ATTENTION: un “sudo apt upgrade” a tendance à remettre de nouvelles versions de certains fichiers de configuration et donc à effacer certaines des modifications proposées ci-dessous !

Installer l’OS Ubuntu en suivant les étapes de ce lien -> [getting_started:os_installation_guide [ODROID Wiki]]

1. Configuration de Linux:
    a. Si le clavier n’est pas configuré correctement: menu → centre de contrôle → clavier . Si ceci est sans effet, on peut le faire en ligne de commande:
    sudo dpkg-reconfigure keyboard-configuration
    sudo dpkg-reconfigure locales

    b. Changer le nom de la machine:

    hostnamectl set-hostname PASTO_PBO41-22001
    - PASTO_ pour les pasteurisateurs
    - P pour les pilotes, F pour la version en fabrication
    - B pour le design B
    - O pour Odroid
    - 41 pour la version de MICHA
    - 22 pour l’année de fabrication
    - 001 pour le numéro de série dans l’année

    c. Changer les mots de passe de “root” et de “odroid” (commande “passwd” avec un “sudo su -” pour accéder à root…

    d. Mettre à jours les dépendances

    ```
    sudo apt update
    ```

    e. Pouvoir envoyer des emails et aussi pouvoir avoir un terminal virtuel au besoin:

    ```
    sudo apt install sendemail screen
    ```

    f. Cette opération doit être refaite après les “upgrade” du système
    Permettre un “`systemctl poweroff`” sans mot de passe administrateur…
    Si cette commande ne fonctionne pas (sans “sudo”) avec un message tel que:
    ==== AUTHENTICATING FOR org.freedesktop.login1.set-wall-message ===
    Dans le répertoire `/usr/share/polkit-1/actions` , il faut modifier le fichier dont le nom correspond au message obtenu avant la dernière section (ici, org.freedesktop.login1) avec l’extension “ .policy “.

    ```
    cd /usr/share/polkit-1/actions
    sudo nano org.freedesktop.login1.policy
    ```

Tout est expliqué ici:
https://winaero.com/how-to-enable-shutdown-and-reboot-for-a-normal-user-in-debian-jessie/
Il faut aller aux actions voulues (ici poweroff) ou encore nommées dans la dernière section du message d'erreur (ici, "set-wall-message") et remplacer le contenu des tags d'autorisation (<allow_xxxx) par "yes":

```
<allow_any>yes</allow_any>
<allow_inactive>yes</allow_inactive>
<allow_active>yes</allow_active>
```

g. Permettre la connexion à un deuxième Wifi: la brancher et y accéder par l'interface utilisateur afin qu'elle soit définie. Le mot de passe est stocké dans le "trousseau de clé": démarrer celui-ci et faire clic-droit sur les "Connexions", choisir "Mot de passe" et mettre un mot de passe vide: ceci rendra le mot de passe du Wifi accessible au démarrage.

   On doit modifier dans le fichier
   `/usr/share/polkit-1/actions/org.freedesktop.NetworkManager.policy`
   les permissions "<allow Yes" pour `settings-modify`, `network-control` et `wifi.scan`

   Il faut aussi modifier le fichier
   `/etc/NetworkManager/system-connections` comme expliqué ici:
   https://ubuntu-mate.community/t/automatic-wifi-connection-on-startup/3216/4
   Il s'agit essentiellement de rajouter une ligne avec `psk=`mot-de-passe et supprimer aussi la référence à l'utilisateur "odroid".

   On doit enfin faire (aussi?):

```
sudo nmcli connection delete TP-LINK_1BFC
sudo nmcli --ask dev wifi connect TP-LINK_1BFC ifname wlan1
```

   (remplacer TP-LINK_1BFC par le SSID du routeur Wifi local

2. Installation du serveur RDP pour un accès à distance::
   a. Installation

      sudo apt install xrdp

   b. Exécution automatique au démarrage

      sudo systemctl enable –now xrdp

   c. Ouverture du port 3389 pour trafic entrant

      sudo ufw allow from any to any port 3389 proto tcp

   d. Sur l'ordinateur distant, exécuter "Connexion au bureau à distance" (sous Windows) pour se connecter.
   e. Si par malheur, l'utilisateur "odroid" est en auto-login au démarrage (ce qui empêche toute connexion supplémentaire sur cet utilisateur par RDP), enlever la référence à cet utilisateur dans le fichier /etc/lightdm/lightdm.conf

Source: Informatique Beaujolaise: Accès à distance au bureau d'Ubuntu 20.04 depuis Windows 10 (xRDP) / problèmes de Look'n feel (informatique-beaujolaise.fr)

3. Installation du HOTSPOT Wifi
    a. Mettre le même nom pour le réseau que le hostname (par exemple PASTO_PBO41_22001)

        nmcli dev wifi hotspot ifname wlan0 ssid pastoB04001 password "past0.NET"

    b. Dans la gestion de ce réseau (NetworkManager “Modifier les connexions”), onglet “Général”, activer “Se connecter automatiquement”, priorité 0
4. Installation de la RTC
    a. Vérifier que le fuseau horaire est bien configuré dans Préférences/Time and Date Manager
    b. NE PAS suivre littéralement les (nombreuses) instructions pour le C4 (attention, pas le C2) dans: https://wiki.odroid.com/accessory/add-on_boards/rtc_shield#tab__odroid-c4hc41
    c. Pour être certain du hardware, faire “sudo apt install i2c-tools” et faire un “sudo i2cdetect 0” pour voir si le device 0x51 est bien présent.
    d. Avec “sudo nano /media/boot/config.ini”, il faut ajouter pcf8563 dans la ligne déjà existante (pas en créer une nouvelle): overlays="spi0 i2c0 i2c1 uart0 pcf8563" puis rebooter (et c’est tout ce qu’il faut faire en réalité!)
    e. après le reboot, vérifier que tout va bien avec:

        sudo dmesg | grep rtc
        sudo hwclock -r
        et rien faire de compliqué normalement…

    f. s’assurer que la RTC est lue au démarrage et initialise la date et l’heure en l’attente d’un accès à NTP: faire `sudo nano /etc/rc.local` et s’assurer d’insérer une ligne avant “exit 0” avec: `hwclock -s`

5. Configuration de l’environnement Arduino:
    a. Il faut installer la version 1.8 de l’IDE afin d’avoir une version ARM et un accès au gestionnaire de carte:
        i. Si une mauvaise version est installée, la désinstaller

            sudo apt autoremove arduino --purge

        ii. Se rendre sur Software | Arduino;
        iii. Télécharger la version ARM 64 bits;
        iv. Se rendre dans le dossier “Téléchargements” et décompresser le fichier téléchargé;
        v. Placer le dossier décompressé dans “odroid” et le renommer en “arduino”
        vi. Pour installer l’IDE, ouvrir le terminal:
            1. se placer dans le dossier arduino

cd arduino

2. installer

sudo ./install.h

b. Configurer pour les carte MKR:
   i. Ouvrir l'IDE dans le menu (onglet "Programmation") et se rendre dans outils → type de carte → gestionnaire de carte
   ii. Faire une recherche sur "MKR" et installer "Arduini SAMD Boards (32-bits ARM Cortex-M0+)"
   iii. Sélectionner le type de carte à programmer (ici la MKR Zero): outils → type de carte → Arduino SAMD → Arduino MKRZERO
c. Choisir le port sur lequel l'Arduino est connecté (ici ttyACM0): outils → Port → /dev/ttyACM0 (la carte Arduino doit au préalable être connectée)
d. Installer les librairies suivantes via croquis → inclure une bibliothèque → gérer les bibliothèques (remarque: la recherche y est très lente…)
   i. ArduinoRS485
   ii. ArduinoModbus
   iii. Flashstorage
   iv. SAMD21turboPWM

6. Configuration de python 3:
   a. Vérifier si python 3 est déjà installé (python3 -V), si ce n'est pas le cas, l'installer

      sudo apt install python3-pip

   b. Vérifier la version par défaut de python: python -V. Si ce n'est pas la version 3 qui apparaît, mettre python 3 par défaut (ici avec les versions 2.7.18 et 3.10) :
      i. sudo update-alternatives --install /usr/bin/python python /usr/bin/python2.7 1
      ii. sudo update-alternatives --install /usr/bin/python python /usr/bin/python3.8 2
   c. Vérifier si python 3 est bien par défaut: python -V
   d. installer pip3: sudo apt install python-pip3
   e. Installer les bibliothèques python:
      i. pip3 install web.py
      ii. pip3 install pyownet
      iii. pip3 install pySerial
      iv. pip3 install py-term (et non pas "term" !)
      v. pip3 install umodbus
      vi. Odroid.GPIO (suivre les instructions de pypi.org/project/odroid.GPIO) Cette installation est buggée. Il faut installer une copie locale de la librairie et la corriger:
         1. git clone https://github.com/hhk7734/Odroid.GPIO
         2. modify Odroid.GPIO/c_src/gpio.cpp (adding "wiringpi2/" in the #include)

3. cd Odroid.GPIO
4. python setup.py install
   (I was obliged to sudo this last command to have it work till the end)

vii. pip3 install spidev

7. Initialisation des GPIO
   a. `sudo su -` pour faire les opérations suivantes en tant que “root”...
   b. créer un fichier /home/odroid/initGPIO.sh avec le contenu suivant:

```
#!/bin/sh
# Initialization of pull-ups for GPIO input pins linked to the Pasteurizer
echo 483 > /sys/class/gpio/export
# echo 476 > /sys/class/gpio/export
echo 480 > /sys/class/gpio/export
# echo 477 > /sys/class/gpio/export
echo 482 > /sys/class/gpio/export
# echo 432 > /sys/class/gpio/export
# echo 495 > /sys/class/gpio/export
echo up > /sys/class/gpio/gpio483/pull
echo up > /sys/class/gpio/gpio480/pull
echo up > /sys/class/gpio/gpio482/pull
```

   On peut aussi ajouter cette ligne pour faciliter le fonctionnement du logiciel:

```
setcap 'cap_net_bind_service=+ep cap_sys_boot+ep' /usr/bin/python3.8
```

   c. et aussi le lancement du logiciel proprement dit dans un terminal virtual détaché (et rattachable: `screen -r`) :

```
su odroid -command “screen -d -m /home/odroid/Pasteurizer.sh”
```

   d. Rendre ce fichier exécutable: `chmod +x /home/odroid/initGPIO.sh`
   e. créer la définition d’un service à exécution unique à l’initialisation dans /etc/systemd/system/init_gpio.service :

```
[Unit]
Description=Initialize pull ups of some Input GPIO
[Service]
Type=oneshot
ExecStart=/bin/sh /home/odroid/initGPIO.sh
[Install]
WantedBy=multi-user.target
```

   f. lancer ce service: `systemctl enable init_gpio.service`
8. Configuration de git:
   a. installer git: `sudo apt install git`
   b. faire un “clone” du projet:

```
cd /home/odroid
git clone https://github.com/AKUINO/Pasteurizer.git
```

   c. vérifier que vous avez la bonne configuration pour votre hardware dans le répertoire `/home/odroid/Pasteurizer/configs` : trouver le fichier de configuration dont le nom a une correspondance entre la partie du nom de la machine (après “pasto” et avant “-”).

9. Préparer le lancement automatique:

a. créer le fichier de lancement dans /home/odroid/Pasteurizer.sh avec:

```
#!/bin/sh
cd /home/odroid
sleep 2
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
  sendemail -f dupriez@akuino.net -u "IP=$_IP" -m "http://$_IP/" -t
dupriez@destin.be dewez@akuino.net -s ssl0.ovh.net:587 -o
username=dupriez%squadratic.com -o password=***DEMANDER***
fi
cd /home/odroid/Pasteurizer
/usr/bin/python3.8 PastoWeb.py 80
sleep 20
```

10. OpenVPN:
    a. Sur le serveur central de l'OpenVPN:
    Un OpenVPN a été installé en téléchargeant openvpn-install.sh
    (`wget https://git.io/vpn -O openvpn-install.sh`)
    (voir
    https://www.cyberciti.biz/faq/howto-setup-openvpn-server-on-ubuntu-linux-14-04-or-16-04-lts/)
    Ce serveur central (IP interne au VPN: 10.8.0.1) est situé à l'IP général 54.37.22.109 (destin-informatique.com) ou autre. Spécifier Google comme DNS.
    b. On modifie la configuration du serveur OpenVPN pour que chacun est une IP fixe et des privilèges clairs: les pasteurisateurs n'ont accès à rien sauf à envoyer des données au serveur central. Les utilisateurs ont tout les droits.
    D'abord éditer /etc/openvpn/server/server.conf et y assurer:

```
local 54.37.22.109      ou tout autre IP du serveur central
port 1194
proto udp
dev tun0
ca ca.crt
cert server.crt
key server.key
dh dh.pem
auth SHA512
tls-crypt tc.key
topology subnet
server 10.8.0.0 255.255.255.0
push "redirect-gateway def1 bypass-dhcp"
#ifconfig-pool-persist ipp.txt
#ifconfig-pool 10.8.0.100 10.8.0.200
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"
keepalive 10 120
cipher AES-256-CBC
user nobody
group nogroup
persist-key
```

```
persist-tun
verb 3
crl-verify crl.pem
explicit-exit-notify
client-config-dir /etc/openvpn/ccd
```
répertoire où on assigne les IP aux utilisateurs
```
push "route 10.8.0.0 255.255.255.0"
```

c. Puis ajouter un fichier `/etc/openvpn/server/iptables-openvpn` contenant:

```
# Generated by xtables-save v1.8.2 on Sun Nov 14
20:56:43 2021
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A POSTROUTING -s 10.8.0.0/24 ! -d 10.8.0.0/24 -j SNAT
--to-source 54.37.22.109
COMMIT
# Completed on Sun Nov 14 20:56:43 2021
# Generated by xtables-save v1.8.2 on Sun Nov 14
20:56:43 2021
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:f2b-sshd - [0:0]
-A INPUT -p tcp -m multiport --dports 22 -j f2b-sshd
-A INPUT -p udp -m udp --dport 1194 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j
ACCEPT
-A FORWARD -s 10.8.0.0/25 -d 10.8.0.0/24 -i tun0 -j
ACCEPT
-A FORWARD -s 10.8.0.128/25 -d 10.8.0.1/32 -i tun0 -p
tcp --dport 9009 -j ACCEPT
-A FORWARD -s 10.8.0.128/24 -d 10.8.0.0/24 -j DROP
-A f2b-sshd -s 88.215.177.224/32 -j REJECT --reject-with
icmp-port-unreachable
-A f2b-sshd -s 103.100.158.57/32 -j REJECT --reject-with
icmp-port-unreachable
-A f2b-sshd -j RETURN
COMMIT
# Completed on Sun Nov 14 20:56:43 2021
```

d. Sur le serveur central mais pour chaque utilisateur privilégié ou pasteurisateur:
    i. Chaque utilisateur et chaque pasteurisateur doit recevoir une IP fixe.
        1. créer un répertoire /etc/openvpn/ccd

2. créer un fichier par utilisateur en donnant des IP entre 2 et 127. Par exemple pour “christophe”, créer un fichier de ce nom contenant: `ifconfig-push 10.8.0.2 255.255.255.0`
   Recommencer pour “sylvain”, etc.
3. créer un fichier par pasteurisateur en donnant des IP entre 128 et 254. Par exemple PASTO-PBO41-22001 avec comme contenu: `ifconfig-push 10.8.0.129 255.255.255.0`

ii. Chaque pasteurisateur doit recevoir un fichier de clé généré sur ce serveur central:
```
cd /ubuntu/OpenVPN
sudo ./openvpn-install.sh
```
Option = 1) Add a new client
Le nom de ce poste client doit correspondre au numéro de série (par exemple PASTO-PBO41-22001).
Le fichier résultant est mis dans /root : déplacez le dans /ubuntu/OpenVPN…
Ce fichier de clé doit être donné qu’à des personnes autorisées: toute autre personne pourrait en faire un mauvais usage.

e. Redémarrer le service OpenVPN du serveur:
```
sudo systemctl restart openvpn-server@server.service
```

f. sur chaque pasteurisateur:

i. installer le Client OpenVPN comme expliqué sur le site du logiciel:
https://openvpn.net/cloud-docs/openvpn-3-client-for-linux/
La distribution Ubuntu concernée est (pour le moment) “focal”.
Aussi:
https://community.openvpn.net/openvpn/wiki/Systemd

ii. Mettre le fichier de configuration dans /etc/openvpn/client avec une extension “.conf” (et non .ovpn)

- Pour tester une configuration, vous pouvez essayer:
```
openvpn3 session-start --config

/etc/openvpn/client/PASTO-PBO41-22001.conf
```

iii. Configurer le service pour un démarrage à chaque démarrage de l’ordinateur:
```
sudo systemctl enable openvpn-client@PASTO-PBO41-22001
```

iv. Redémarrer pour voir si tout est OK. Tester d’un autre ordinateur dans le VPN en faisant “ping” ou “http” de l’adresse. Exemple:
`ping 10.8.0.129` ou `http://10.8.0.129`