne se fait pas avec les outils GNU/Linux habituel : *grub* ou *lilo* comme sur un PC normal. Le démarrage va dépendre de la présence de cette fameuse partition fat32 au début de la carte. Le choix de ce format permet toutefois aux utilisateurs de Windows de modifier les paramètres de démarrage.

Le fichier important sur un Raspberry est le fichier "config.txt". En effet ce fichier est utilisé pour personnaliser le démarrage du Pi en ajoutant la caméra, modifier la mémoire réservé au gpu, ... On y trouve aussi plein de paramètres pour configurer la sortie HDMI si vous lancez le Pi avec une interface graphique.

La mémoire d'un Raspberry est commune entre le GPU et le CPU. Selon l'usage que vous faites du Raspberry, par exemple utiliser ou non la caméra ou l'utiliser comme centre multimédia, il peut être intéressant de modifier la quantité de mémoire allouée au GPU. Le minimum pour le GPU est 16Mo. Il est nécessaire d'avoir 64Mo pour utiliser la caméra qui est aussi la valeur par défaut :

```
gpu_mem=16
```

Ici on règle la mémoire à 16Mo

Les valeurs maximales pour la mémoire attribuée au GPU sont quant à elles de 448Mo pour 512Mo de mémoire et 944Mo pour 1024Mo de mémoire.

Il y a aussi la possibilité de définir la mémoire en fonction de la taille mémoire installée sur le Pi. La spécification de la mémoire en fonction de la taille mémoire installée écrase le paramètre fixé par "gpu_mem".

```
gpu_mem_1024=64
gpu_mem_512=32
gpu_mem_256=16
```

Ici on fixe 64Mo pour une mémoire de 1Go (Raspberry Pi 3), 32Mo pour une Raspberry Pi2+, ... Ces paramètres sont utiles si vous personnalisez une carte pour l'installer sur différents Pi.

Les paramètres de “config.txt” pour la plupart ne sont pas nécessairement à taper à la main dans le fichier. En effet l'utilitaire “raspi-config” permet d'éditer le fichier via un menu plus convivial.

Il ya aussi le fichier “cmdline.txt” qui contient les paramètres à passer au noyau Linux au moment du démarrage.

5 Connexion SSH à la Pi

Pour se connecter à la Pi, il faut que la Pi et l'ordinateur soient connectés tous les deux sur ethernet et que la Pi soit accessible depuis l'ordinateur (pas de routeur ou de pare-feu entre les deux par exemple).

5.1 Se connecter à la Pi

Pour se connecter il suffit de taper :

```
$ssh 192.168.0.56 -l pi
$raspberrypi (c'est le mot de passe par défaut)
```

```
The authenticity of host '192.168.0.56 (192.168.0.56)' can't be established.
ECDSA key fingerprint is SHA256:jxN8A+IwAD+axlznP4wLME8Tpi36yCVW8duJmvA0yfs.
Are you sure you want to continue connecting (yes/no)?
```

```
$yes
```

```
Warning: Permanently added '192.168.0.56' (ECDSA) to the list of known hosts.
```

Les Pis lors du premier démarrage créent une clef unique pour le serveur, il n'y a donc pas besoin de régénérer les clefs pour le serveur SSH.

5.2 Créer une clef pour la connexion sans mot de passe

Vous pouvez le faire sur votre poste linux, utiliser putty-keygen ou le faire dans la pi.

Pour générer un clef si vous n'en avez pas déjà, il faut utiliser "ssh-keygen"

```
$ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/pi/.ssh/id_rsa):
$ (entrée) par défaut
Created directory '/home/pi/.ssh'.
Enter passphrase (empty for no passphrase):
$ (taper un mot de passe, vide pour pas de mot de passe)
Enter same passphrase again:
$ (confirmer le mot de passe)
Your identification has been saved in /home/pi/.ssh/id_rsa.
Your public key has been saved in /home/pi/.ssh/id_rsa.pub.
The key fingerprint is:
31:96:37:e1:0d:93:95:a2:51:ff:80:1a:79:be:bb:ad pi@raspberrypi
The key's randomart image is:
+---[RSA 2048]-----+
|          .o..      |
|          .+oB.     |
```

```

|      Bo*.+      |
|      ..0 . o    |
|      S . .      |
|      .          |
|      .          |
|      o          |
|      Eo.        |
+-----+

```

Pour activer la connexion par clef, il suffit de quelques modifications :

```

$mv .ssh/id_rsa.pub .ssh/authorized_keys
$vi .ssh/id_rsa
(copier la clef privée dans un fichier ‘clef_raspberry’ du répertoire .ssh
de votre ordinateur)
$rm .ssh/id_rsa

```

Renommer le fichier “id_rsa.pub” en “authorized_keys” permet notamment au serveur de reconnaître que l’ordinateur avec la clef privé correspondante (id_rsa) est autorisée à se connecter sur la Pi avec ce compte.

Il faut donc bien copier “id_rsa” sous ce nom ou un autre dans votre compte personnel et dans le répertoire .ssh pour un ordinateur GNU/Linux ou Mac OS.

Avec putty sous Windows vous trouverez de l’aide à cette page.

Sous GNU/Linux (et Mac OS), après il suffit de configurer ou de créer le fichier “.ssh/config” :

```

Host monpi
  Hostname 192.169.0.56
  User pi
  Port 22
  IdentityFile clef_raspberry

```

Vous pourrez vous connecter sans mot de passe simplement en tapant “ssh monpi”.

6 Les utilitaires de configuration de la Pi

Pour simplifier les choses pour les débutants, les concepteurs de la Raspbian ont inclus deux utilitaires spécifique à la Pi dans la distribution.

Le premier outil est *raspi-config*. C’est un utilitaire clique-bouton (NCURSES en mode console) pour accéder à des fonctions spécifiques de la Pi ou pour lancer des commandes Linux un peu complexes pour les débutants. On y trouve le jeu de caractères utilisé par la Pi, le fait d’activer ou non la PiCam, le SPI, ...

L’autre outil est *rpi-update* qui permet de mettre à jour le firmware de la Pi. Il est à lancer régulièrement pour avoir son système à jour. C’est différent des mises à jour du système Raspbian.

6.1 raspi-config

Quand vous lancez un Raspberry Pi, il est important de faire quelques réglages :

1. Changer le mot de passe
2. Internationalization Options, changer les locales (fr.FR.UTF-8 UTF-8)
3. Internationalization Options, changer le timezone (Paris)

Toutes ces commandes sont en fait des commandes debian mais elle sont plus simples à utiliser dans “raspi-config”.

Parmi les autres options, il y a l’activation de la caméra et dans les options avancées, il y a également des choses intéressantes :

Hostname Changer l’Hostname pour y accéder par ce nom plutôt que par l’ip

Memory Split la mémoire allouée au GPU

SPI utile pour activer la compatibilité avec des hat comme PiFace

...

6.2 rpi-update

Cet outil est lié au Raspberry, c’est pour récupérer et mettre à jour le firmware. Il faut l’installer absolument pour mettre à jour régulièrement le firmware de votre Pi.

```
$sudo apt-get install rpi-update
```

En plus des paquets, il est important de maintenir à jour son Pi y compris son firmware. Il faut éviter absolument de couper l’alimentation ou de couper la connexion SSH pendant la mise à jour du firmware.

La commande est très simple :

```
$rpi-update
```

7 Sécurisation de votre Raspberry Pi

7.1 Changer le mot de passe

La première étape est de changer le mot de passe par défaut. Il suffit de se connecter et de faire :

```
passwd
```

Choisissez un mot de passe le plus sécuritaire possible.

Vous pouvez aussi créer un utilisateur spécifique. Par exemple pour ajouter un utilisateur *pascal* :

```
$sudo adduser pascal
```

```
....
```

Il faut veiller si vous bricolez avec votre pi de lui donner les mêmes droits que l'utilisateur pi. Pour cela, il faut éditer le fichier */etc/groups*.

Ce sont surtout le groupe *sudo*, *SPI* qui sont utiles car ils permettront de lancer des commandes en tant que *root* et d'avoir le contrôle sur les interfaces de la pi.

Avec le nouvel utilisateur "username", vous devez écrire la commande suivante pour avoir la même configuration que l'utilisateur "Pi" (le tout sur une même ligne)

```
$sudo usermod -a -G adm,dialout,cdrom,sudo,audio,video,plugdev,games,users,input,netdev,spi,i2c,gpio username
```

Si vous voulez supprimer le compte pi par sécurité, il suffit de faire :

```
$sudo deluser pi
```

7.2 Pare-feu

Vous devez mettre un pare-feu. Surtout si vous utilisez le Pi comme serveur et donc qui est "face" à internet.

Attention, quelque soit le pare-feu, il faut laisser le port 22 ouvert pour vous connecter avec SSH sinon votre Raspberry Pi deviendra une brique... Il faut ouvrir le port et lancer le pare-feu au cours d'une session SSH et se connecter dans une nouvelle connexion pour vérifier que le port est bien ouvert !

Avant tout il est utile de sauvegarder un pare-feu vide au cas où les choses tournent mal.

```
$sudo iptables-save ~/iptables.empty
```

Vous pouvez faire un pare-feu simple "à la main" avec *iptables* :

```
#!/bin/sh

IPTABLES=/sbin/iptables

# Creation des tables
$IPTABLES -N TCP
$IPTABLES -N UDP

# Politique par défaut
$IPTABLES -P FORWARD DROP
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -P INPUT DROP

# Laisser passer les connexions existantes ou provenant du loop
$IPTABLES -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
$IPTABLES -A INPUT -i lo -j ACCEPT

# Bloquer les invalides
$IPTABLES -A INPUT -m conntrack --ctstate INVALID -j DROP

# Misc
$IPTABLES -A INPUT -p icmp --icmp-type 8 -m conntrack --ctstate NEW -j ACCEPT
$IPTABLES -A INPUT -p udp -m conntrack --ctstate NEW -j UDP
$IPTABLES -A INPUT -p tcp --syn -m conntrack --ctstate NEW -j TCP
$IPTABLES -A INPUT -p udp -j REJECT --reject-with icmp-port-unreachable
$IPTABLES -A INPUT -p tcp -j REJECT --reject-with tcp-reset
$IPTABLES -A INPUT -j REJECT --reject-with icmp-proto-unreachable

# Ouvrir le port 22 pour SSH
$IPTABLES -A TCP -p tcp --dport 22 -j ACCEPT

...
```

En cas de problème, utiliser le fichier qui permet de restaurer un pare-feu vide :

```
sudo iptables-restore ~/iptables.empty
```

Attention tous les fichiers du firewall doivent être accessibles seulement par le root.

```
$sudo chmod 700 02-firewall
$sudo chown root:root 02-firewall
```

Le fichier du firewall est à mettre dans le répertoire “/etc/network/if-pre-up.d”. Il sera exécuter dès que le réseau va devenir accessible. Vous pouvez aussi le placer en fin de lancement de Linux : dans le fichier “/etc/rc.local”.

Sinon il y a un paquet très efficace et pratique : *arno-iptables-firewall*. C'est un script BASH qui fait un pare-feu de bonne facture. Il faut indiquer les ports à laisser ouvert lors de l'installation. Il gère aussi des règles plus complexes : NAT, DMZ, ports ouverts sur le LAN et pas sur internet, ... Le fichier de configuration est assez simple et assez didactique.

Pour l'installer, c'est comme pour n'importe quel paquet Debian :

```
$sudo apt-get install arno-iptables-firewall
```

Lors de l'installation, il vous pose les questions suivantes :

1. Faut-il gérer automatiquement le pare-feu. La réponse est oui.
2. Les interfaces réseaux externes sont les interfaces réseaux qui seront filtrées par le pare-feu. Pour être le plus sécuritaire il vaut mieux toutes les ajouter en mettant : eth+ wlan+
3. Il demande si le DHCP est autorisé sur les réseaux externes. Si vous avez ajouter toutes les interfaces répondez oui. Si vous n'avez pas mis votre réseau local en interface externe vous pouvez répondre non.
4. Ports TCP ouverts. Il y en a un à rajouter absolument c'est le port 22 (du SSH) pour se connecter au Pi. Après vous pourrez en rajouter à loisir au fil de votre configuration.
5. Ports UDP ouverts. normalement il y en a pas sauf si vous installez un serveur OpenVPN par exemple.
6. Faut-il autoriser les ping. . . Si vous n'installez pas OpenVPN il vaut mieux répondre non.
7. Interfaces réseaux ethernes. CE sont les interfaces réseaux qui ne seront pas protégées. Par défaut c'est vide (cf 2.).
8. Faut-il redémarrer le pare-feu. Oui

Pour stopper le pare-feu, relancer et lancer le pare-feu les commandes sont respectivement :

```
$sudo service arno-iptables-firewall stop
$sudo service arno-iptables-firewall restart
$sudo service arno-iptables-firewall start
```

Pour le reconfigurer vous avez deux possibilités. Soit utiliser l'interface NCURSES en mode console en faisant :

```
$sudo dpkg-reconfigure arno-iptables-firewall
```

Soit en éditant le fichier “/etc/arno-iptables-firewall/conf.d/00debconf.conf” directement.

Il y a des réglages beaucoup plus avancés disponibles dans le fichier firewall.conf du répertoire “/etc/arno-iptables-firewall”.

7.3 Mise à jour

Il faut maintenir votre Raspberry Pi à jour... Pour tout ce qui est fourni par les paquets Debian, il suffit de faire de temps en temps un petit :

```
$sudo apt-get update
$sudo apt-get upgrade -y
$sudo apt-get dist-upgrade -y
```

Si la mise à jour contient une mise à jour du noyau, il faudra redémarrer le Pi.

Comme vu précédemment il y a aussi le firmware à mettre à jour avec la commande spéciale :

```
$sudo rpi-update
```

Le mieux est de créer un alias dans le fichier *.bashrc* :

```
alias update=sudo rpi-update && sudo apt-get update && sudo apt-get upgrade -y && sudo apt-get
```

Si vous avez installé un serveur web avec WordPress ou Yunohost il vous faudra aussi mettre à jour les composants en plus.

De temps en temps, des paquets ne sont plus nécessaires, dans ce cas il vous le signale quand vous faites des mises à jour. Il faut alors taper :

```
$sudo apt-get autoremove
```

7.4 Bloquer les connexions SSH non basées sur les clefs

Pour éviter les connexions qui utilisent un mot de passe au lieu de la clef privé/clef publique, il faut changer cette ligne (c'est la dernière ligne) dans "/etc/ssh/sshd_config" :

```
UsePAM no
```

Il faut aussi empêcher les connexions en tant que root (c'est la ligne 28) :

```
PermitRootLogin no
```

Puis il faut redémarrer le serveur :

```
$sudo service ssh restart
```

8 Installer Privoxy

Privoxy est un proxy qui ne met pas en cache mais filtre le contenu des pages lues pour enlever notamment les publicités et peut également modifier les pages web. Il va filtrer la majorité des publicités sur le protocole HTTP. En effet souvent les contenus publicitaires relèvent du protocole HTTP et non HTTPS même si la page est cryptée.

8.1 Installation

Pour l'installation il suffit de faire un :

```
$sudo apt-get install privoxy
```

Il y a quelques réglages à faire. Tout d'abord il faut le configurer pour qu'il écoute le réseau local.

```
$sudo nano +761 /etc/privoxy/config  
listen-address 192.168.0.56:8118
```

On va enlever le logging de privoxy pour éviter de stocker votre historique Web (bien que par défaut privoxy ne stocke que les événements graves). Vous devez le réactiver si vous avez des problèmes de navigation pour le débogage.

```
$sudo nano +455 /etc/privoxy/config  
#logfile logfile
```

Pour limiter l'accès à votre réseau local (c'est mieux...), il suffit de faire :

```
$sudo nano +1062 /etc/privoxy/config  
permit-access 192.168.0.0/24
```

On redémarre le serveur pour tenir compte des changements :

```
$sudo service privoxy restart
```

Ensuite il y a un nouveau port ouvert, le port 8118, il faut donc modifier le firewall. Soit avec :

```
$sudo dpkg-reconfigure arno-iptables-firewall
```

ou bien dans le fichier du firewall

```
$IPTABLES -A TCP -p tcp --dport 8118 -j ACCEPT
```

Ensuite il faut dans votre navigateur, régler le proxy :

Firefox je vous conseille d'utiliser l'extension FoxyProxy qui permet de changer à la volée le proxy

Chromium SwitchyOmega qui permet de changer à la volée le proxy

Si vous voulez que le proxy se configure automatiquement pour les machines de votre réseau. Il faut pouvoir ajouter une ligne à la configuration de votre serveur DHCP et mettre à disposition un fichier sur le serveur web. La manipulation est expliquée ici. Pour suivre ce tutoriel, il vous faut avoir un serveur DHCP sur une machine de type GNU/Linux.

8.2 Configuration

Les fichiers “user.action” et “user.filter” sont là que vous allez faire vos modifications.

Par exemple si vous souhaitez qu’un site ne soit pas filtré, comme numerama un super site que vous voulez aider en acceptant la pub. Il faut faire :

```
$sudo nano +158 user.action
.numerama.com
```

Par défaut, les cookies sont autorisés seulement s’il s’agit de cookies de session, si vous voulez qu’un cookie à plus long terme soit conservé, il faut ajouter le site dans la partie adéquate :

```
$sudo nano +90 user.action
.github.com
```

Vous pouvez utiliser le fichier trust pour limiter les web à visiter comme par exemple limiter l’accès seulement à quelques sites, pour les enfants par exemple.

```
$sudo nano +511 config
trustfile trust
```

Dans trust, vous pouvez les sites qui sont autorisés :

```
~.disneylandparis.fr
~.raspberrypi.org
```

N’oubliez pas de relancer le serveur.

9 Installer un serveur VPN

Le but est de créer un serveur VPN. Qu'est ce que le VPN ? C'est l'acronyme de Virtual Private Network. Il va créer un pont crypté entre deux ordinateurs et les requêtes réseaux vont être déportés sur le serveur. Toutes les requêtes réseaux du client vont sortir seulement à travers du tunnel crypté.

Son utilité ? Avoir son serveur VPN est utile par exemple quand vous êtes dans un Starbucks ou dans un hotel. Vous ne savez qui est à l'écoute sur le réseau et vos communications peuvent être interceptés. Pour éviter ce désagrément, vous activez le VPN et ainsi les communications seront cryptés et hors de portée de ceux qui écoutent. Les requêtes seront "déplacées" sur l'ordinateur serveur, votre Raspberry au chaud à la maison.

Il peut également être utilisé comme accès au réseau local pour permettre des réparations sur l'ordinateur d'un tiers (après avoir son accord). Ce système peut se substituer ou s'ajouter à un VNC (Virtual Network Connexion) : en effet vous pouvez sécuriser l'accès au VNC en passant par le tunnel sécurisé du VPN.

Il est à noter que des serveurs VPN commerciaux existent. Dans ce cas vous données sont cryptées de votre ordinateur à au site du VPN commercial et après passent en clair. Le stockage de vos données de connexion varient selon les opérateurs de VPN que vous utilisez.

Sur le Raspberry, on peut établir un réseau VPN de type PPTP qui n'est pas le protocole le plus sécurisé. Nous nous intéresserons au protocole le plus sécurisé : le protocole OpenVPN.

La cryptage est asymétrique avec une paire clef privée et clef publique. Il y a une paire de clefs pour le serveur et une paire pour le client.

9.1 Installation

L'installation à la main supposerait :

1. télécharger les paquets nécessaires
2. créer les deux paires de clefs
3. configurer le serveur
4. préparer le fichier client
5. installer la connexion VPN sur le client avec le fichier client

Pour en profiter, il faut installer git qui est un logiciel permettant de récupérer des applications à compiler (entre autres) :

```
$sudo apt-get install git
```

Fort heureusement un script disponible ici, facilite ça et le rend interactif :

```
$wget https://git.io/vpn -O openvpn-install.sh  
$sudo bash openvpn-install.sh
```

Le premier renseignement demandé est l'IP du Raspberry sur le réseau interne. Pour moi 192.168.0.56.

Le second est le port par défaut. Ce n'est pas la peine de le changer.

Ensuite vous devez choisir les DNS qui seront utilisés pour que le serveur puisse résoudre les noms de domaine en sortie du VPN. Je vous conseille OpenDNS bien que OpenDNS appartienne maintenant à Oracle. Le mieux est d'utiliser les serveurs de l'OpenNIC.

Ensuite il vous demande le nom du fichier à créer. Ici on va laisser le nom par défaut. Si vous avez plusieurs profils il vous faudra donner un nom plus explicite.

Il vous demande de fournir l'"external IP", c'est qu'il a remarqué que vous étiez derrière un NAT donc il demande l'adresse publique qu'il va mettre dans le fichier client pour que ce dernier puisse se connecter au serveur depuis internet.

Enfin il vous demande le mot de passe pour vous connecter au VPN. Vous devez choisir un mot de passe fort voire très fort car la plupart des applications permettent de le garder en mémoire.

Après le script va télécharger et installer les paquets. Il installe notamment :

- OpenVPN, qui est le programme permettant de se connecter à un VPN ou qui permet de faire un serveur VPN.
- Easy-RSA, sur un dépôt git, c'est le programme qui permet de créer facilement des clefs RSA pour le client et pour le serveur.

L'étape la plus longue est la création de la clef privée du serveur qui peut parfois prendre dix minutes sur un Raspberry Pi 2 si il y a un manque d'entropie. Pour aller plus vite dans la génération de la client, vous pouvez générer de l'entropie par exemple en téléchargeant quelque chose de gros sur le réseau dans une deuxième session :

Dans le répertoire courant, un nouveau fichier ".ovpn" est apparu. C'est le fichier client qui servira à configurer votre portable.

```
client
dev tun
proto udp
sndbuf 0
rcvbuf 0
remote 88.190.40.134 1194
resolv-retry infinite
nobind
persist-key
persist-tun
remote-cert-tls server
cipher AES-256-CBC
comp-lzo
setenv opt block-outside-dns
key-direction 1
```

verb 3

...

Dans le répertoire “/etc/openvpn”, vous trouverez les clefs du serveur ainsi que le fichier de configuration, “server.conf”, auquel il faut faire des modifications.

Il faut modifier la première ligne dans ce fichier de configuration en rajoutant :

```
local 192.168.0.56
```

Ce qui donne comme fichier de configuration :

```
local 192.168.0.56
port 1194
proto udp
dev tun
sndbuf 0
rcvbuf 0
ca ca.crt
cert server.crt
key server.key
dh dh.pem
tls-auth ta.key 0
topology subnet
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist ipp.txt
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
keepalive 10 120
cipher AES-256-CBC
comp-lzo
user nobody
group nogroup
persist-key
persist-tun
status openvpn-status.log
verb 3
crl-verify crl.pem
```

Le serveur openvpn se comporte pour ses clients comme un serveur DHCP. La ligne “server” indique une plage d’adresse doit être une plage d’adresse qui est *absolument différente* de la plage d’adresse que vous utilisez pour votre réseau interne.

Cette plage d’adresse contient les IPs des clients du VPN. Vous savez donc que cette plage d’adresse est le réseau “interne” pour le firewall. Vous avez aussi

besoin de savoir que l'interface réseau créée par un VPN est "tun0". Vous pouvez le vérifier en tapant :

```
$ifconfig
```

Pour ajouter les serveurs DNS de l'OpenNIC, il suffit de les choisir sur leur page.

On choisit par exemple 87.98.175.85 et 193.183.98.154. Cela nous donnerait comme fichier de configuration :

```
local 192.168.0.56
port 1194
proto udp
dev tun
sndbuf 0
rcvbuf 0
ca ca.crt
cert server.crt
key server.key
dh dh.pem
tls-auth ta.key 0
topology subnet
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist ipp.txt
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 87.98.175.85"
push "dhcp-option DNS 193.183.98.154"
keepalive 10 120
cipher AES-256-CBC
comp-lzo
user nobody
group nogroup
persist-key
persist-tun
status openvpn-status.log
verb 3
crl-verify crl.pem
```

Ensuite redémarrez le serveur.

Pour respectivement démarrer, éteindre et redémarrer le serveur OpenVPN, il suffit de taper :

```
$sudo service openvpn start
$sudo service openvpn stop
$sudo service openvpn restart
```

Il faut modifier le firewall pour qu'il fasse du NAT c'est-à-dire qu'il fasse suivre les paquets du réseau créé par le VPN vers les adresses extérieures. Pour ça, il faut un firewall sous Linux qui est correctement paramétré.

Pour ça, si vous utilisez arno-iptables-firewall, il suffit de modifier le fichier de configuration :

```
$sudo vi /etc/arno-iptables-firewall/conf.d/00debconf.conf
```

Et le fichier doit avoir cette tête :

```
EXT_IF="eth0"
EXT_IF_DHCP_IP=1
OPEN_TCP="22 8118"
OPEN_UDP="1194"
INT_IF="tun0"
NAT=1
INTERNAL_NET=""
NAT_INTERNAL_NET="10.8.0.0/24"
OPEN_ICMP=0
```

Vous remarquez les réglages pour le réseau interne et le fait qu'on ouvre un port en UDP au port 1194, c'est le port du serveur. Attention, le protocole est le protocole UDP et non TCP.

Ensuite il faudra que sur votre box, vous mettiez une redirection de ports. La manipulation se trouve partout sur internet car c'est la même que pour les applications de partage de fichiers.

Ici la redirection sera le port 1194 en UDP vers l'adresse du Pi sur le port 1194.

Pour un firewall "à la main" il faudrait rajouter quelque chose comme ça en fin de fichier de configuration du firewall :

```
$iptables --table nat --append POSTROUTING --out-interface eth0 -j MASQUERADE
$iptables --append FORWARD --in-interface tun0 -j ACCEPT
```

Ensuite il faut activer le NAT dans le kernel :

```
$sudo nano /etc/sysctl.conf
```

et décommentez la ligne :

```
net.ipv4.ip_forward=1
```

Puis il faut redémarrer ou exécuter :

```
$sudo sysctl -p /etc/sysctl.conf
```

9.2 Ajouter la possibilité de se connecter en réseau local

Pour utiliser votre imprimante, accéder à un ordinateur du réseau local, ... à travers le VPN pour un peu partout comme chez vous, il suffit de modifier le réglage du serveur.

Attention si ce réseau local du client et du serveur ont le même masque, les connexions en cours seront perturbées et les nouvelles connexions passeront par

le tunnel. Il vaut mieux pratiquer ce sport avec deux masques de réseau différent entre le réseau local du serveur et le réseau local du client.

Si le réseau serveur est en 192.168.0.0, il suffit d'ajouter une ligne à la configuration du serveur :

```
local 192.168.0.56
port 1194
proto udp
dev tun
sndbuf 0
rcvbuf 0
ca ca.crt
cert server.crt
key server.key
dh dh.pem
tls-auth ta.key 0
topology subnet
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist ip.txt
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 87.98.175.85"
push "dhcp-option DNS 193.183.98.154"
push \route 192.168.0.0 255.255.255.0"
keepalive 10 120
cipher AES-256-CBC
comp-lzo
user nobody
group nogroup
persist-key
persist-tun
status openvpn-status.log
verb 3
crl-verify crl.pem
```

Ensuite redémarrez le serveur.

Pour respectivement démarrer, éteindre et redémarrer le serveur OpenVPN, il suffit de taper :

```
$sudo service openvpn start
$sudo service openvpn stop
$sudo service openvpn restart
```

La ligne ajoutée est :

```
push \route 192.168.0.0 255.255.255.0"
```

Elle dit, redirige le trafic du 192.168.0.0 vers le tunnel VPN.

9.3 Ajouter un nouveau client

Pour ajouter un nouveau client, un autre ordinateur à la liste des ordinateurs pouvant se connecter, il faut simplement relancer le script et suivre les instructions.

9.4 Côté client

Il vous faut le fichier “.ovpn” créé tout à l’heure. Il contient toutes les informations pour vous connecter au serveur.

Sur Linux, il faut installer le paquet “openvpn” et si vous utilisez GNOME les paquets pour le gestionnaire de réseaux.

```
$sudo apt-get install openvpn
$sudo apt-get install network-manager-openvpn
$sudo apt-get install network-manager-openvpn-gnome
```

Pour KDE, pas besoin du paquet gnome...

Pour les gens sous Windows, le logiciel à installer est à télécharger sur le site d’OpenVPN. Il était dans le package que j’avais préparé en avance.

10 Application avec Python sur les ports GPIO

10.1 Qu'est ce que le GPIO ?

GPIO est l'acronyme de *General Purpose Input Output*.

C'est la série de broches situé sur un côté du Raspberry. Sa taille a varié entre le premier Raspberry et les suivants. On a gagné quelques broches.

Sur la figure 2, du site [element14](#), est représenté le GPIO pour le Raspberry Pi 3, il faut que vous cherchiez sur internet celui qui correspond à votre modèle de Pi si nécessaire.

Vous pouvez remarquer qu'il y a des alimentations de 3,3V et de 5V (pin 01, 02, 04, 17) selon les périphériques que vous voulez brancher. Et Il y a des masses *Ground*.

Les autres broches sont sur du courants de 3,3V. Le courant en sortie peut varier de 2 à 16mA.

Le GPIO permet de soit de contrôler soi même des éléments électroniques soit de mettre des extensions (HAT) pour avoir des périphériques. Vous en aurez à disposition le jour de l'atelier en petit nombre.

Le plus simple pour gérer ces sorties est le langage Python. Une bibliothèque Python est fourni pour contrôler le GPIO et aussi les périphériques qu'on branche dessus (le plus souvent).

Sinon vous avez la possibilité de contrôler le GPIO via un programme en C.

Pour installer les paquets nécessaires en python il faut faire :

```
$sudo apt-get install python-pip
$sudo pip install RPi.GPIO
```

10.2 Eclairer une diode

C'est la partie que je maitrise pas trop, c'est la partie électronique.






















Pour allumer une diode, il faut mettre la résistance adéquate pour ne pas griller la diode. Pour une tension de 3,3V et avec une diode qui fait baisser la tension de 0,7V il faut absorber $3,3 - 0,7 = 2,6V$. Avec la loi d'Ohms, $U = R/I$, on calcule pour un courant de 5mA la résistance nécessaire : $2,6/(5/1000) = 520ohms$. La résistance la plus proche est de 570 ohms.

Il faut chercher la résistance qui va bien. Installer fil, diode et résistance sur la breadboard.

Ensuite le programme Python est assez simple :

```
import RPi.GPIO as GPIO
import time

# Activer la bonne numérotation des pins
GPIO.setmode(GPIO.BCM)
```

Raspberry Pi 3 GPIO Header				
Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I ² C)		DC Power 5v	04
05	GPIO03 (SCL1 , I ² C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I ² C ID EEPROM)		(I ² C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40
				

Rev. 2
29/02/2016

www.element14.com/RaspberryPi

FIGURE 2 – Le GPIO du Raspberry Pi 3

```

# Si la diode est branché sur la pin 18.
# 1) activer la broche en sortie de courant
GPIO.setup(18, GPIO.OUT)
# 2) activer la diode
GPIO.output(18, GPIO.HIGH)

# Attendre 2 secondes
time.sleep(2)

# Eteindre la diode
GPIO.output(18, GPIO.LOW)

    C'est tout...
    Pour faire un programme qui va faire clignoter indéfiniment la diode.

import Rpi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

while True:
    GPIO.output(18, GPIO.HIGH)
    time.sleep(1)
    GPIO.output(18, GPIO.LOW)
    time.sleep(1)

    Attention en l'arrêtant avec controle+C vous allez laisser la broche dans un
    état ambigu.
    Il vaut mieux quitter proprement :

import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

while True:
    try:
        GPIO.output(18, GPIO.HIGH)
        time.sleep(1)
        GPIO.output(18, GPIO.LOW)
        time.sleep(1)
    except KeyboardInterrupt:
        break

GPIO.cleanup()

```

10.3 PiFaceCAD

Le PiFace est une carte avec un afficheur deux lignes 16 caractères, des boutons poussoirs et un capteur infra-rouge.

Pour l'utiliser il faut activer la sortie SPI dans 'raspi-config', redémarrer puis installer le paquet python-piface.

```
$sudo apt-get install python-pifacecad

import pifacecad
import time
import netifaces as ni

# Trouver l'adresse IP
ni.ifaddresses('eth0')
ip = ni.ifaddresses('eth0')[2][0]['addr']

# Allumer le PiFace
cad = pifacecad.PiFaceCAD()
cad.lcd.backlight_on()
cad.lcd.clear()

# Afficher au premier caractère sur la première ligne
cad.lcd.set_cursor(1, 0)
cad.lcd.write('Hello World')

cad.lcd.set_cursor(1, 1)
cad.lcd.write(ip)

sleep(5)

cad.lcd.backlight_off()
```

Pour utiliser les boutons poussoirs, on crée un processus qui va surveiller et faire un "goto" dans une fonction pour traiter l'information.

```
import pifacecad
import time

def update_pin_text(event):
    cad.lcd.clear()

    pin = event.pin_num + 0

    cad.lcd.set_cursor(1, 1)
    cad.lcd.write(pin)
```

```

        sleep(0.5)

    return()

listener = pifacecad.SwitchEventListener(chip=cad)

for i in range(8):
    listener.register(i, pifacecad.IODIR_FALLING_EDGE, update_pin_text)

listener.activate()

while True:
    try:
        pass
    except:
        break

```

10.4 Sense HAT

Il s'agit d'un "chapeau" ayant comme capteur :

- Gyroscope
- Accéléromètre
- Magnetomètre
- Capteur de temperature
- Capteur d'humidité
- Capteur de pression barometric

Il possède aussi un affichage avec une matrice de 8x8 en RGB.

Il est célèbre pour avoir voyagé dans l'espace.

Pour installer le package il faut faire :

```
$sudo apt-get install python-sense-hat
```

Ci-dessous un script pour stocker dans un fichier l'humidité et la température que j'utilise chez moi.

```

#!/usr/bin/python
import sys
import os
import datetime

from sense_hat import SenseHat

sense = SenseHat()
humidity = sense.get_humidity()
print("Humidity: %s %rH" % humidity)

```

```
temp = sense.get_temperature()
print("Temperature: %s C" % temp)

pressure = sense.get_pressure()
print("Pressure: %s Millibars" % pressure)

dateheure = datetime.datetime.now().strftime("%Y/%m/%d %H:%M:%S")

fh = open("temp.csv", 'a')
fh.write("%s;%s;%s;%s\n" % (dateheure,temp,humidity,pressure))
```

11 Application avec la caméra

11.1 Description des caméras

Les caméras Pi sont de deux générations, les premières ont un capteur de 3 millions de pixels et les secondes de 5 millions.

Attention quand vous manipulez ces caméras, elle sont très sensibles à l'électricité statique. Par conséquent avant de les prendre en main il faut toucher un objet à la terre comme la cage de votre ordinateur ou un radiateur.

Elles se branchent sur le port CSI, entre le port HDMI et le jack audio/video. Les parties conductrices du ruban doivent se trouver vers la prise HDMI.

Il y a deux types de caméras :

- la caméra standard. C'est une caméra qui prend des photos ou vidéos que dans le visible.
- la caméra IR. C'est une caméra qui peut capturer des infra-rouges. Attention il faut une source de lumière adaptée comme des LEDs infra-rouges ou un dispositif pour les caméras extérieures. Quand la lumière dans le visible est forte elle capture le "visible" comme la caméra standard. Elle a donc un double usage.

Pour activer la caméra il faut lancer "raspi-config" et dans le menu activer la caméra.

11.2 Capturer des images

Le programme pour capturer des images s'appelle raspistill.
pour capturer une image il suffit de faire :

```
raspistill -o testcapture.jpg
```

Pour régler la définition, il y a les arguments "w" et "h".

```
raspistill -w 1920 -h 1080 -o fullhdcapture.jpg
```

Attention il y a un temps de latence avant la prise de vue ! Par défaut ce temps est de 5 secondes.

Pour éliminer ce temps de latence il faut utiliser le paramètre "t". Le temps à indiquer est un entier en millisecondes.

```
# pas de temps de latence
raspistill -t 1 -o tensecondcapture.jpg
# temps de latence d'une minute
raspistill -t 60000 -o tensecondcapture.jpg
```

Le voyant de la caméra est rouge quand on prends un cliché.

11.3 Capturer des vidéos

Cette fois la commande est “raspivid”. Par défaut elle encode en h264, un format propriétaire.

```
raspivid -o testvideo.h264
```

Le temps de capture par défaut est de 5 secondes.

Pour le modifier, il suffit d'utiliser l'argument “t” toujours en millisecondes

```
raspivid -t 60000 -o testvideo.h264
```

Pour changer la résolution ce sont les mêmes arguments que pour “raspistill” :

```
raspivid -w 1280 -h 720 -t 60000 -o testvideo.h264
```

Selon les modèles de caméras, les possibilités en matière de définition varient.

11.4 Timelapse video

Vous pouvez programmer la caméra pour prendre des photos à intervalle régulier. Par exemple pour prendre durant une minute une image toutes les secondes :

```
raspistill -o frame%08d.jpg -tl 10000 -t 600000
```

Dans ce cas vous verrez des fichiers appelés frame00000001.jpg, frame00000002.jpg, etc. dans le répertoire courant.

Pour les transformer en vidéo, il faut utiliser un utilitaire “avconv”. Pour l'installer :

```
$sudo apt-get install libav-tools
```

Ensuite :

```
avconv -r 10 -i frame%08d.jpg -r 10 -vcodec libx264 timelapse.mp4
```

L'argument r indique le nombre de frames par secondes. L'argument vcodec indique le type de compression.

“avconv” est un outil très complexe avec lequel on peut beaucoup de choses.