MoodleBox building reference
Version 4.8.0-dev*

Nicolas Martignoni†

August 2, 2024

Contents

†MoodleBox founder and maintainer; teacher in college level mathematics and computer science and teacher trainer in Fribourg, Switzerland.

1 Introduction

This document explains how to build a MoodleBox manually. It is intended for developers or system administrators and provide background information on how a MoodleBox is built. It is not meant to be an end-user documentation.

For the MoodleBox end-user documentation, consult the MoodleBox website.

Caveat

The actual build of the MoodleBox image is done automatically using Ansible.[1] 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, with optional captive portal.
- 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[2] providing a GUI to manage most aspects of the MoodleBox.

Specific Moodle features

- Moodle version 4.3.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.[3] The *cron* service is launched every minute.
- When a USB sticks are inserted into the MoodleBox, their files are available to users in Moodle's *USB Drives* repository.
- Ability to upload files via SFTP directly to the MoodleBox; these files are then available to users in Moodle's *SFTP Documents* 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] The MoodleBox project is a free (as in speech and in beer) and open source project. The Ansible playbook source to build the MoodleBox image is provided under a AGPL v3.0 License. It is available at GitHub.

[2] https://github.com/moodlebox/moodle-tool_moodlebox.

[3] https://download.moodle.org/mobile/.

Acknowledgements

MoodleBox uses some of the great ideas from its first proof of concept by Christian Westphal[4], for which he deserves special thanks.

2 Preparation of the Raspberry Pi

Although the MoodleBox works fine on other Raspberry Pi models, a Raspberry Pi Model 3B, 3B+, 4B or 5 is recommended to **build** it.

Copy of Raspberry Pi OS Lite on a microSD card

Download the latest version of *Raspberry Pi OS Lite (64-bit)* from the Raspberry Pi web site,[5] and copy it on a (good quality!) microSD card with at least 8 GB capacity. For this process, it is recommended to use Raspberry Pi Imager. A complete description of the process is available on Raspberry Pi web site.[6]

Insert then the microSD 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 partition of the SD card which can be seen when it is mounted in a macOS or Windows computer. (Other partitions aren't visible on these OS.) The content of the file `ssh.txt` doesn't matter, so the file can be empty. The following commands will do this.

```
$ cd <mounting point of "bootfs" 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.

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.[7]

In the MoodleBox image, the default user account has the username *moodlebox* and the password *Moodlebox4$*.

[4] Christian Westphal, Académie de Strasbourg, see https://moodle.org/user/view.php?id=1378197&course=20.
[5] See https://www.raspberrypi.com/software/operating-systems.
[6] See https://www.raspberrypi.org/documentation/computers/getting-started.html.
[7] See https://www.raspberrypi.com/documentation/computers/configuration.html#configuring-a-user.

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`.[8]

From now on, all operations are done via the command line in a regular terminal on macOS and Linux, or with `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 update
$ sudo apt full-upgrade -y
```

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

Fix default user home directory permissions

We must fix the default user home directory permissions, otherwise certain features will not be available.

```
$ sudo chmod u=rwx,g=rx,o=rx /home/moodlebox/
```

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*:
  - `en_GB.UTF-8` (should be already here)

[8] 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.

  - en_AU.UTF-8
  - fr_FR.UTF-8
  - de_DE.UTF-8
  - es_ES.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*.

If hostname is changed, MoodleBox won't work correctly. To prevent this issue, we check the hostname at first boot and fix it if needed. This is done by copying the following file to /etc/init.d/fix_hostname_once, with adequate permissions.

```
#! /bin/bash
#
# MoodleBox init script
# Created for MoodleBox
#
# This script is part of MoodleBox
# Copyright (C) 2022 onwards Nicolas Martignoni
#
# This script is free software: you can redistribute it and/or modify
# it under the terms of the GNU Affero General Public License as published
# by the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This script  is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU Affero General Public License
# along with this script.  If not, see <https://www.gnu.org/licenses/>.

### BEGIN INIT INFO
# Provides: fix_hostname_once
# Required-Start:
# Required-Stop:
# Default-Start: 3
# Default-Stop:
# Short-Description: Starts fix_hostname_once to fix hostname
# Description:
### END INIT INFO

. /lib/lsb/init-functions

SCRIPTPATH="$(cd -- "$(dirname "$0")" >/dev/null 2>&1 ; pwd -P)"
SCRIPTNAME="$(basename $0)"
```

```
case "$1" in
  start)
    log_daemon_msg "Fixing hostname" &&
    NEW_HOSTNAME=$(hostnamectl --static) &&
    CURRENT_HOSTNAME="moodlebox" &&
    sed -ri "s/(((25[0-5]|(2[0-4]|1[0-9]|[1-9]|)[0-9])\.?\b){4})\s+${
        CURRENT_HOSTNAME}/\1\t${NEW_HOSTNAME}/g" /etc/hosts &&
    update-rc.d ${SCRIPTNAME} remove &&
    rm -f ${SCRIPTPATH}/${SCRIPTNAME} &&
    log_end_msg $?
    ;;
  *)
    echo "Usage: $0 start" >&2
    exit 3
    ;;
esac
```

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 5.

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/firmware/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/firmware/config.txt`, after `# Additional overlays...`, to get:

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

Limit journald file size to 16 MB

The `journald` log file can become very large. We limit its size to 16 MB by creating the file `/etc/systemd/journald.conf.d/journald_moodlebox.conf`, with the following content:

```
[Journal]
SystemMaxUse=16M
```

3 Wireless access point feature

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

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.

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 available that has been tuned to maximise the number of clients in access point mode (AP) while still supporting client mode (STA).

We use this alternative firmware in the MoodleBox. To switch the firmware to the alternative one, we launch the following command and select `cyfmac43455-sdio-minimal.bin` firmware.

```
$ sudo update-alternatives --config cyfmac43455-sdio.bin
```

Setup of the wireless regulatory country

We have to set the wireless regulatory country. MoodleBox default setting for the regulatory country is `CH`. Two steps are required to do this. We first set it via the `iw` utility:

```
$ sudo iw reg set CH
```

Then we set it in the kernel command line, by adding `cfg80211.ieee80211_regdom=CH` at the end of the unique line of the file `/boot/firmware/cmdline.txt`.

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 named `uap0`, 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 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.

We edit now the file `/etc/hosts`, replacing the last line, beginning with `127.0.1.1`, with the following, containing the static IP address. For the MoodleBox, the static IP address chosen is `10.0.0.1`. Any other private IP address can alternatively be used.

```
10.0.0.1    moodlebox
```

Configuration of the access point

We create the wifi access point connection using NetworkManager. Note we use the interface `uap0` and the static IP address we just defined. Other settings such as SSID, password and wireless channel can be changed.

```
$ sudo nmcli c add type wifi ifname uap0 con-name WifiAP ssid MoodleBox
$ sudo nmcli c mod WifiAP autoconnect yes 802-11-wireless.mode ap \
    802-11-wireless.band bg 802-11-wireless.channel 11
$ sudo nmcli c mod WifiAP ipv4.address 10.0.0.1/24 ipv4.gateway 10.0.0.1 \
    ipv4.method shared
$ sudo nmcli c mod WifiAP wifi-sec.key-mgmt wpa-psk wifi-sec.psk moodlebox
$ sudo nmcli c mod WifiAP wifi-sec.proto rsn wifi-sec.group ccmp \
    wifi-sec.pairwise ccmp
```

We change some settings of the AP, in particular the DHCP settings, by adding a configuration file to `dnsmasq` instance managed by NetworkManager. The file has to be placed in `/etc/NetworkManager/dnsmasq-shared.d/` Here's the content of MoodleBox setting file `/etc/NetworkManager/dnsmasq-shared.d/00-dhcp.conf`:

```
server=9.9.9.9
server=149.112.112.112
domain-needed
bogus-priv
domain=home
local=/home/
address=/home/10.0.0.1
expand-hosts
dhcp-option=6,10.0.0.1
cache-size=100
log-facility=/var/log/dnsmasq.log
```

Configuration of the wireless connection as a client

To be used as a client (STA mode) as well as an access point, a valid configuration should be provided to NetworkManager, with SSID and the password of an existent wireless network with

which to connect. We also set the routing metric of the connection adequately. Note we use the interface `wlan0`.

```
$ sudo nmcli c add type wifi ifname wlan0 con-name WifiSTA ssid WirelessLAN
$ sudo nmcli c mod WifiSTA wifi-sec.key-mgmt wpa-psk wifi-sec.psk Password
$ sudo nmcli c mod WifiSTA ipv4.route-metric 99
```

Configuration of mDNS services advertising

In order to make MoodleBox services discoverable 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>
```

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.[9] 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:

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

```
$ sudo dpkg -i nodogsplash_5.0.2-1_arm64.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
    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.[10] 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.

[10] 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.

```
$ sudo apt-get install -y nginx php8.2-fpm php8.2-mbstring \
    php8.2-curl php8.2-gd php8.2-intl php8.2-soap php8.2-mysql \
    php8.2-xml php8.2-zip php8.2-apcu php8.2-tidy php8.2 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.

```
# MoodleBox default web server configuration
#
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    # Uncomment the following 4 lines to enable HTTPS on the MoodleBox
    # 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/php8.2-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 variables 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/php8.2-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/8.2/fpm/pool.d/www.conf` to set the correct group ownership for PHP, replacing line `group = www-data` with `group = 1000`, so we have now

```
...
user = www-data
group = 1000
...
```

We set also the PHP time zone, so that OS time zone and PHP time zone are the same. For consistency, this is done for command line and fpm declination of PHP. We set the value of the variable `date.timezone` to `Europe/Zurich` in both of the files `/etc/php/8.2/fpm/php.ini` and `/etc/php/8.2/cli/php.ini`:

```
date.timezone = Europe/Zurich
```

If needed, variables in `/etc/php/8.2/fpm/pool.d/www.conf` can be tweaked, e.g. for performance gains. See page 27 for examples.

We then relaunch the web server:

```
$ sudo systemctl restart nginx php8.2-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 certificate is not used by default. If HTTPS is not required, this can be left out.

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 4).

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

In order to allow flexible access to all the databases, a new database user with all privileges is created in MariaDB. For this installation, we set the credentials as earlier (see page 5), for consistency.

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

We also set the database default collation, according to Moodle recommendation.[11], by setting the variable `collation-server` to `utf8mb4_unicode_ci` in the file `/etc/mysql/mariadb.conf.d/50-server.cnf`:

```
collation-server = utf8mb4_unicode_ci
```

If needed, other variables in `/etc/mysql/mariadb.conf.d/50-server.cnf` can be tweaked, e.g. for performance gains. See page 26 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.

[11] See https://docs.moodle.org/en/MySQL_full_unicode_support

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_403_STABLE \
    https://github.com/moodle/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
```

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/
$ sudo chmod -R ug+w,o-w /var/www/moodle /var/www/moodledata/
$ 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 (recommended).[12] During the installation, we set the administrator account with the usual credentials.

If using the command line interface, this steps takes about 10 minutes.

When Moodle installation is complete, we 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
```

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

Moodle configuration

Moodle is now installed and works normally. We have to configure it specifically 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 13):

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

Definition of a custom filetype for the certificate SSL

Defining a custom file type in Moodle makes it easier to download the SSL certificate from MoodleBox home page. The definition is hardcoded in `/var/www/moodle/config.php`:

```
$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;
```

Summary of tweaks in Moodle configuration file

Here's the final content of the file `/var/www/moodle/config.php`:

```
<?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[13] provides 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 correctly.

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.

It's also possible to install it via Git, e.g.

```
$ sudo -u www-data -g moodlebox git clone --depth=1 \
    https://github.com/moodlebox/moodle-tool_moodlebox.git \
    /var/www/moodle/admin/tool/moodlebox
```

Don't forget in this case to checkout a stable branch, and to complete the installation of the plugin by visiting http://moodlebox.home/admin.

Finalisation of the installation of the MoodleBox plugin

To complete 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.

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

```
$ 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
```

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";
}
```

We also need to regularly copy the DHCP leases file to a location accessible for PHP, so we configure a cron task.

```
$ sudo crontab -e
```

and add following line to the crontab

```
* * * * * install -C -m 644 /var/lib/NetworkManager/dnsmasq-uap0.leases /tmp/
    dnsmasq.leases
```

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
```

Rebooting is necessary after these steps!

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 Moodle's *File system* repositories.

This is done by installing the `udevil` package. We also install standard packages to support exFAT, HFS+ and NTFS formatted devices. Support for other formats is built-in.

```
$ sudo apt-get install -y udevil ntfs-3g exfat-fuse
```

By adding the formats `exfat` and `hfsplus` to the line beginning with `allowed_types` of the configuration file of udevil `/etc/udevil/udevil.conf`, support for USB sticks of these formats is activated. This line should now read

```
allowed_types = $KNOWN_FILESYSTEMS, file, cifs, smbfs, nfs, curlftpfs, ftpfs,
    sshfs, davfs, tmpfs, ramfs, exfat, hfsplus
```

We remove the nonempty option from the line beginning with `default_options_exfat` of the same file. This line should now read

```
default_options_exfat = nosuid, noexec, nodev, noatime, uid=$UID, gid=$GID,
    iocharset=utf8
```

Finally, we add a script fixing the permissions on the mounted USB devices, by completing the line beginning with `success_rootexec` this way:

```
success_rootexec = /usr/local/bin/fix-mount-permissions.sh
```

We copy the script file `fix-mount-permissions.sh` to `/usr/local/bin/`. Here's the content of this file:

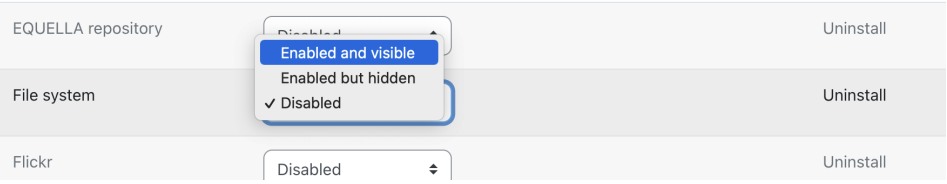
```
#! /bin/bash
# This script MUST be run as root.
[[ $EUID -ne 0 ]] && { echo "This script must be run as root"; exit 1; }

USERNAME=$1

/usr/bin/chmod 775 /media/${USERNAME}/

exit 0
```

We create then a directory with adequate permissions, and a link from the USB stick mount point to this directory:

```
$ 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:[14] 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.

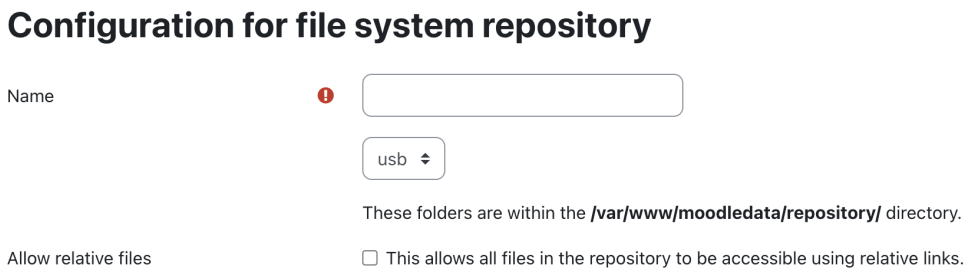


Click on *Save*. Then, on the same row, click on *Settings*, then on *Create a repository instance*. Finally select *usb* from the drop-down menu, and enter *USB Drives* in the mandatory *Name* field.

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.

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




```
$ 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.

If the main username of the MoodleBox is changed, the soft link will be broken. To prevent this issue, we check the link at login time and fix it if needed, by adding the following lines to file `/home/moodlebox/.profile`.

```
# Fix soft link to repository if broken
SOFTLINK="/var/www/moodledata/repository/files"
if [ ! -e ${SOFTLINK} ] ; then
    rm -f ${SOFTLINK}
    ln -s /home/$(getent passwd 1000 | cut -d: -f1)/files ${SOFTLINK}
fi
```

To use the SFTP upload feature, MoodleBox's admin will log in to the MoodleBox using a SFTP software[15], with their usual credentials. They can then upload files in the `files` directory.

8 Additional Moodle configurations

Enable MoodleBox access 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[16] about the SSL certificate can be safely ignored.

Install MathJax locally

By default, Moodle loads the MathJax library from the Internet. We want that this library is available locally, since MoodleBox is offline most of the time.

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

[16] 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.*

We install the MathJax library with Git at the correct location; we get the version supported by Moodle, in a shallow clone to save space, and we set the adequate permissions and ownership.

```
$ sudo git clone --depth=1 -b 2.7.9 https://github.com/mathjax/MathJax.git /
    var/www/moodle/lib/MathJax
$ sudo chown -R www-data:moodlebox /var/www/moodle/lib/MathJax
$ sudo chmod -R ug+w,o-w /var/www/moodle/lib/MathJax
```

We can now update the MathJax library URL in *Site administration* > *Plugins* > *Filters* > *MathJax*, setting it to `/lib/MathJax/MathJax.js`.

Install utilities for PDF files manipulation 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
```

Set Moodle system paths

We configure the Moodle system paths relevant for the MoodleBox, by visiting *Site administration* > *Server* > *System paths*, and we set there the following paths:

- Path to PHP CLI: `/usr/bin/php`
- Path to du: `/usr/bin/du`
- Path to ghostscript: `/usr/bin/gs`
- Path to pdftoppm: `/usr/bin/pdftoppm`
- Path to Python: `/usr/bin/python`

Download H5P libraries

To be able to use H5P out of the box without needing to connect to Internet, we download all the H5P libraries within Moodle user interface, by visiting *Site administration* > *General* > *H5P* > *H5P overview* and click *Run now* under *H5P scheduled task*.

The download time is rather long (around 10 minutes) depending on your Internet connection.

Make sure to disable Moodle's debug messages display

We then disable Moodle debug messages display, by visiting *Site administration* > *Development* > *Debugging*, making sure that the checkbox *Display debug messages* is unchecked.

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.

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

To manage the database with *Adminer*, the interface should be accessed at http://moodlebox.home/adminer.php. To log in, use the credentials set at the MariaDB installation, section 4.

10 Installation of `rpi-clone`

The `rpi-clone` utility enables to make copies of the currently running MoodleBox image to an external device. 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.

```
$ wget -c https://raw.githubusercontent.com/geerlingguy/rpi-clone/master/rpi-
    clone -O /usr/local/sbin/rpi-clone
$ sudo chown root:root /usr/local/sbin/rpi-clone
$ sudo chmod -R 775 /usr/local/sbin/rpi-clone
```

Usage of `rpi-clone` is not covered in this document.

11 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.

Setup RAM disks for specific Moodle directories[17]

Some files in Moodle data directory are needed very frequently and quickly, e.g. temporary files and session data. To enable faster access to them, we deliver them directly from the server's RAM, faster than the microSD card, using RAM disks.

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 Pi's mount table. To do this, we add the following lines to the file `/etc/fstab`:

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

[17] Partly inspired by https://www.leading-interactive.de/e-learning/moodle-performance-tuning-mit-tmpfs/.

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

Install Redis for caching

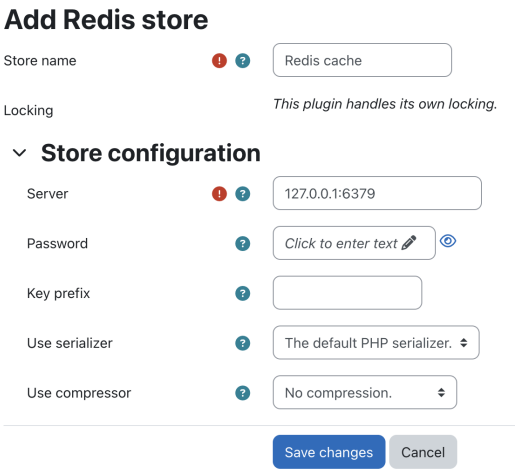
We use Redis to speed up Moodle caching. Let's install Redis and its PHP module.

```
$ sudo apt-get install -y redis php-redis
```

Moodle cache configuration

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

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



Store name: *Redis cache*, Server: *127.0.0.1:6379*, and we save the changes.

Finally, we map the Application and Session stores to this new cache instance, by clicking on *Edit mappings* in the *Stores used when no mapping is present* section, at the very bottom of the page. We map there Application and Session to *Redis cache*.

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
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/8.2/fpm/pool.d/www.conf`:

```
pm.max_requests = 50
```

12 Setup of partition auto-resizing

This setting is only useful if you intend to distribute a disk image that you'll minimise. It can be skipped otherwise.

In order to avoid the user having to manually resize the working partition, automatic resizing at first boot can be configured. The method used is the same than when building the *Raspberry Pi OS Lite* disk image.[18]

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 minimising the disk image, we finish by running the command

```
$ sudo systemctl enable resize2fs_once
```

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

then we add at the end of the unique line of the file `/boot/firmware/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.

13 Cleanup

The commands below will clean up the MoodleBox and reduce the amount of disk space it requires, if you intend distributing it as a disk image.

```
$ 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/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
```