Pi-Factory Documentation

Release 0.5.0

LNCM

Nov 18, 2019

Contents:

CHAPTER 1

Pi-Factory

1.1 *Easily build your own Alpine Linux box on Raspberry PI or Raspberry Zero!*

This repository lets you build you build an *Alpine Linux* box for Raspberry Pi Model B/B+/4/Zero

Documentation Status

1.2 Features

- pi-factory is now a persistent OS Only! For Nodes, please refer to earlier versions or go to the noma project
- Latest version updated to Alpine 3.10.2
- We now have 64 bit (aarch64) and 32 bit images (armhf). The Raspberry PI 3/3b works well on 64 bit
- Deterministic builds. Images now built and released automatically through github actions.

1.3 Hardware Requirements

- **Raspberry Pi**
  - Recommended: model 3B+
  - Optional: case, heatsink, LAN cable, HDMI cable and monitor (for troubleshooting and issue tracking), Keyboard (to use for troubleshooting and issue tracking)
- **microSD** card
  - Recommended: SanDisk 16GB or more
  - microSD card to USB adapter or built-in hardware

Quality goes over quantity here!

- **Power Supply** (5V/2.5A) with micro-USB cable
  - Recommended: official Raspberry Pi power supply
  - Alternatively:
    * high-quality USB charger (e.g. Samsung)
    * short USB to micro-USB cable. (a longer cable can work with 5.1V chargers)

Warning! Your Raspberry Pi will *not work* properly without a correctly rated power supply and cable, and may result in data loss. We are not responsible if you lose any data.

1.4 Instructions

1. Download Etcher
2. Download the image from the Releases page or simply clone the `https://github.com/lncm/pi-factory.git` repository on github
3. Insert SD Card and open up Etcher
4. Etch one of the images onto the SD card
5. Remount the SD card and create a file called `wpa_supplicant.conf`
6. Inside the file put the following

```
network={
        ssid="Your Wifi SSID goes here"
        key_mgmt=WPA-PSK
        psk="YOUR Password goes here"
}
```

1. Unmount the drive and put it into a PI or PI zero and then start it up. And in about 10-20 minutes (PI-Zero will take longer), you will be able to login through avahi/mdns `box.local` , or if you don't have avahi/mdns on your desktop you will need to grab the IP address from plugging in your PI to a TV or from your router.

1.5 Access

1.5.1 Users & Passwords

- lncm
  - **username**: lncm
  - **password**: chiangmai
- root
  - **username**: root
  - **password**: chiangmai

Note: `sudo` is not installed, use `su` instead. We also highly recommend that you change the password.

1.5.2 Command-line via ssh

```
ssh lncm@box.local
```

Note: First boot will take some time as ssh host keys are generated.

1.5.2.1 Advanced configuration

When building the image yourself you can create a `wpa_supplicant.automatic.conf` file with all your wifi passwords.

You may disable several stuff by placing an empty file inside the FAT partition. This should be done **before first boot**

- *noswap* : disables SWAP generation (not recommended unless you know what you are doing!)
- *noavahi* : disables install for avahi-daemon / mdns discovery (not recommended unless you know what you are doing!)
- *nodocker* : disables Docker installation
- *nopython* : Disables python3 installation
- *notor* : Disables tor installation

1.6 Documentation

Read the Docs

1.7 Building

To generate a fresh image from source run `./make_img.sh` as **root** on a *Debian*, *Ubuntu* or *Alpine* system.

For convenience, we also support `vagrant` to automate setting up your development VM.

1.7.1 MacOS instructions:

Install dependencies (homebrew, virtualbox, vagrant):

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/
↪master/install)"
brew cask install virtualbox
brew cask install vagrant
git clone pi-factory
cd pi-factory
```

Create VM and generate image:

- `vagrant up`

Rebuild image without wiping VM:

- `vagrant up --provision`

Also useful:

- `vagrant ssh`

- `vagrant halt`
- `vagrant destroy -f`
- `brew cask install vagrant-manager` (optional menu-bar utility)

1.8 Support

If you are having problems, please create an issue

1.9 Contribute

Bug reports, pull-requests and suggestions are very welcome!

- Issue Tracker
- Source Code
- Contributing Guidelines

1.10 License

The project is licensed under the permissive Apache 2.0 license.

CHAPTER 2

Welcome to Pi-Factory's documentation!

Pi-Factory builds Alpine Linux images with **aarch64** and **armhf** architecture ready for booting on any Raspberry Pi.

2.1 Customization & Settings

These options apply to creating your own images for burning to microSD card.

Before running, **make_img.sh** or **make_apkovl.sh** you may wish to etch your wifi settings into the box so it will complete its setup smoothly. To do this, create a file called `wpa_supplicant.automatic.conf` and place the following in there:

```
network={
        ssid="Your Wifi SSID goes here"
        key_mgmt=WPA-PSK
        psk="YOUR Password goes here"
}
```

2.1.1 Security

If you wish to disable passwords altogether (highly recommended, especially vs the default password), simply place a file called `authorized_keys.automatic` into the root of this repository with all your public keys.

Passwords should be disabled when you create a new image.

For those who still are using password authentication it is recommended that you change both root and lncm users with the `passwd` utility.

2.1.2 Networking

If you have console access:

As **root** use `wpa_passphrase` tool to set wifi settings

```
wpa_passphrase "WiFi Name" "Password" >> /etc/wpa_supplicant/wpa_supplicant.
conf
```

Or, run `setup-interfaces` if you have access to a running box.

In order to ship correct WiFi configuration, edit settings in `etc/wpa_supplicant/wpa_supplicant.conf`, run `make_apkovl.sh` and copy **box.apkovl.tar.gz** to SD card root directory (FAT partition).

Alternatively you may copy wpa_supplicant.conf to the FAT partition of the box (can be done at any time). The box will boot up and copy this file into the correct place and OVERWRITE any changes.

2.1.3 Alpine specific

Alpine wiki holds further information related to system administration.

2.1.3.1 Committing changes to SD card

Initially the system is mounted read-only!

Important note: Alpine will not persist user changes upon reboot until it is installed and restarted.

Use `lbu commit` to persist changes. Add `-v` to see what is being committed.

`lbu status` will show changes to be committed.

Note: By default `lbu commit` only applies to *some* directories.

After setup is fully complete, the system will be a full persistant system

2.1.3.2 Package management

- `apk update` Update repositories
- `apk upgrade` Upgrade packages
- `apk add` Install package
- `apk del` Uninstall package

2.1.3.3 Init system

- `rc-update` show startup services

Installation of LNCM specific components belongs in `etc/init.d/lncm`. The script is OpenRC compatible and must be executable, without a file name extension.

`etc/apk/world` contains all apk packages to be installed by LNCM's install script.

- `service -l` list available services

The boot sequence is logged to `/var/log/rc.log` by default.

More information in OpenRC user guide

2.1.3.4 Misc

There are various configuration tools included to help you customize to your needs:

- `setup-hostname` (change the hostname)
- `setup-timezone` (change the timezone)
- `setup-keymap`
- `setup-dns` (change the DNS)
- `setup-alpine` (Go through ALL the setup scripts. Useful if you choose to setup networking manually)

2.2 Advanced

2.2.1 Using nmap

Raspberry Pi's can be easily intentified when on the same subnet by their distinct MAC address.

Using `nmap` you can find your Raspberry Pi like so,

```
sudo nmap -v -sn 192.168.0.0/24 | grep -B 2 "Raspberry Pi Foundation"
```

2.2.2 Connecting to console via serial cable

(serial TTY via TTL on uart)

Connect cable to *GND*, *RX*, *TX* pins, make sure you are using 3.3V and **not** 5V to prevent damage! With some devices RX & TX may have to be crossed.

Add `enable_uart=1` to `config.txt` on SD card FAT partition. (may not be necessary on older models)

e.g. `screen /dev/tty.usbserial-XYZ 115200`

2.2.3 Automated builds

Use `make_img.sh` to create latest lncm-box.img

2.2.4 Auditing

Follow the steps outlined in `make_img.sh` to create your own image or SD card.

2.2.5 Re-creating apkovl.tar.gz from source

```
make_apkovl.sh
```

2.2.6 Unpacking apkovl from lncm-box.tar.gz

```
tar xzf box.apkovl.tar.gz
```

2.2.7 Creating new apkovl

`lbu pkg /path/to/tar.gz` will produce a tarball of current system state.

2.2.8 *Important notes for distributing fresh apkovl:*

Remove unique and security sensitive files

```
rm etc/machine-id
rm etc/docker/key.json
rm etc/ssh/ssh_host_*
```

Rewrite `/etc/resolv.conf` to be network independent.

Be mindful of passwords you set.