Booter sur cle USB ou DD sur RaspberryPI3

-1-

Le Raspberry Pi 3 est capable de booter depuis un stockage USB, disque dur externe ou clé USB, ce qui peut être une bénédiction pour les systèmes ayant de nombreuses lectures écritures. Les cartes SD supporte mal ce genre d'exercices, j'en ai fait les frais plusieurs fois, principalement sur mon installation domotique à base de Jeedom, au bout de quelques semaines le système se corrompait. L'autre avantage est de supporté les vitesses de transferts propres à l'USB 2, le système est bien plus réactif que sur une simple carte SD.

L'opération est relativement simple, en quelques minutes on est capable d'activer le boot USB et le clonage du contenu de la carte vers le support externe. Tellement simple que cela peut aussi servir à créer des backups du système avec une clé USB de taille équivalente à la carte SD

Rendre compatible la partition /boot avec le lancement sur support USB

Si raspbian lite il faut rpi-update : sudo apt-get update; sudo apt-get install rpi-update sudo BRANCH=next rpi-update

Ensuite, activer la fonction

echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt #Suivi d'un reboot du Raspberry PI 3 pour que ça soit pris en compte sudo reboot

Après reboot, on vérifie que c'est actif, la bonne réponse est 17:3020000a

vcgencmd otp_dump | grep 17:

Maintenant vient le partitionnement du support externe,

Pour trouver l'identifiant du stockage externe, qui devrait normalement être /dev/sda sudo fdisk -l # Maintenant le partitionnement sudo parted /dev/sda

(parted) mktable msdos Warning: The existing disk label on /dev/sda will be destroyed and all data on this disk will be lost. Do you want to continue? Yes/No? Yes (parted) mkpart primary fat32 0% 100M (parted) mkpart primary ext4 100M 100% (parted) print Model: SanDisk Ultra (scsi) Disk /dev/sda: 30.8GB Sector size (logical/physical): 512B/512B Partition Table: msdos Disk Flags:

Number Start End Size Type File system Flags 1 1049kB 99.6MB 98.6MB primary fat32 lba 2 99.6MB 30.8GB 30.7GB primary ext4 lab # Pour en sortir, c'est un habituel "ctrl+c"

Création des deux partitions boot et root

sudo mkfs.vfat -n BOOT -F 32 /dev/sda1 sudo mkfs.ext4 /dev/sda2

Montage et transfert de la carte SD vers le support externe, c'est assez rapide avec rsync

sudo mkdir /mnt/target sudo mount /dev/sda2 /mnt/target/ sudo mkdir /mnt/target/boot sudo mount /dev/sda1 /mnt/target/boot/ sudo apt-get update; sudo apt-get install rsync sudo rsync -ax –progress / /boot /mnt/target

Optionnellement, régénérer une nouvelle clé SSH pour ce nouveau système

cd /mnt/target sudo mount –bind /dev dev sudo mount –bind /sys sys sudo mount –bind /proc proc sudo chroot /mnt/target rm /etc/ssh/ssh_host* dpkg-reconfigure openssh-server exit sudo umount dev sudo umount sys sudo umount proc

Dernière étape, rendre le nouveau boot actif sur le support externe

sudo sed -i “s,root=/dev/mmcblk0p2,root=/dev/sda2,” /mnt/target/boot/cmdline.txt sudo sed -i “s,/dev/mmcblk0p,/dev/sda,” /mnt/target/etc/fstab

Finir le taff en demontant et stoppant le système cd ~ sudo umount /mnt/target/boot sudo umount /mnt/target sudo poweroff

Déconnecter l’alimentation du Raspberry PI 3, retirer la carte SD, rebrancher

-2-

Raspberry Pi : déplacer Raspbian sur un disque dur externe ou une clé USB

Par défaut, le système d'exploitation Raspbian pour le Raspberry Pi est installé sur une microcarte SD.

On a l'avantage du faible encombrement et d'une consommation électrique très réduite.

Les inconvénients sont la lenteur de l'écriture sur la microcarte SD, sa faible longévité et selon la carte, le manque d'espace disque.

On peut choisir de déplacer le système :

- sur une clé USB pour disposer d'un espace plus important (peu de gains en revanche sur la durée de vie et la rapidité),
- sur un disque dur USB externe qui a tous les avantages sauf celui de la dépense électrique.

Le Raspberry Pi 3 est capable d'alimenter un disque externe. Pour le Raspberry Pi 2, il faudra prévoir une source d'alimentation électrique séparée.

Ce guide explique comment faire la migration du système depuis la carte SD sur le disque ou la clé USB. Préparation de la migration

Pour commencer, il est préférable d'arrêter le Raspberry Pi avant de brancher le disque ou la clé.

Dans cet exemple, j'ai pris une clé USB ayant déjà servi.

On se connecte déjà en SSH au pi (remplacer lambda_user par votre utilisateur sur le Pi)

soozx@soozx:~ $ ssh lambda_user@192.168.0.15

et on arrête proprement le Pi :

lambda_user@myweb:~ $ sudo halt

Après avoir débranché le Pi, on insère le disque USB puis on rebranche.

On se reconnecte sur le Pi :

soozx@soozx:~ $ ssh lambda_user@192.168.0.15

puis on liste les disques connectés :

lambda_user@myweb:~ $ lsblk NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT sda 8:0 1 7,3G 0 disk └─sda1 8:1 1 7,3G 0 part mmcblk0 179:0 0 58,6G 0 disk ├─mmcblk0p1 179:1 0 63M 0 part /boot └─mmcblk0p2 179:2 0 3,9G 0 part /

Nous trouvons ici deux disques :

- sda avec une partition sda1 non montée
- mmcblk0 avec deux partitions mmcblk0p1 montée sur /boot et mmcblk0p2 montée sur / qui est la racine du système d'exploitation.

L'objectif final sera de transférer / sur sda. Nous voyons que mmcblk0p2 a une taille de 3,9G ici. Il faudra au moins autant de place sur le disque dur.

Nous installons si ce n'est pas déjà fait l'utilitaire rsync qui nous permettra de synchroniser les fichiers depuis la carte SD sur le disque :

lambda_user@myweb:~ $ sudo apt install rsync

Préparation du disque dur externe

Nous allons maintenant préparer le disque dur externe.

Cette étape est inutile si vous disposez déjà d'une partition inutilisée de taille suffisante et formatée en ext4 sur le disque dur.

Attention, les données présentes sur le disque vont être effacées.

lambda_user@myweb:~ $ sudo fdisk /dev/sda

Welcome to fdisk (util-linux 2.25.2). Changes will remain in memory only, until you decide to write them. Be careful before using the write command.

Par sécurité, nous vérifions que nous sommes bien sur le bon disque avec la commande 'p' (print) qui va lister les partitions présentes :

Command (m for help): p Disk /dev/sda: 7,3 GiB, 7811891200 bytes, 15257600 sectors Units: sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes Disklabel type: dos Disk identifier: 0x0080ea51

Device Boot Start End Sectors Size Id Type /dev/sda1 * 2048 15257599 15255552 7,3G c W95 FAT32 (LBA)

Ici, nous avons un disque de 7.3G qui contient une seule partition en format FAT32.

Nous allons effacer la partition avec la commande 'd' (delete) :

Command (m for help): d Selected partition 1 Partition 1 has been deleted.

Si vous avez plusieurs partitions, répéter cette commande pour chacune d'entre elles.

Enfin, on écrit les changements sur le disque avec la commande 'w' (write) :

Command (m for help): w The partition table has been altered. Calling ioctl() to re-read partition table. Syncing disks.

Nous allons maintenant créer une partition pour accueillir le système Raspbian (le boot ou démarrage devra rester sur la carte SD) :

lambda_user@myweb:~ $ sudo fdisk /dev/sda

La commande 'n' pour new va permettre la création de la partition :

Command (m for help): n Partition type p primary (0 primary, 0 extended, 4 free) e extended (container for logical partitions) Select (default p): p [taper Entrée ou p]

Partition number (1-4, default 1): 1 [taper Entrée ou 1]

First sector (2048-15257599, default 2048): [taper Entrée] Last sector, +sectors or +size{K,M,G,T,P} (2048-15257599, default 15257599): [Entrée pour utiliser tout le disque, ou une taille suffisante par exemple +6G]

Created a new partition 1 of type 'Linux' and of size 7,3 GiB.

On écrit les changements avec 'w' :

Command (m for help): w The partition table has been altered. Calling ioctl() to re-read partition table. Syncing disks.

Nous allons ensuite formater la partition créée :

lambda_user@myweb:~ $ sudo mke2fs -t ext4 -L rootfs /dev/sda1

Nous pouvons passer à la migration du système sur le disque. Migration du système Raspbian sur le disque ou la clé USB

On commence par monter la partition nouvellement créée dans le répertoire mnt :

lambda_user@myweb:~ $ sudo mount /dev/sda1 /mnt

Si on a des services qui tournent comme mysql (mariadb), apache, php7.0-fpm, il est préférable de les arrêter :

lambda_user@myweb:~ $ sudo systemctl stop apache2 mysql php7.0-fpm

De même, si on a installé log2ram, il faut arrêter ce service sinon les logs ne seront pas recopiés sur le disque :

lambda_user@myweb:~ $ sudo systemctl stop log2ram.service

On peut même désactiver le service ensuite, car il n'a plus d'utilité si on a transféré l'ensemble du système sur un disque dur (par contre il reste utile si on se sert d'une clé USB) :

lambda_user@myweb:~ $ sudo systemctl disable log2ram.service

Puis on copie l'ensemble de la racine (/) présente sur la carte SD vers le disque dur monté sur mnt :

lambda_user@myweb:~ $ sudo rsync -avx / /mnt

(…)

var/tmp/

sent 985,682,776 bytes received 570,984 bytes 2,888,005.15 bytes/sec total size is 983,418,703 speedup is 1.00

On indique au programme de démarrage que le système se trouve maintenant sur le disque externe (par prudence, nous faisons une copie du fichier avant de le modifier si nous souhaitons revenir en arrière) :

lambda_user@myweb:~ $ sudo cp /boot/cmdline.txt /boot/cmdline.txt.bak

lambda_user@myweb:~ $ sudo nano /boot/cmdline.txt

On recherche dans la ligne root=/dev/ ou root=PARTUUID

boot/cmdline.txt avant modification

Nous allons remplacer root=/dev/mmcblk0p2 ou root=PARTUUID=xxxxxxxx-02 par root=/dev/sda1 :

/boot/cmdline.txt après modification

Ctrl-o pour enregistrer, Ctrl-x pour fermer.

Une autre modification de fichier pour que le programme de démarrage attende la mise en ligne du disque USB :

lambda_user@myweb:~ $ sudo nano /boot/config.txt

A la fin du fichier, nous ajoutons program_usb_timeout=1 :

Ctrl-o pour enregistrer, Ctrl-x pour fermer.

Enfin nous modifions la table des fichiers de fstab :

lambda_user@myweb:~ $ sudo nano /mnt/etc/fstab

mnt fstab dernière version

en remplaçant /dev/mmcblk0p2 ou PARTUUID=xxxxxxxx-02 par /dev/sda1 :

fstab modifié

Ctrl-o pour enregistrer, Ctrl-x pour fermer.

Redémarrage et vérification

La migration sur le disque USB est terminée, nous pouvons redémarrer :

lambda_user@myweb:~ $ sudo reboot

soozx@soozx:~ $ ssh lambda_user@192.168.0.15

On vérifie avec la commande lsblk :

lambda_user@myweb:~ $ lsblk NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT sda 8:0 1 7,3G 0 disk └─sda1 8:1 1 7,3G 0 part / mmcblk0 179:0 0 58,6G 0 disk ├─mmcblk0p1 179:1 0 63M 0 part /boot └─mmcblk0p2 179:2 0 3,9G 0 part

On voit ici que la partition / (c'est à dire le système d'exploitation) est maintenant montée sur sda1.

La partition mmcblk0p2 est quant à elle inutilisée. Elle contient toutefois la copie du système avant la migration.

-3-

raspberry-pi-3-booter-sur-un-disque-cle-usb

Le Raspberry Pi 3 est capable de booter depuis un stockage USB, disque dur externe ou clé USB, ce qui peut être une bénédiction pour les systèmes ayant de nombreuses lectures écritures. Les cartes SD supporte mal ce genre d'exercices, j'en ai fait les frais plusieurs fois, principalement sur mon installation domotique à base de Jeedom, au bout de quelques semaines le système se corrompait. L'autre avantage est de supporté les vitesses de transferts propres à l'USB 2, le système est bien plus réactif que sur une simple carte SD.

L'opération est relativement simple, en quelques minutes on est capable d'activer le boot USB et le clonage du contenu de la carte vers le support externe. Tellement simple que cela peut aussi servir à créer des backups du système avec une clé USB de taille équivalente à la carte SD

Rendre compatible la partition /boot avec le lancement sur support USB

Si raspbian lite il faut rpi-update : sudo apt-get update; sudo apt-get install rpi-update sudo BRANCH=next rpi-update

Ensuite, activer la fonction

echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt #Suivi d'un reboot du Raspberry PI 3 pour que ça soit pris en compte sudo reboot

Après reboot, on vérifie que c'est actif, la bonne réponse est 17:3020000a

vcgencmd otp_dump | grep 17:

Maintenant vient le partitionnement du support externe,

Pour trouver l'identifiant du stockage externe, qui devrait normalement être /dev/sda sudo fdisk -l
Maintenant le partitionnement sudo parted /dev/sda

(parted) mktable msdos Warning: The existing disk label on /dev/sda will be destroyed and all data on this disk will be lost. Do you want to continue? Yes/No? Yes (parted) mkpart primary fat32 0% 100M (parted) mkpart primary ext4 100M 100% (parted) print Model: SanDisk Ultra (scsi) Disk /dev/sda:

30.8GB Sector size (logical/physical): 512B/512B Partition Table: msdos Disk Flags:

Number Start End Size Type File system Flags 1 1049kB 99.6MB 98.6MB primary fat32 lba 2 99.6MB 30.8GB 30.7GB primary ext4 lab # Pour en sortir, c'est un habituel “ctrl+c”

Création des deux partitions boot et root

sudo mkfs.vfat -n BOOT -F 32 /dev/sda1 sudo mkfs.ext4 /dev/sda2

Montage et transfert de la carte SD vers le support externe, c’est assez rapide avec rsync

sudo mkdir /mnt/target sudo mount /dev/sda2 /mnt/target/ sudo mkdir /mnt/target/boot sudo mount /dev/sda1 /mnt/target/boot/ sudo apt-get update; sudo apt-get install rsync sudo rsync -ax –progress / /boot /mnt/target

Optionnellement, régénérer une nouvelle clé SSH pour ce nouveau système

cd /mnt/target sudo mount –bind /dev dev sudo mount –bind /sys sys sudo mount –bind /proc proc sudo chroot /mnt/target rm /etc/ssh/ssh_host* dpkg-reconfigure openssh-server exit sudo umount dev sudo umount sys sudo umount proc

Dernière étape, rendre le nouveau boot actif sur le support externe

sudo sed -i “s,root=/dev/mmcblk0p2,root=/dev/sda2,” /mnt/target/boot/cmdline.txt sudo sed -i “s,/dev/mmcblk0p,/dev/sda,” /mnt/target/etc/fstab

Finir le taff en demontant et stoppant le système cd ~ sudo umount /mnt/target/boot sudo umount /mnt/target sudo poweroff

Déconnecter l’alimentation du Raspberry PI 3, retirer la carte SD, rebrancher, prier, ça boot, Ayé

Comment démarrer à partir d'un périphérique de stockage de masse USB sur un Raspberry Pi Trad Google

Le démarrage USB est disponible uniquement sur les modèles Raspberry Pi 3B, 3B +, 3A + et Raspberry Pi 2B v1.2.

Ce tutoriel explique comment démarrer votre Raspberry Pi à partir d'un périphérique de stockage de masse USB tel qu'un lecteur flash ou un disque dur USB. Soyez averti que cette fonctionnalité est expérimentale et ne fonctionne pas avec tous les périphériques de stockage de masse USB. Consultez ce billet de Gordon Hollingworth pour savoir pourquoi certains périphériques de stockage de masse USB ne fonctionnent pas et pour obtenir des informations générales. Programme de démarrage USB

Le Raspberry Pi 3+ peut démarrer à partir de l'USB sans aucune modification, mais le Raspberry Pi 3 nécessite que le bit de démarrage USB soit défini dans la mémoire OTP (programmable une fois). Si vous utilisez un Raspberry Pi 3+, passez à la section suivante.

Pour activer le bit de démarrage USB, le Raspberry Pi 3 doit être démarré à partir d'une carte SD avec une option de configuration pour activer le mode de démarrage USB.

Une fois ce bit défini, la carte SD n'est plus nécessaire. Notez que toute modification apportée à l'OTP est permanente et ne peut pas être annulée.

Vous pouvez utiliser n'importe quelle carte SD exécutant Raspbian ou Raspbian Lite pour programmer le bit OTP. Si vous ne possédez pas une telle carte SD, vous pouvez installer Raspbian ou Raspbian Lite de la manière habituelle - voir Installation des images.

Commencez par préparer le répertoire / boot avec les fichiers de démarrage à jour (cette étape n'est pas nécessaire si vous utilisez la version 2017-04-10 de Raspbian / Raspbian Lite ou une version ultérieure):

```
$ sudo apt-get update && sudo apt-get upgrade
```

Activez ensuite le mode de démarrage USB avec ce code:

```
echo programme_usb_boot_mode = 1 | sudo tee -a /boot/config.txt
```

Ceci ajoute program_usb_boot_mode = 1 à la fin de /boot/config.txt. Redémarrez le Raspberry Pi avec sudo reboot, puis vérifiez que l'OTP a été programmé avec:

```
$ vcgencmd otp_dump | grep 17:
17: 3020000a
```

Vérifiez que la sortie 0x3020000a est affichée. Si ce n'est pas le cas, le bit OTP n'a pas été programmé avec succès. Dans ce cas, recommencez la procédure de programmation. Si le bit n'est toujours pas activé, cela peut indiquer une erreur dans le matériel Pi lui-même.

Si vous le souhaitez, vous pouvez supprimer la ligne program_usb_boot_mode de config.txt. Ainsi, si vous insérez la carte SD dans un autre Raspberry Pi, elle ne programme pas le mode de démarrage USB. Assurez-vous qu'il n'y a pas de ligne vierge à la fin de config.txt. Vous pouvez modifier le fichier config.txt à l'aide de l'éditeur nano à l'aide de la commande sudo nano /boot/config.txt, par exemple. Préparer le périphérique de stockage de masse USB

À partir de la version 2017-04-10 de Raspbian, vous pouvez installer un système Raspbian en état de fonctionnement sur un périphérique de stockage de masse USB en copiant l'image du système d'exploitation directement sur votre périphérique USB, de la même manière que vous le feriez pour une carte SD. Pour effectuer cette étape, suivez les instructions décrites ci-dessous, sans oublier de sélectionner le lecteur correspondant à votre périphérique de stockage de masse USB.

Une fois que vous avez fini d'imager votre périphérique de stockage de masse USB, retirez-le de votre ordinateur et insérez-le dans votre Raspberry Pi. Démarrez votre Raspberry Pi à partir du périphérique de stockage de masse USB

Connectez le périphérique de stockage de masse USB à votre Raspberry Pi et allumez-le. Après cinq à dix secondes, le Raspberry Pi devrait commencer à démarrer et afficher l'écran de démarrage arc-en-ciel sur un écran connecté.

Notez que si le bit de démarrage USB est défini, vous n'avez pas besoin d'insérer une carte SD dans le Raspberry Pi pour que le démarrage par USB fonctionne.

EEPROM de démarrage Raspberry Pi 4

Le Raspberry Pi 4 possède une EEPROM (4 Mo / 512 Ko) connectée à SPI, qui contient le code

permettant de démarrer le système et remplace le fichier bootcode.bin précédemment trouvé dans la partition d'amorçage de la carte SD. Notez que si un bootcode.bin est présent dans la partition de démarrage de la carte SD dans un Pi 4, il est ignoré. Pourquoi utiliser une EEPROM SPI?

La procédure de démarrage de Raspberry Pi 4 et la configuration de la mémoire SDRAM sont considérablement plus compliquées que sur les modèles Raspberry Pi précédents. Il y a donc plus de risques inhérents au code incorporé de manière permanente dans la ROM du SoC.

L'USB a été transféré sur un bus PCIe et le pilote Ethernet Gigabit est complètement différent des modèles précédents. Il était donc impossible de le fixer de manière permanente dans la ROM du SoC.

Une petite EEPROM SPI permet de corriger les bugs et d'ajouter des fonctionnalités après le lancement, sur le terrain

L'état modifiable localement signifie que les paramètres de mode d'amorçage OTP ne seront pas nécessaires pour l'amorçage PXE ou de stockage de masse USB sur le Raspberry Pi 4. Il n'y a pas de bits de mode d'amorçage OTP modifiables par l'utilisateur sur Pi4.

Démarrage PXE et USB

La prise en charge de ces modes de démarrage supplémentaires sera ajoutée à l'avenir via des mises à jour facultatives du chargeur de démarrage. La planification actuelle consiste à libérer d'abord le démarrage PXE, puis le démarrage USB. Le chargeur de démarrage fonctionne-t-il correctement?

Pour vérifier que le chargeur de démarrage fonctionne correctement, mettez l'appareil hors tension, débranchez tout le Raspberry Pi 4, y compris la carte SD, puis rallumez-le. Si le voyant vert clignote avec un motif qui se répète, le chargeur de démarrage fonctionne correctement et indique que start.elf n'a pas été trouvé. Toute autre action implique que le chargeur de démarrage ne fonctionne pas correctement et doit être réinstallée à l'aide de recovery.bin. récupération.bin

Si la mémoire EEPROM doit être mise à jour ou si elle est corrompue, elle peut être réinitialisée à l'aide d'une nouvelle carte SD avec une copie de recovery.bin dans la première partition d'une carte SD, formatée au format FAT.

recovery.bin est un utilitaire spécial qui s'exécute directement à partir de la carte SD et met à jour l'EEPROM. Il ne s'agit pas en soi d'un chargeur de démarrage. Le voyant vert clignote rapidement (pour toujours) en cas de succès. Comme ce n'est pas un chargeur de démarrage, il ne se chargera pas de démarrer *. Ainsi, dès que le voyant vert clignote rapidement, réinsérez une carte SD Raspbian normale et redémarrez le Pi. Protection en écriture de l'EEPROM

Raspbian ne mettra à jour ni l'EEPROM de démarrage ni l'EEPROM VLI (contrôleur USB) sans demander l'autorisation de l'utilisateur. Cependant, il est possible de protéger physiquement en écriture les deux EEPROM via un simple changement de résistance sur la carte. Les détails seront publiés dans les schémas