

Building Homebridge with the Yocto Project

Leon Anavi

Konsulko Group

leon.anavi@konsulko.com

leon@anavi.org

FOSDEM 2020

Agenda

- Homebridge
- Yocto Project and OpenEmbedded
- Building a distribution with Homebridge using Yocto/OE
- Exploring distribution features
- Ideas for improvements
- Conclusions

Why? The Story ...

- A friend, electrical engineer, asked me for help with a very expensive proprietary solution for smart home that didn't support Apple HomeKit and Siri
- Setting up Raspbian, Homebridge and all dependencies is annoying and time consuming, especially for non-Linux users
- For a long time I wanted to make a custom Linux distro as a complementary hub for various servers to my open source home automation setup
- Although I have numerous years of professional experience with the Yocto Project, I was curious to see if it is useful for makers

Homebridge



- Lightweight server that emulates Apple iOS HomeKit API
- Written in Node.js
- Numerous plugins exist for integrating various devices
- Can be installed on macOS, MS Windows 10, GNU/Linux distributions and Docker
- Available at GitHub under Apache License 2.0:
<https://github.com/nfarina/homebridge>
- <https://homebridge.io/>

Homebridge Community



- Started by Nick Farina in 2014
- Thousands of plugin developers
- Dozens of core contributors
- Based on the work of Alex Skalozub (@pieceofsummer) who reverse engineered HomeKit and Khaos Tian (@KhaosT) who built the HAP-NodeJS, implementation of the HomeKit Accessory Server

Homebridge Plugins



Homebridge plugin allow integration of various Internet of Things. Popular plugins are:

- Config-UI-X (web interface)
- Legrand (BTicino) MyHome
- Sonoff (for Sonoff Basic devices with Tasmota firmware)
- Alexa (exposes homebridge controlled devices to Amazon Alexa)
- IKEA Trådfri Gateway
- MQTT
- Many other plugins...

Let's Build an Embedded Linux Distro! How?

- Yocto Project
- Buildroot
- PTXdist
- OpenWRT
- Other ... including customizing a Debian derivative

Chris Simmonds at Embedded Linux Conference EU 2019: Debian or Yocto Project? Which is the Best for your Embedded Linux Project?

What to Include in Our Distro?

- BSP for optimal performance (64-bit where possible):
bootloader, Linux kernel and device drivers
- Init system: Systemd
- Connectivity and interfacing options: WiFi, SSH, VNC, serial
- Node.js and NPM
- Homebridge with plugins
- Mosquitto MQTT broker
- X11 windowing system with openbox, pcmanfm, xterm, gedit,
network manager, surf (minimalist web browser)
- Support low-cost mini OLED display for showing system status

Yocto Project & OpenEmbedded

- Open source collaborative project of the Linux foundation for creating custom Linux-based systems for embedded device using the OpenEmbedded Build System
- OpenEmbedded Build System includes BitBake and OpenEmbedded Core
- Poky is a reference distribution of the Yocto Project provided as metadata, without binary files, to bootstrap your own distribution for Internet of Things and embedded devices
- Bi-annual release cycle

Yocto Project Releases

Codename	Version	Release Date	Support Level
Gatesgarth	3.2	Oct 2020	Dreaming
Dunfell	3.1	April 2020	Under development
Zeus	3.0	October 2019	Stable
Warrior	2.7	April 2019	Stable
Thud	2.6	Nov 2018	Stable
Sumo	2.5	April 2018	Community
Rocko	2.4	Oct 2017	Community

- For details: <https://wiki.yoctoproject.org/wiki/Releases>

Building an Image

- Checkout the source code with **Repo**:

```
mkdir anavi-hub && cd anavi-hub
repo init -u ssh://git@github.com/AnaviTechnology/anavi-
hub.git
repo sync
```

- Set up build environment (by default for Raspberry Pi 4, edit **local.conf** to change the machine):

```
TEMPLATECONF=../meta-homebridge/conf/ source poky/oe-
init-build-env
```

- Build an image:




```
bitbake core-image-homebridge
```

Alternatively, Just Download an Image

- Binary images for the supported hardware platforms (as of the moment several Raspberry Pi versions) are available as assets at GitHub with each release:

<https://github.com/AnaviTechnology/anavi-hub/releases>

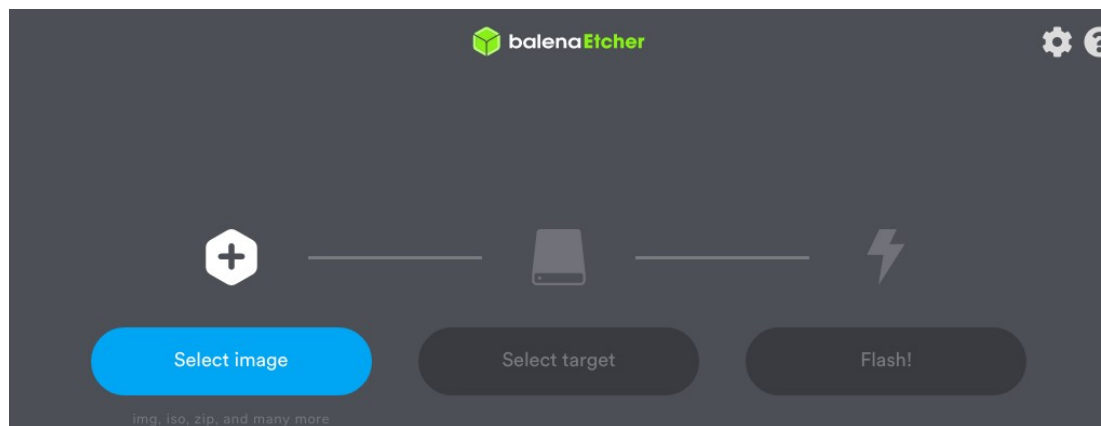
▼ Assets 3

 core-image-homebridge-raspberrypi4-64-0.0.1.wic.xz	131 MB
 Source code (zip)	
 Source code (tar.gz)	

- Recommended for users

Flashing and Booting

- Flash the image on a microSD card using Balena Etcher



- Alternatively, advanced Linux users, can flash the image through the terminal with **dd**:

```
sudo umount /dev/sdX*  
xzcat tmp/deploy/images/raspberrypi4-64/core-image-homebridge-raspberrypi4-64.wic.xz |  
sudo dd of=/dev/sdX bs=4M
```

- Plug the microSD card and turn on your Raspberry Pi

Homebridge Config-UI-X in Surf (web browser)

The screenshot shows the Homebridge Config-UI-X dashboard. At the top, there are navigation tabs for 'Status', 'Plugins', and 'Config'. The main area is divided into several sections: a QR code for pairing, system status indicators (Up To Date, Homebridge running, Plugins Out of Date), system information (CPU: 60% Load, 49°C Temp, Memory: 1.82 GB Total / 1.25 GB Free, Uptime: 4m), and a terminal window showing logs. The logs indicate that Homebridge is running on port 51826 and that the user 'admin' has been set as default.

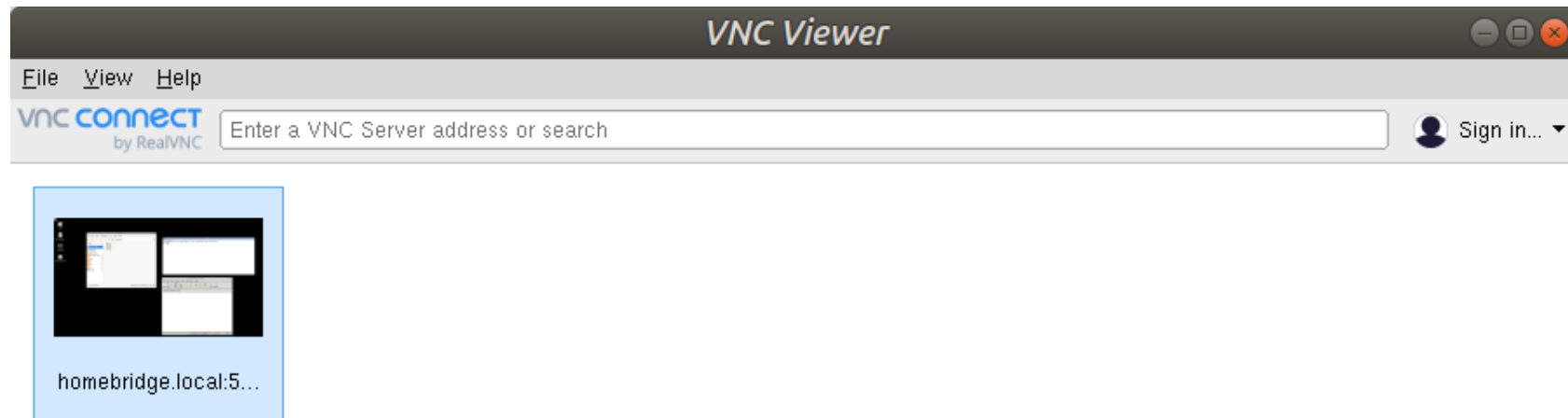
```
1 {
2   "bridge": {
3     "name": "Homebridge",
4     "username": "CC:22:3D:E3:CE:30",
5     "port": 51826,
6     "pin": "031-45-154"
7   },
8   "description": "This is an example configuration file with one fake accessory and one fake platform. You can use this as a template for creating your own configuration file containing devices you actually own.",
9   "ports": {
10    "start": 52100,
11    "end": 52150,
12    "comment": "This section is used to control the range of ports that separate accessory (like camera or television) should be bind to."
13  },
14  "accessories": [],
15  "platforms": [
16    {
17      "name": "Config",
18      "port": 80,
19      "auth": "form",
20      "restart": "systemctl restart homebridge",
21      "sudo": false,
22      "log": {
23        "method": "systemd"
24      }
25    },
26    {
27      "platform": "config"
28    }
29  ]
30 }
```

The screenshot shows the 'Plugins' page in Homebridge Config-UI-X. It features a search bar at the top and a list of plugins. Two plugins are visible: 'Homebridge Config UI X' (version v4.7.0) with an 'Update Available' badge, and 'Homebridge Myhome Tng' (version v0.0.21) which is 'Installed'. Each plugin entry includes the user's GitHub handle (@oznu and @angelox) and links for 'NPM', 'UPDATE', and 'SETTINGS'.

Connectivity and Interfacing Options

- SSH (port 22)
- VNC (port 5900)
- Serial

```
Leon@Leon-ThinkPad-T480s:~$ nmap homebridge.local
Starting Nmap 7.60 ( https://nmap.org ) at 2020-01-31 01:15 EET
Nmap scan report for homebridge.local (192.168.4.11)
Host is up (0.0044s latency).
rDNS record for 192.168.4.11: homebridge
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain
80/tcp    open  http
5900/tcp   open  vnc
Nmap done: 1 IP address (1 host up) scanned in 0.48 seconds
```



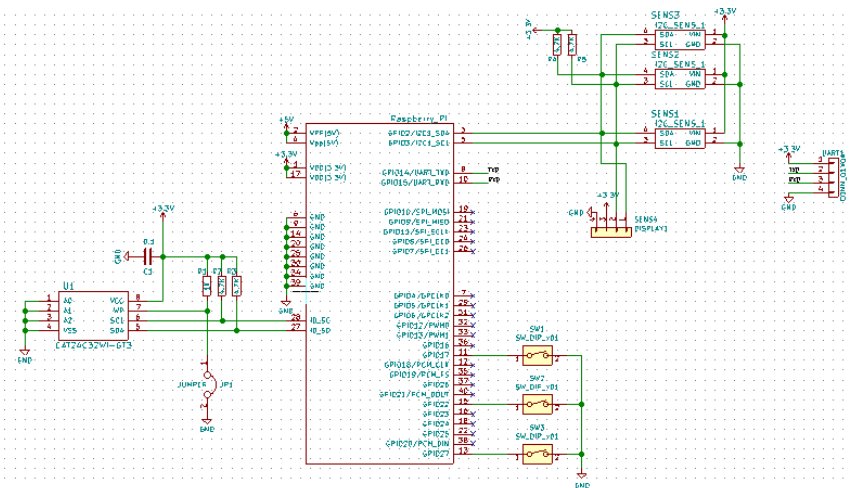
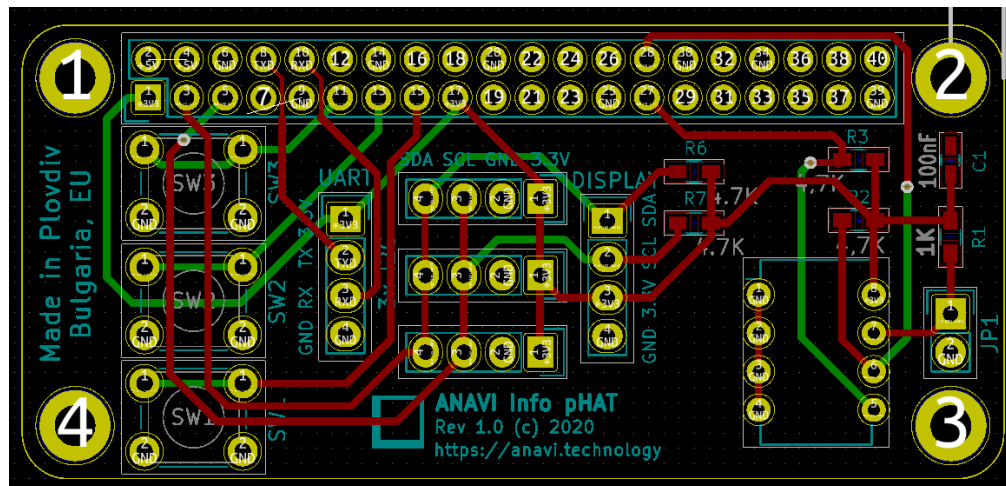
Mounting Raspberry Pi on DIN Rail

- “DIN rail is a metal rail of a standard type widely used for mounting circuit breakers and industrial control equipment inside equipment racks”
https://en.wikipedia.org/wiki/DIN_rail
- Camdenboss cases
<https://www.camdenboss.com/news/posts/2019/september/raspberry-pi-din-rail-enclosure/>
- Joy-It cases for Raspberry Pi 4B **or** B+, 2B, 3B and 3B+
<https://www.joy-it.net/en/products/RB-CaseP4+07>
<https://www.joy-it.net/en/products/RB-Case+07>



Raspberry Pi HAT for mini OLED display

- Open source hardware Raspberry Pi hardware attached on top (HAT) with slot for attaching mini OLED display (SSD1306) over I2C, designed with KiCad
- Python 3 script for drawing on the display with luma.core and luma.oled



Raspberry Pi 4 with Case for DIN Rail



How Does it Work?

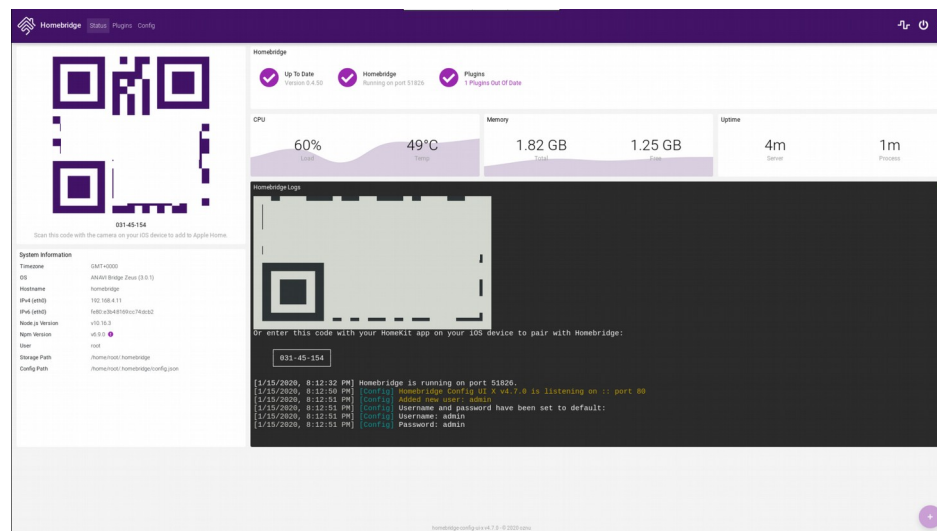
- Systemd service starts Homebridge and its plugins
- Openbox with X11 starts Surf (web browser) automatically
- Surf displays the Config-UI-X web interface of Homebridge
- Systemd services starts Python script for showing the statuses of Homebridge and Mosquitto on mini OLED display attached over I2C

Yocto/OpenEmbedded Layers

- Poky
- meta-raspberrypi
- meta-openembedded/meta-oe
- meta-openembedded/meta-python
- meta-openembedded/meta-gnome
- meta-openembedded/meta-networking
- meta-homebridge

Surf (web browser)

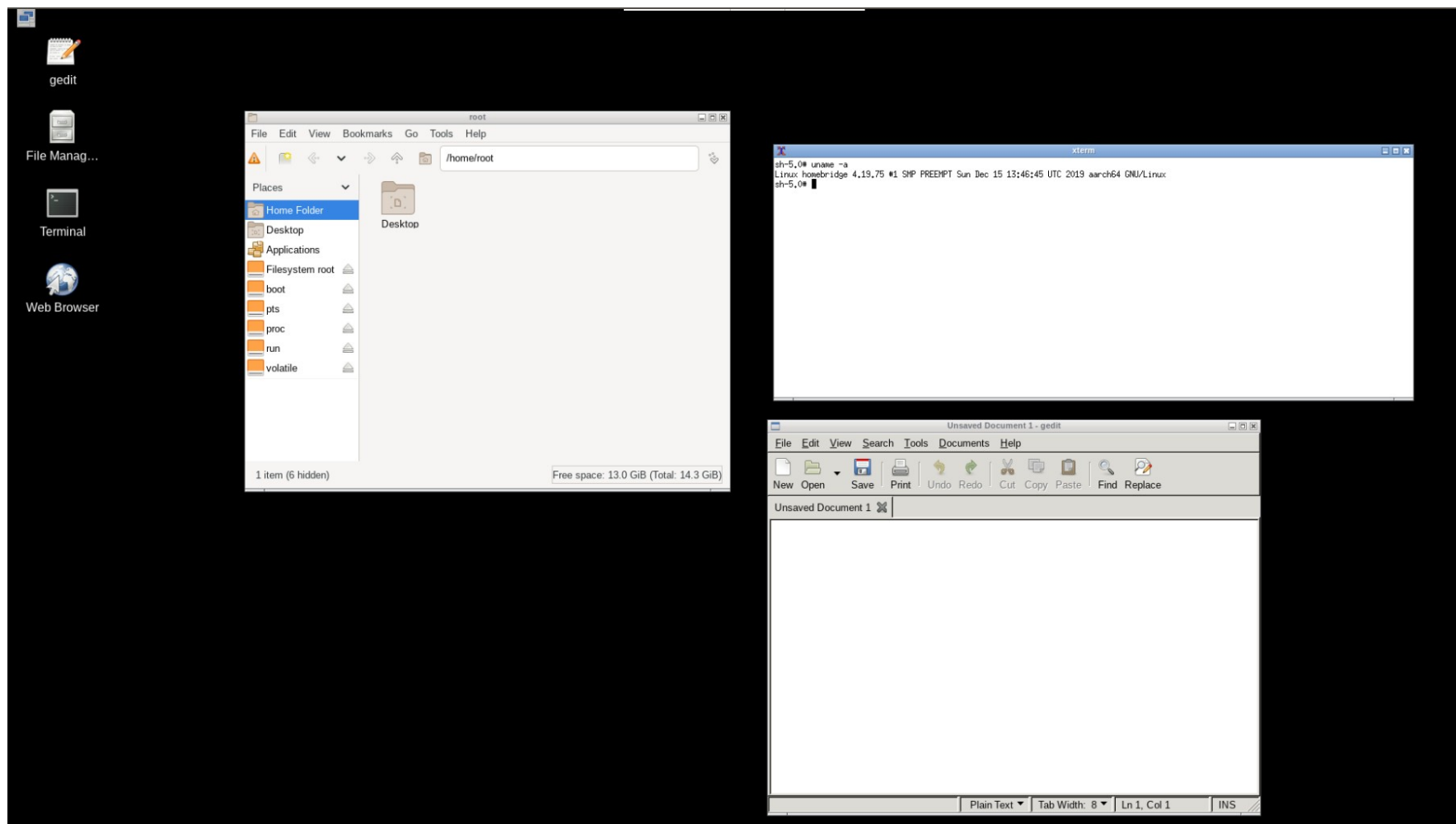
- Minimalist web browser without any graphical control elements, controlled by keyboard shortcuts
- Developed by suckless.org
- Written in C with WebKitGTK
- Available under MIT License
- <https://surf.suckless.org/>



Openbox

- Highly configurable stacking window manager for X11
- Written in C and XML for configurations, licensed under GPLv2
- **rc.xml** - main configuration file of the overall session
- **menu.xml** - configuration file for the desktop menu, accessible by right-clicking the background
- **autostart** - automatically starts applications, for our distribution: nm-applet and stalonetray
- <http://openbox.org/>

Openbox



homebridge_0.4.50.bb

- Snippet from the Yocto/OE recipe:

```
inherit npm systemd
```

```
SRC_URI = "npm://registry.npmjs.org;name=${BPN};version=${PV} \  
    file://config.json \  
    file://homebridge \  
    file://homebridge.service \  
"
```

```
NPM_SHRINKWRAP := "${THISDIR}/${PN}/npm-shrinkwrap.json"
```

```
NPM_LOCKDOWN := "${THISDIR}/${PN}/lockdown.json"
```


```
S = "${WORKDIR}/npmpkg"
```

```
RDEPENDS_${PN} += " homebridge-config-ui-x"
```


Homebridge at npmjs.com

homebridge

0.4.50 • Public • Published 8 months ago

 [Readme](#)

 [Explore](#) BETA

 [6 Dependencies](#)

 [34 Dependents](#)

 [76 Versions](#)



HOMEBRIDGE

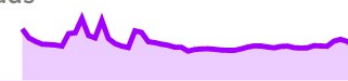
Homebridge

Install

```
> npm i homebridge
```

± Weekly Downloads

3,969



Version

0.4.50

License

ISC

homebridge.service

```
[Unit]
Description=Homebridge
After=syslog.target network-online.target

[Service]
Type=simple
EnvironmentFile=/etc/default/homebridge
ExecStart=/usr/bin/homebridge \$HOMEBRIDGE_OPTS
Restart=on-failure
RestartSec=3
KillMode=process
CapabilityBoundingSet=CAP_IPC_LOCK CAP_NET_ADMIN CAP_NET_BIND_SERVICE
CAP_NET_RAW CAP_SETGID CAP_SETUID CAP_SYS_CHROOT CAP_CHOWN
CAP_FOWNER CAP_DAC_OVERRIDE CAP_AUDIT_WRITE CAP_SYS_ADMIN
AmbientCapabilities=CAP_NET_RAW

[Install]
WantedBy=multi-user.target
```

What's Next?

TODO:

- Continuous integration (CI) and support for future releases of the Yocto Project
- Support more hardware platforms, especially STM32MP1
- Software over the air updates:
Mender.io or OSTree with meta-updater
- Integration of more Homebridge plugins out of the box
- Integration of additional open source home automation tools

Benefits for the Ecosystem

Hopefully my efforts so far had the following impact:

- User-friendly Linux distribution for providing Homebridge and other IoT tools out of the box
- Practical example for using Yocto and OpenEmbedded in a maker's project
- Upstream contributions to **meta-openembedded** to add completely new recipes for **surf** (web browser) and **stalonetray**, to update and improve the recipes for **mosquitto** and **openbox**

Conclusions

- **Homebridge** is an excellent open source software to connect non-officially supported Internet of Things and do-it-yourself (DIY) devices to Apple HomeKit and Siri through various open source plugins
- **The Yocto Project** and **OpenEmbedded** are super powerful tools for building and optimizing GNU/Linux distribution for the very specific needs of a particular embedded device
- Although the Yocto Project is de-facto an industry standard, it is still not maker-friendly because of the steep learning curve, long build times and sometimes missing recipes for software that is existing as packages in the ecosystems of popular GNU/Linux distributions like Debian

Thank you! Any Questions?

- <https://homebridge.io/>
- <https://www.npmjs.com/package/homebridge-config-ui-x>
- <https://www.yoctoproject.org/docs/current/mega-manual/mega-manual.html>
- https://wiki.yoctoproject.org/wiki/Main_Page

