MoodleBox building reference*
Version 2.0 dev†

Nicolas Martignoni

October 18, 2022

Contents

†This is work in progress, do not use!

1 Introduction

This article documents how to build a MoodleBox. It is intended for developers or system administrators to provide background information on how a MoodleBox is built.

This is not meant to be an end-user documentation. For end-user documentation, please consult MoodleBox website.

Caveat

The actual build of the MoodleBox image is done using Ansible. Using Ansible enables most of the build to be automated, as well as ensuring that it is reproducible. Though complete, the instructions in this document should not be understood as a method to obtain a MoodleBox that is totally equivalent to the officially published MoodleBox images.

What MoodleBox does

- Configurable routed wireless access point.
- Internet router: If MoodleBox is connected via ethernet or wireless LAN to an Internet-connected network, it acts as a router and gives its clients access to the Internet.
- DHCP server for wireless clients.
- Moodle server (http://moodlebox.home/). This is a fully standard Moodle installation.
- MoodleBox plugin[1] providing a GUI to manage most aspects of the MoodleBox.

Specific Moodle features

- Moodle version 4.0.x in its basic configuration, with no content (no courses). The only Moodle user account is an administrator account (username: *moodlebox*, password: *Moodlebox4$*). The server is configured to accept clients from the official Moodle app.[2] The *cron* service is launched every minute.
- When a USB stick is inserted into the MoodleBox, its files are available to users in Moodle's *File system* repository.
- Ability to upload files via SFTP directly to the MoodleBox; these files are then available to users in Moodle's *File system* repository.
- Adminer, a full-featured database management tool, is installed (http://moodlebox.home/adminer.php).

What MoodleBox doesn't do

- Email server: MoodleBox is intended to be used "in the field", independently of any network infrastructure; email server functionality is not relevant for this purpose.
- Coffee machine.

[1] https://github.com/moodlebox/moodle-tool_moodlebox.
[2] https://download.moodle.org/mobile/.

2 Preparation of the Raspberry Pi

Although the MoodleBox works fine on other Raspberry Pi models, a Raspberry Pi Model 3B, 3B+ or 4B is recommended to **build** it.

Copy of Raspberry Pi OS Lite on a microSD card

We download the current version of *Raspberry Pi OS Lite (64-bit)* from the Raspberry Pi web site,[3] and copy it on a (good quality!) microSD card. A complete description of the process is available on Raspberry Pi web site.[4]

We then insert the SD card into a computer and mount it.

SSH access enabling

For security reasons, SSH access isn't enabled by default in Raspberry Pi OS. We must enable it, and to do so we create a file with name `ssh.txt` in the boot partition of the microSD card; this is the part of the SD card which can be seen when it is mounted in a macOS or Windows computer. The content of the file `ssh.txt` doesn't matter. The following commands will do this.

```
$ cd <mounting point of "boot" partition>
$ touch ssh.txt
```

Configuration of the main user account

For security reasons (again), the default image of Raspberry Pi OS Lite has no user account defined. We need to create one, by adding a file `userconf.txt` in the boot partition of the microSD card. The MoodleBox image uses for this account the username *moodlebox* and the password *Moodlebox4$*.

This file should contain a single line of text, consisting of `username:password`: so the desired username, followed immediately by a colon, followed immediately by an **encrypted** representation of the password to use.[5]

First login via SSH

Eject the microSD card from the computer and insert it into the Raspberry Pi.

Connect the power supply of the Raspberry Pi. Plug the Raspberry Pi with an Ethernet cable into a network with a DHCP server and wait 20-30 seconds. The Raspberry Pi is now reachable on the network using the address `raspberrypi.local`.[6]

[3] https://www.raspberrypi.com/software/operating-systems.

[4] See https://www.raspberrypi.org/documentation/computers/getting-started.html.

[5] See https://www.raspberrypi.com/documentation/computers/configuration.html.

[6] This network address is provided by the *zeroconf* standard protocol. Some older Android and Windows devices do not understand this protocol. With such devices, it is necessary to get access to the Raspberry Pi via its numerical IP address, which must be discovered manually.

From now on, all operations are done via the command line, using `ssh` (a regular terminal on macOS and Linux, or `Putty` on Windows).

We can now log in to the Raspberry Pi using the main account we've just setup: username is *moodlebox* with password *Moodlebox4$*.

```
$ ssh moodlebox@raspberrypi.local
$ moodlebox@raspberrypi.local's password:
```

Raspberry Pi OS upgrade

We begin by upgrading the Raspberry Pi's operating system and its installed packages, with these commands:

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

This step can take several minutes, depending on the available Internet connection speed. We just wait until it's finished.

Configuration of important settings

Launch *raspi-config* utility:

```
$ sudo raspi-config
```

With *raspi-config* utility, we configure the following settings.

- Expand Filesystem.
- Install following *locales*:
  - `de_DE.UTF-8`
  - `en_AU.UTF-8`
  - `en_GB.UTF-8`
  - `es_ES.UTF-8`
  - `fr_FR.UTF-8`
  - `it_IT.UTF-8`

  and set `en_GB.UTF-8` as default *locale*.
- Set time zone and WLAN country. This will also unblock wireless use. MoodleBox default settings are respectively `Europe/Zurich` and `CH`.
- Set the hostname to *moodlebox*.

From now on, to connect to the Raspberry Pi (via SSH or SFTP), we'll use the address `moodlebox.local`.

Let's reboot and log in to the MoodleBox with the default credentials set up page 4.

Switch to the alternative wireless chip firmware

The Raspberry Pi is usually used as a wireless client, and its wireless firmware has enhanced client features that are not useful for a MoodleBox, which is mainly used as an access point, not a wireless client.

However, the SRAM on the wireless chip is used both for data storage whilst in use, but also for these additional features and for bug fixes. Each time a feature is added or a bug fix made, the amount of SRAM available for runtime variables decreases, and so the number of clients in access point mode.

An alternative firmware is now available that has been tuned to maximise the number of clients in access point mode (AP) while still supporting client mode (STA). We want to use this alternative firmware in the MoodleBox. To switch the firmware to the alternative one, we launch the following commands.

```
$ cd /lib/firmware/brcm/
$ sudo ln -sf ../cypress/cyfmac43455-sdio-minimal.bin \
    brcmfmac43455-sdio.bin
```

Raspberry Pi firmware upgrade (optional)

In very specific and very rare occasions, it's advisable to upgrade the firmware of the Raspberry Pi. We only do this if we know why, since it could brick the device or the image! And we reboot immediately.

```
$ sudo apt-get install -y rpi-update
$ sudo SKIP_BACKUP=1 PRUNE_MODULES=1 rpi-update
$ sudo apt-get remove rpi-update
$ sudo reboot
```

Configuration of the system memory

To increase the system available memory, we reduce the memory reserved for the graphics chip to 16 MB. This is of no consequence, as our system does have no graphical interface. We do this by adding the following line at the end of the file `/boot/config.txt`:

```
gpu_mem=16
```

Configuration of the shutdown/startup hardware button feature

We enable shutdown/startup hardware button feature by further editing file `/boot/config.txt`. Insert following line after line beginning with `# Additional overlays` to get:

```
# Additional overlays and parameters are documented /boot/overlays/README
dtoverlay=gpio-shutdown
```

3 Wireless access point feature

We will now setup the wireless access point (AP) of the MoodleBox.

Configuration of the wireless interface for access point mode

We want to be able to use standard wireless interface `wlan0` to optionally connect to a wireless LAN, so we need another interface for our AP.

We use a *udev* rule to define this new interface, creating a file named `90-wireless.rules` in directory `/etc/udev/rules.d/`.

Here's the content of `/etc/udev/rules.d/90-wireless.rules` on the MoodleBox:

```
# File /etc/udev/rules.d/90-wireless.rules
# Add wireless interface for AP mode
ACTION=="add", SUBSYSTEM=="ieee80211", KERNEL=="phy0", \
  RUN+="/sbin/iw phy %k interface add uap0 type __ap"
```

Configuration of the wireless connection as a client (optional)

If MoodleBox is intended to be used as a client (STA mode) as well as an access point, a valid configuration file named `wpa_supplicant.conf` should be provided to `wpa_supplicant` service, in directory `/etc/wpa_supplicant/`.

Following listing shows an example of such file:

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

network={
    scan_ssid=1
    ssid="<Name of your wireless LAN>"
    psk="<Password for your wireless LAN>"
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    auth_alg=OPEN
}
```

Obviously the file should be edited to enter the real SSID and password of an existent wireless network.

Installation of the packages for the access point feature

The AP feature requires the packages `hostapd` and `dnsmasq`. Let's install them.

```
$ sudo apt-get install -y hostapd dnsmasq
```

Configuration of the static IP address

MoodleBox gets dynamically via DHCP its IP address on the ethernet interface `eth0` or as a wireless client on the `wlan0` interface.

With its routed access point feature, it will also act as a DHCP provider on its wireless interface `uap0`, so we need a static IP address on `uap0` interface. For the MoodleBox, we choose `10.0.0.1`. Any other private IP address can alternatively be used.

At the very end of file `/etc/dhcpcd.conf`, we add the lines

```
interface wlan0
# Uncomment following line to disable AP+STA mode
# nohook wpa_supplicant

interface uap0
static ip_address=10.0.0.1/24
nohook wpa_supplicant
```

Configuration of the access point

We can now configure the access point, by editing the file `/etc/hostapd/hostapd.conf`. This is where the access point SSID and password are set, as well as other options, such as the broadcast channel and regulatory country. We define *MoodleBox* as the SSID and *moodlebox* as the password.

Here's the content of `/etc/hostapd/hostapd.conf` on the MoodleBox:

```
# Set country code
country_code=CH
# Name of the Wi-Fi interface
interface=uap0
# Use the nl80211 driver
driver=nl80211
# Wi-Fi network name (SSID)
ssid=MoodleBox
# Show or hide SSID
ignore_broadcast_ssid=0
# Use the 2.4GHz band
hw_mode=g
# The Wi-Fi channel
channel=11
# Enable 802.11n
ieee80211n=1
# Enable WMM
wmm_enabled=1
# Enable 40 MHz channels with short guard interval for 20 Mhz
ht_capab=[HT40][SHORT-GI-20][DSSS_CCK-40]
# Accept all MAC addresses
macaddr_acl=0
# Use WPA authentication
auth_algs=1
```

```
# Use WPA2
wpa=2
# Use a pre-shared key
wpa_key_mgmt=WPA-PSK
# The network passphrase
wpa_passphrase=moodlebox
# Use AES, instead of TKIP
rsn_pairwise=CCMP
# Enable hostapd_cli
ctrl_interface=/var/run/hostapd
ctrl_interface_group=0
```

We now need to define where `hostapd` gets its configuration. This is set in file `/etc/default/hostapd`, where we replace the line `#DAEMON_CONF=""` with `DAEMON_CONF="/etc/hostapd/hostapd.conf"`. It should then have the following line:

```
...
DAEMON_CONF="/etc/hostapd/hostapd.conf"
...
```

Finally, we unmask and enable `hostapd` service:

```
$ sudo systemctl unmask hostapd
$ sudo systemctl enable hostapd
```

Configuration of the DHCP and DNS server

The DHCP and DNS server configuration on `uap0` interface is provided through `dnsmasq`. We edit `/etc/dnsmasq.conf` to have this content, where one notes the static IP address we set up on page 8:

```
interface=uap0           # Use interface uap0
listen-address=127.0.0.1 # Explicitly specify the address to listen on
listen-address=10.0.0.1  # Explicitly specify the address to listen on
bind-interfaces          # Make sure we aren't sending things elsewhere
server=9.9.9.9           # Forward DNS requests to external public DNS
server=149.112.112.112   # Forward DNS requests to external public DNS
domain-needed            # Don't forward short names
bogus-priv               # Don't forward addresses in the non-routed spaces
domain=home              # Set private domain name to 'home'
local=/home/             # Don't forward queries for private domain 'home'
address=/home/10.0.0.1   # Resolve subdomains '*.home'
expand-hosts             # Add private domain name to hostnames
dhcp-range=wifi,10.0.0.10,10.0.0.254,255.255.255.0,1h # Assign IP addresses
   with 1h lease, subnet name 'wifi'
dhcp-option=wifi,6,10.0.0.1 # Set DNS server for subnet wifi
txt-record=moodlebox.home,"MoodleBox by Nicolas Martignoni"
log-facility=/var/log/dnsmasq.log # Enable log
```

We edit now the file `/etc/hosts`, replacing the last line, beginning with `127.0.1.1`, with the following, containing the same static IP address defined earlier:

```
10.0.0.1    moodlebox
```

This configuration enables any device to access the MoodleBox via its URL http://moodlebox.home/, even those that do not implement *zeroconf*[7] standard protocol.

Finally, we fix a race condition between `dhcpcd` and `dnsmasq` by editing `dnsmasq` service file `/lib/systemd/system/dnsmasq.service`. We add following lines just before `[Install]`:

```
RestartSec=5
Restart=on-failure
```

Configuration of mDNS services advertising

In order to make MoodleBox services visible on the network, we create the file `/etc/avahi/services/moodlebox.service`, with following content.

```
<?xml version="1.0" standalone='no'?>
<!DOCTYPE service-group SYSTEM "avahi-service.dtd">
<service-group>
  <name replace-wildcards="yes">%h</name>
  <service>
    <type>_device-info._tcp</type>
    <port>0</port>
    <txt-record>model=MoodleBox</txt-record>
  </service>
  <service>
    <type>_ssh._tcp</type>
    <port>22</port>
  </service>
  <service>
    <type>_sftp-ssh._tcp</type>
    <port>22</port>
  </service>
  <service>
    <type>_http._tcp</type>
    <port>80</port>
  </service>
</service-group>
```

Routing configuration

We configure now the routing, so that the wireless clients can browse the Internet when MoodleBox is connected to an Internet router (via ethernet or wireless).

We edit the file `/etc/sysctl.conf`, uncommenting or adding the line

[7] https://en.wikipedia.org/wiki/Zero-configuration_networking

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

The installation of the package `iptables-persistent` enables routing rules to survive MoodleBox reboot or shutdown.

```
$ sudo apt-get install -y iptables-persistent
```

We can now define the routing rules:

```
$ sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
$ sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
$ sudo iptables -A FORWARD -i eth0 -o uap0 \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
$ sudo iptables -A FORWARD -i wlan0 -o uap0 \
    -m state --state RELATED,ESTABLISHED -j ACCEPT
$ sudo iptables -A FORWARD -i uap0 -o eth0 -j ACCEPT
$ sudo iptables -A FORWARD -i uap0 -o wlan0 -j ACCEPT
```

And we reboot.

```
$ sudo reboot
```

Installation and configuration of the captive portal

We can now install the captive portal, based on the free software Nodogsplash.[8] Nodogsplash needs first to be compiled for the Raspberry Pi platform. Nodogsplash documentation gives info about its compilation, which is not covered in this guide.

Let's install the Nodogsplash package we got after compilation:

```
$ sudo dpkg –i nodogsplash_5.0.0-1_armhf.deb
```

Nodogsplash configuration file `/etc/nodogsplash/nodogsplash.conf` should read (note the static IP address defined earlier)

```
# Nodogsplash Configuration File
GatewayInterface uap0
FirewallRuleSet authenticated-users {
    FirewallRule allow all
}
FirewallRuleSet preauthenticated-users {
}
FirewallRuleSet users-to-router {
    FirewallRule allow udp port 53
    FirewallRule allow tcp port 53
    FirewallRule allow udp port 67
    FirewallRule allow tcp port 22
    FirewallRule allow tcp port 80
```

[8] https://nodogsplashdocs.readthedocs.io/.

```
    FirewallRule allow tcp port 443
}
GatewayName MoodleBox
GatewayAddress 10.0.0.1
RedirectURL http://moodlebox.home/
GatewayPort 2050
MaxClients 50
SessionTimeout 360
```

We can now edit Nodogsplash splash page at our convenience. The files to edit are located in directory `/etc/nodogsplash/htdocs/`.

In the official MoodleBox image, Nodogsplash captive portal is not enabled. If we want to disable it too, we type following commands in our shell:

```
$ sudo systemctl stop nodogsplash.service
$ sudo systemctl disable nodogsplash.service
```

4 Installation of the LEMP stack

The LEMP software stack is a group of software that can be used to serve dynamic web pages and web applications. The term LEMP is an acronym that represents a Linux operating system with an Nginx (pronounced "engine-x", hence the E in the acronym) web server. The backend data is stored in a MariaDB database and the dynamic processing is handled by PHP.

Installation of Nginx and PHP

We install first Nginx and all needed PHP packages, notably those required by Moodle.

```
$ sudo apt-get install -y nginx php7.4-fpm php7.4-cli php7.4-common \
    php7.4-json php7.4-mbstring php7.4-opcache php7.4-readline \
    php7.4-xmlrpc php7.4-curl php7.4-gd php7.4-intl php7.4-soap \
    php7.4-mysql php7.4-xml php7.4-zip php7.4 php-apcu
```

Configuration of Nginx and PHP

Nginx web server configuration is set in file `/etc/nginx/sites-available/default`, which content should be like below.

```
# Default server configuration
#
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    listen 443 ssl;
    listen [::]:443 ssl;
```

```
    ssl_certificate /etc/nginx/ssl/moodlebox.pem;
    ssl_certificate_key /etc/nginx/ssl/moodlebox.key;

    root /var/www/moodle;

    error_page 404 /error/index.php;
    error_page 403 =404 /error/index.php;

    index index.php index.html index.htm index.nginx-debian.html;

    server_name moodlebox;

    # Hide all dot files but allow "Well-Known URIs" as per RFC 5785
    location ~ /\.(?!well-known).* {
        return 404;
    }

    location / {
        try_files $uri $uri/ =404;
    }

    location /dataroot/ {
        internal;
        alias /var/www/moodledata/;
    }

    location ~ [^/]\.php(/|$) {
        include fastcgi_params;
        fastcgi_split_path_info ^(.+\.php)(/.+)$;
        fastcgi_read_timeout    300;
        fastcgi_pass    unix:/var/run/php/php7.4-fpm.sock;
        fastcgi_index   index.php;
        fastcgi_param   PATH_INFO   $fastcgi_path_info;
        fastcgi_param   SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param   PHP_VALUE "max_execution_time=300\n
           upload_max_filesize=50M\n post_max_size=50M\n max_input_vars=5000";
        client_max_body_size    50M;
    }

    # Don't allow direct access to various internal files. See MDL-69333
    location ~ (/vendor/|/node_modules/|composer\.json|/readme|/README|readme
        \.txt|/upgrade\.txt|db/install\.xml|/fixtures/|/behat/|phpunit\.xml|\.
        lock|environment\.xml) {
        deny all;
        return 404;
    }

}
```

The `fastcgi_param PHP_VALUE` line, together with line `client_max_body_size`, sets vari-

ables specific to the MoodleBox typical usage. It increases to 50 MB the maximum upload file size, as well as script maximum execution time to 300 s. It also sets `max_input_vars` to 5000.

The ownership, group and access rights of Nginx and PHP should now be changed, so that the default user *moodlebox* can easily edit the files, while allowing Moodle updates and plugin installation via the web interface.

To do this, we edit the two files `/lib/systemd/system/nginx.service` and `/lib/systemd/system/php7.4-fpm.service`, adding after the line `[Service]` the line `UMask=0002`, to get in both files something like

```
...
[Service]
UMask=0002
Type=...
...
```

We then edit the file `/etc/php/7.4/fpm/pool.d/www.conf` to set the correct group ownership for PHP, replacing line `group = www-data` with `group = moodlebox`, so we have now

```
...
user = www-data
group = moodlebox
...
```

If needed, variables in `/etc/php/7.4/fpm/pool.d/www.conf` can be tweaked, e.g. for performance gains. See section 10 for examples.

We then relaunch the web server:

```
$ sudo systemctl restart nginx php7.4-fpm
```

SSL certificate and key

The SSL certificate is needed to prepare HTTPS feature of the MoodleBox. We copy the SSL certificate `moodlebox.pem` and its key `moodlebox.key` to the directory `/etc/nginx/`.

The certicificate is not used by default, but without it, Nginx setup above will not be valid. If HTTPS is not required, this can be left out, but the lines 7 to 10 of Nginx configuration must then be deleted.

SSL certificate generation is not covered in this guide, but any help can be found in any adequate documentation on SSL certificate generation, e.g. on https://stackoverflow.com/a/10176685.

MariaDB installation and configuration

We install now MariaDB package. PHP MariaDB support was already installed with PHP (see section ??).

```
$ sudo apt-get install -y mariadb-server
```

During the installation, we set the root password of the MariaDB. For this installation, we set the password as earlier (see page 4).

In order to allow flexible access to all the databases, a new database user is created in MariaDB, with all privileges. We choose again the same credentials for consistency.

```
$ sudo mysql
    > CREATE USER 'moodlebox'@'localhost' IDENTIFIED BY 'Moodlebox4$';
    > GRANT ALL PRIVILEGES ON *.* TO 'moodlebox'@'localhost';
    > FLUSH PRIVILEGES;
    > QUIT;
```

If needed, variables in `/etc/mysql/mariadb.conf.d/50-server.cnf` can be tweaked, e.g. for performance gains. See section 10 for examples.

5 Installation and configuration of Moodle

We have now setup the Raspberry Pi as a fully functional routed wireless access point with a complete LEMP stack. It's time to install and configure Moodle.

Creation of the database for Moodle

We first create the database which Moodle will use.

```
$ sudo mysql
    > CREATE DATABASE moodle;
    > QUIT;
```

Moodle download

We then download Moodle with Git in order to facilitate future updates. First, let's install Git.

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

We get now Moodle source in the appropriate location, specifying the current stable branch of Moodle. To save space, we use a shallow clone.

```
$ sudo git clone --depth=1 -b MOODLE_400_STABLE \
    git://git.moodle.org/moodle.git /var/www/moodle
```

Creation of Moodle data directories

Moodle data and several other directories are now created.

```
$ sudo mkdir -p /var/www/moodledata/repository /var/www/moodledata/temp \
    /var/www/moodledata/backup /var/cache/moodle \
    /var/cache/moodle-cache-backup
```

Adequate permissions and ownership are set on these directories, including SGID permission on Moodle data directory. We also set the correct permissions to Moodle source directory:

```
$ sudo chown -R www-data:moodlebox /var/www/moodle /var/www/moodledata/ \
    /var/cache/moodle /var/cache/moodle-cache-backup
$ sudo chmod -R ug+w,o-w /var/www/moodle /var/www/moodledata/ \
    /var/cache/moodle /var/cache/moodle-cache-backup
$ sudo chmod -R g+s /var/www/moodledata/
```

We can now launch Moodle installation, either by loading URL http://moodlebox.home/ in a browser and follow the instructions on screen, or by using Moodle command line interface (preferred).[9] During the installation, we set the administrator account with the usual credentials.

When using the command line interface, this steps takes at least 10 minutes.

We finally set the adequate permissions and ownership on Moodle's configuration file `/var/www/moodle/config.php`:

```
$ sudo chown www-data:moodlebox /var/www/moodle/config.php
$ sudo chmod ug+w,o-w /var/www/moodle/config.php
```

Moodle configuration

Moodle is now installed and works normally. We configure it for MoodleBox.

Moodle course backup directory

As we use for Moodle course backup a different directory than the usual data directory, we set the following line in Moodle configuration file `/var/www/moodle/config.php`:

```
$CFG->backuptempdir = '/var/www/moodledata/backup';
```

Setup of *X-Sendfile*

We set *X-Sendfile* in `/var/www/moodle/config.php` to allow files in the Moodle data directory to be uploaded faster directly through the web server:

```
$CFG->xsendfile = 'X-Accel-Redirect';
$CFG->xsendfilealiases = array ('/dataroot/' => $CFG->dataroot);
```

For *X-Sendfile* to be active, we already added the following lines to Nginx configuration file `/etc/nginx/sites-available/default`, inside `server` block (see page 12):

```
location /dataroot/ {
    internal;
    alias /var/www/moodledata/;
}
```

[9] See https://docs.moodle.org/400/en/Installing_Moodle#Command_line_installer.

Destinition of a custom filetype for the certificate SSL

Defining a custom file type in Moodle facilitates the SSL certificate from MoodleBox home page. The definition is hardcoded in `/var/www/moodle/config.php` too:

```
$CFG->customfiletypes = array(
  (object)array(
    'extension' => 'crt',
    'icon' => 'sourcecode',
    'type' => 'application/x-x509-ca-cert',
    'customdescription' => 'X.509 CA certificate'
  )
);
```

Hiding content that doesn't make sense for MoodleBox

We disable Moodle registration and hide Moodle campaign and Moodle services and support sections from notifications page. These content don't make sense for MoodleBox, which is offline most of the time.

```
$CFG->site_is_public = false;
$CFG->showcampaigncontent = false;
$CFG->showservicesandsupportcontent = false;
```

Final content of the file `/var/www/moodle/config.php` is then:

```
<?php  // Moodle configuration file

unset($CFG);
global $CFG;
$CFG = new stdClass();

$CFG->dbtype    = 'mariadb';
$CFG->dblibrary = 'native';
$CFG->dbhost    = 'localhost';
$CFG->dbname    = 'moodle';
$CFG->dbuser    = 'moodlebox';
$CFG->dbpass    = 'Moodlebox4$';
$CFG->prefix    = 'mdl_';
$CFG->dboptions = array (
  'dbpersist' => 0,
  'dbport' => '',
  'dbsocket' => '',
  'dbcollation' => 'utf8mb4_general_ci',
);

$CFG->wwwroot   = 'http://moodlebox.home';
$CFG->dataroot  = '/var/www/moodledata';
$CFG->admin     = 'admin';
```

```
$CFG->backuptempdir = '/var/www/moodledata/backup';
$CFG->xsendfile = 'X-Accel-Redirect';
$CFG->xsendfilealiases = array('/dataroot/' => $CFG->dataroot);
$CFG->customfiletypes = array(
  (object)array(
    'extension' => 'crt',
    'icon' => 'sourcecode',
    'type' => 'application/x-x509-ca-cert',
    'customdescription' => 'X.509 CA certificate'
  )
);
$CFG->site_is_public = false;
$CFG->showcampaigncontent = false;
$CFG->showservicesandsupportcontent = false;
$CFG->directorypermissions = 02777;

require_once(__DIR__ . '/lib/setup.php');

// There is no php closing tag in this file,
// it is intentional because it prevents trailing whitespace problems!
```

Cron configuration

Moodle cron should be launched every minute. Moreover, since MoodleBox is offline most of the time and doesn't have a mail server, no mail should be sent by cron process. We edit cron configuration with the shell command

```
$ sudo crontab -e
```

and add following lines to the crontab

```
MAILTO=""
* * * * * nice -n10 /usr/bin/php /var/www/moodle/admin/cli/cron.php
```

6 MoodleBox plugin

MoodleBox plugin[10] enables monitoring of the MoodleBox as well as managing several of its settings, such as password change, date and time setting, wireless access settings, etc.

This plugin works only on Raspberry Pi hardware, and needs some prerequisites to work

Installation of the MoodleBox plugin in Moodle

As usually, we install the plugin in Moodle by visiting *Site administration* > *Plugins* > *Repositories* > *Install plugins*. Click on the *Install plugins from the Moodle plugins directory* button and select *MoodleBox* administration plugin (*Admin tools*).

[10] MoodleBox plugin source code is available at https://github.com/moodlebox/moodle-tool_moodlebox.

It's also possible to install it via Git.

```
$ cd /var/www/moodle/admin/tool/
$ sudo git clone https://github.com/moodlebox/moodle-tool_moodlebox.git
    moodlebox
```

In this case, we need to complete the installation of the plugin by visiting the page http://moodlebox.home/admin.

Finalisation of the installation of the MoodleBox plugin

To finalise the installation of the MoodleBox plugin, we need to create of a few files and set their permissions. We'll also set the correct owner, group and permissions for all the plugin files.

```
$ cd /var/www/moodle/admin/tool/moodlebox
$ sudo touch .reboot-server .shutdown-server .set-server-datetime \
    .newpassword .wifisettings .resize-partition
$ sudo chown -R www-data:moodlebox /var/www/moodle/admin/tool/moodlebox
$ sudo chmod -R ug+w,o-w /var/www/moodle/admin/tool/moodlebox
$ sudo chmod -R 774 /var/www/moodle/admin/tool/moodlebox/bin
```

We install `direvent` and configure some jobs so that the MoodleBox plugin works correctly.

```
$ sudo apt-get install -y direvent
```

Here's the configuration file of `direvent`:

```
watcher {
  path /var/www/moodle/admin/tool/moodlebox/;
  file .reboot-server;
  event CLOSE_WRITE;
  command "/sbin/shutdown -r now";
}

watcher {
  path /var/www/moodle/admin/tool/moodlebox/;
  file .shutdown-server;
  event CLOSE_WRITE;
  command "/sbin/shutdown -h now";
}

watcher {
  path /var/www/moodle/admin/tool/moodlebox/;
  file .set-server-datetime;
  event CLOSE_WRITE;
  command "/bin/bash /var/www/moodle/admin/tool/moodlebox/.set-server-datetime
      ";
}

watcher {
  path /var/www/moodle/admin/tool/moodlebox/;
```

```
  file .newpassword;
  event CLOSE_WRITE;
  command "/bin/bash /var/www/moodle/admin/tool/moodlebox/bin/changepassword.
      sh";
}

watcher {
  path /var/www/moodle/admin/tool/moodlebox/;
  file .wifisettings;
  event CLOSE_WRITE;
  command "/usr/bin/python3 /var/www/moodle/admin/tool/moodlebox/bin/
      changewifisettings.py";
}

watcher {
  path /var/www/moodle/admin/tool/moodlebox/;
  file .resize-partition;
  event CLOSE_WRITE;
  command "/bin/bash /var/www/moodle/admin/tool/moodlebox/bin/resizepartition.
      sh";
}
```

Lastly, we copy these lines at the end of `/etc/sudoers.d/020_www-data-nopasswd` (or create it if it's not there):

```
www-data ALL=(ALL) NOPASSWD:/sbin/parted /dev/mmcblk0 unit MB print free
www-data ALL=(ALL) NOPASSWD:/usr/bin/vcgencmd
```

And we don't forget to reboot now!

7 Moodle repositories for USB sticks and SFTP uploads

Let's now configure the Moodle *File system* repositories, which will make it easier for MoodleBox's users to manage the files they want to provide with Moodle.

USB sticks

We set up automatic mounting of USB sticks (regardless of their format), as well as access to all files on inserted USB sticks via the *File system* repository.[11]

This is done by installing the `usbmount` package, then creating a directory in the Moodle data directory, and finally creating a link from the USB stick mount point to this directory.

The `usbmount` package should be built from its source code before to be installed.[12] We also install standard packages to support exFAT and NTFS formatted devices. Support for other formats is built-in.

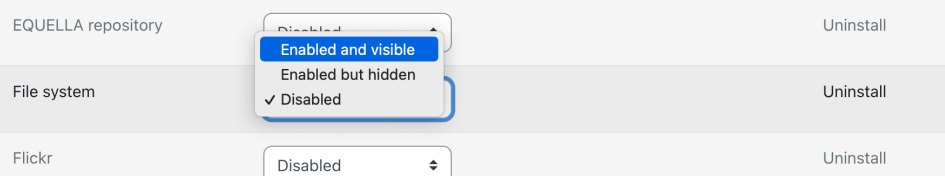
[11] If more than one USB stick is inserted into the MoodleBox, only the files on the first inserted stick are accessible via the File system repository.

[12] See https://github.com/rbrito/usbmount. Follow the instructions in the README file.

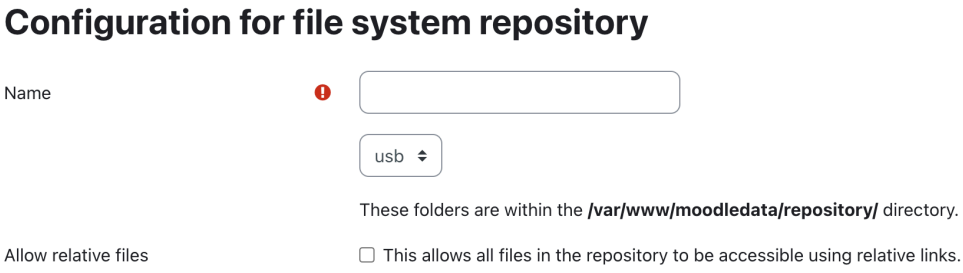
```
$ sudo apt-get install -y ntfs-3g exfat-fuse
$ sudo dpkg -i usbmount_0.0.24_all.deb
$ sudo mkdir -p /var/www/moodledata/repository
$ sudo chown -R www-data:moodlebox /var/www/moodledata/
$ sudo ln -s /media/usb /var/www/moodledata/repository
```

We then have to configure Moodle appropriately:[13] having logged in to Moodle as an administrator, visit *Site administration* > *Plugins* > *Repositories* > *Manage repositories*.

Select *Enabled and visible* in the row *File system* to enable this repository.



Click on *Save*. Then, on the same row, click on *Settings*, then on *Create a repository instance*. Finally select *usb* from the drop-down menu, and enter *USB Drive* in the mandatory *Name* field.



SFTP upload

We create a directory in which the files will be dropped in order to be accessible from Moodle, as well as a link to the Moodle data directory. Appropriate permissions and ownership are set on this directory, including SGID permission.

```
$ mkdir -p /home/moodlebox/files
$ sudo chown -R moodlebox:www-data files/
$ sudo chmod g+s files/
$ sudo ln -s /home/moodlebox/files /var/www/moodledata/repository
```

The repository is then configured in a similar way to the *USB Drive* repository above, by specifying the directory *files* and enter *SFTP Documents* as the repository name.

To use this feature, MoodleBox's admin will log in to the MoodleBox using a SFTP software[14], with the user name *moodlebox* and the password *Moodlebox4$*. He can then upload files in the `files` directory.

[13] See https://docs.moodle.org/en/File_system_repository.

[14] For instance: FileZilla, Cyberduck, WinSCP.

8 Additional Moodle configurations

Enabling access to the MoodleBox via the Moodle mobile app

After logging in to Moodle with the administrator account, we visit *Site Administration* > *Advanced features*. We check the box *Enable web services for mobile devices* and save the changes.

For MoodleBox, which is not intended to be published on the Internet, the warning[15] about the SSL certificate can be safely ignored.

Utilities to use PDF files in Moodle

We install Ghostscript and Poppler to enable Moodle to manage PDF annotations and PDF to PNG conversions.

```
$ sudo apt-get install -y ghostscript poppler-utils
```

9 Installation of *Adminer*

Adminer is a full-featured database management tool written in PHP. It consists of a single file ready to be deployed to the server. We install it by downloading it to the adequate location, and set its appropriate permissions.

```
$ cd /var/www/moodle/
$ sudo wget -c https://www.adminer.org/latest.php -O adminer.php
$ sudo chown moodlebox:www-data adminer.php
$ sudo chmod -R 664 adminer.php
```

To manage the database with *Adminer*, the interface should be accessed at http://moodlebox.home/adminer.php. To log in, use usual credentials: by default, username *moodlebox* and password *Moodlebox4$*.

10 Optimisation

To make the MoodleBox more comfortable to use, it is necessary to take care of its optimisation. We configure Moodle's cache, as well as its management of file uploads and downloads.

RAM disks for some Moodle directories[16]

Some files in Moodle data directory are needed very frequently. To enable faster access to them, we deliver them directly from the server's RAM, faster than the SD-card.

We create a directory as a mount point for the RAM disk, with adequate permissions. This directory will contain the cache files that we will define below.

[15] Text of the warning: *It is recommended to enable HTTPS with a valid certificate. The Moodle app will always try to use a secured connection first.*

[16] Inspired by https://www.leading-interactive.de/e-learning/moodle-performance-tuning-mit-tmpfs/.

```
$ sudo mkdir /var/cache/moodle
$ sudo chown www-data:moodlebox /var/cache/moodle/
$ sudo chmod ug+w,o-w /var/cache/moodle/
```

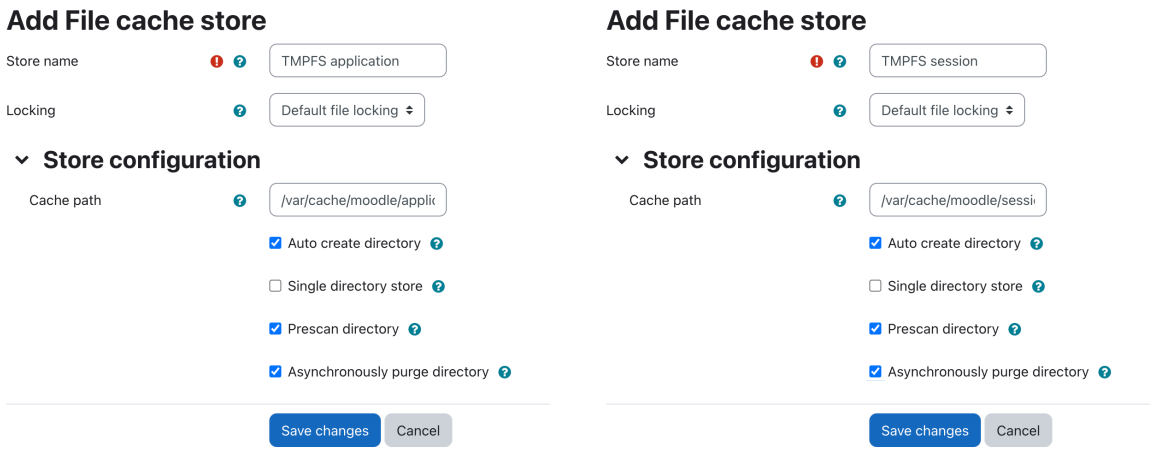
RAM disks are also used for the temporary and sessions directories in Moodle. These two directories are located Moodle data directory *moodledata*.

We define the RAM disks in the Raspberry's mount table. To do this, we add the following lines to the file `/etc/fstab`:

```
tmpfs /var/cache/moodle tmpfs size=64M,mode=775,uid=www-data,gid=www-data 0 0
tmpfs /var/www/moodledata/temp tmpfs size=64M,mode=775,uid=www-data,gid=www-
    data 0 0
tmpfs /var/www/moodledata/sessions tmpfs size=16M,mode=775,uid=www-data,gid=
    www-data 0 0
```

After a reboot of the MoodleBox, the cache can be configured in Moodle.

Log in to Moodle with the administrator account, then visit *Site Administration* > *Plugins* > *Caching* > *Configuration*. We create two new cache stores, by clicking on *Add Instance* in the *Installed cache stores* section (at the top of the page).



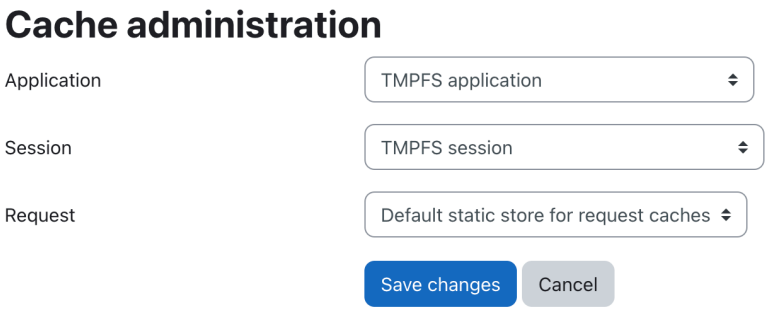
1. Store name: *TMPFS application*, Cache path: */var/cache/moodle/application*, check the boxes *Auto create directory*, *Prescan directory* and *Asynchronously purge directory*;

2. Store name: *TMPFS session*, Cache path: */var/cache/moodle/session*, check the boxes *Auto create directory*, *Prescan directory* and *Asynchronously purge directory*.

Finally, we map these new cache instances with their destination, by clicking on *Edit mappings* in the *Stores used when no mapping is present* section, at the very bottom of the page.

Putting the cache in a RAM disk has one big drawback: the data in it disappears every time the MoodleBox is restarted, and without proper handling Moodle must rebuild its cache each time. To keep the cache between reboots, we copy the contents of the RAM disk to the microSD card at regular intervals, and at each boot, we do the opposite.

Let's create the backup directory, then define the cron job:




```
$ sudo mkdir /var/cache/moodle-cache-backup/
$ sudo crontab -e
```

The following two lines are added to the cron table to back up the cache every 20 minutes and to restore the cache at startup:

```
*/20 * * * * rsync -a --delete /var/cache/moodle/ /var/cache/moodle-cache-
   backup/
@reboot cp -Rpf /var/cache/moodle-cache-backup/* /var/cache/moodle/
```

MariaDB optimisation

For a better performance, we tweak the value of some MariaDB variables in the file `/etc/mysql/mariadb.conf.d/50-server.cnf`:

```
skip-name-resolve
key_buffer_size         = 16M
max_allowed_packet      = 16M
query_cache_size        = 2M
log_error = /var/log/mysql/error.log
```

PHP optimisation

For a better performance, we tweak the value of a PHP variable in the file `/etc/php/7.4/fpm/pool.d/www.conf`:

```
pm.max_requests = 50
```

11 Setup of partition auto-resizing

In order to avoid the user having to manually resize the working partition, automatic resizing is configured at first boot. The method is the same as that used when building the *Raspberry Pi OS Lite* disk image.[17]

[17] See https://github.com/RPi-Distro/pi-gen.

We copy the file `resize2fs_once` below into the `/etc/init.d/` directory, and give it the appropriate permissions to be launched on reboot.

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          resize2fs_once
# Required-Start:
# Required-Stop:
# Default-Start: 3
# Default-Stop:
# Short-Description: Resize the root filesystem to fill partition
# Description:
### END INIT INFO
. /lib/lsb/init-functions
case "$1" in
  start)
    log_daemon_msg "Starting resize2fs_once"
    ROOT_DEV=$(findmnt / -o source -n) &&
    resize2fs $ROOT_DEV &&
    update-rc.d resize2fs_once remove &&
    rm /etc/init.d/resize2fs_once &&
    log_end_msg $?
    ;;
  *)
    echo "Usage: $0 start" >&2
    exit 3
    ;;
esac
```

Finally, before the last shutdown before cloning and resizing the disk image, we finish by running the command

```
$ sudo systemctl enable resize2fs_once
```

then we add at the end of the unique line of the file `/boot/cmdline.txt` the instruction below, immediately after the text `rootwait` (without forgetting a space).

```
quiet init=/usr/lib/raspi-config/init_resize.sh
```

It is essential that the MoodleBox is not restarted, otherwise the entire operation described in this section will have to be repeated.

12 Cleanup

The commands below will clean up the MoodleBox and reduce the amount of disk space it requires, before cloning and distributing it.

```
$ sudo systemctl stop dnsmasq
$ sudo truncate -s 0 /var/lib/misc/dnsmasq.leases
$ sudo rm -rf /var/www/moodledata/cache/*
```

```
$ sudo rm -rf /var/www/moodledata/localcache/*
$ sudo rm -rf /var/www/moodledata/temp/*
$ sudo rm -rf /var/www/moodledata/trashdir/*
$ sudo rm -rf /var/www/moodledata/sessions/*
$ sudo rm -rf /var/cache/moodle/*
$ sudo rm -rf /var/cache/moodle-cache-backup/*
$ sudo rm -rf /var/backups/
$ sudo mysql moodle -e "truncate table moodle.mdl_logstore_standard_log"
$ sudo mysql moodle -e "truncate table moodle.mdl_config_log"
$ sudo mysql moodle -e "truncate table moodle.mdl_upgrade_log"
$ sudo mysql moodle -e "truncate table moodle.mdl_task_log"
$ sudo apt-get clean
$ sudo rm -rf /var/cache/debconf/*
$ sudo rm -rf /var/lib/apt/lists/*
$ sudo rm -rf /tmp/*
$ sudo rm -rf /var/tmp/*
$ sudo rm -f ~/.mysql_history ~/.nano_history ~/.bash_history
$ sudo apt-get --purge autoremove
$ sudo truncate -s 0 /root/.bash_history
```

13 Acknowledgements

MoodleBox uses some of the great ideas from the first proof of concept by Christian Westphal[18], for which he deserves special thanks.

[18] Christian Westphal, Académie de Strasbourg, see https://moodle.org/user/view.php?id=1378197&course=20.