

Contact Information

- Name: Morgan Gangwere
- Country: United States
- School and degree: University of New Mexico, Albuquerque NM; Communications
- Email: morgan.gangwere@gmail.com
- Twitter: @indrora
- IRC: indrora on freenode
- Phone: +1 505 639 7551
- Which method of communication do you prefer? (i.e. in person, email, chat, video conference, etc.): IRC, email, in-person when possible. Hangouts and Skype are fine too. GitHub PRs are awesome too. Gitter is totally fine too.
- What Berkman project(s) are you interested in? LibraryBox

providing feedback: Post feedback Via e-mail or IRC.

About Me/My reasons

I'm a student at the University of New Mexico in Albuquerque NM studying Organizational Communications; I spend my summers in Seattle so I can be sous-vide cooked by the time I get back for the New Mexico winters. I've been writing software for about the last 12 years, almost entirely self-taught. I've "lagged behind" on a regular basis, focusing my development efforts on *low-power, low-cost devices* (up until very recently, my development device was a Pentium 3 laptop with 512M of RAM) and my work today in software focuses on low-resource VPS environments (my personal website runs in 128M of RAM on a single core).

I'm interested in the LibraryBox project because I've used it. I want to get LibraryBox running on a bunch of platforms and on a long-term stable system that can be brought to different hardware easily. I should be able to scavenge hardware and make it work, and a lot of libraries don't have much money to start with!

meta

- website: <https://zaibatsutel.net/>
- I've checked the dates on GSoC

- I know this is a full-time commitment
- My only schedule conflict would be talking at Open Source Bridge in late June.
- I host a LOT of code at GitHub: <https://github.com/indrora>

I will potentially have one conflict: I am possibly speaking at OpenSource Bridge in late June.

Proposal

Overview

Currently, LibraryBox is built upon OpenWRT. While fine for the traditional "plug and go" setup, the TP-Link devices targeted by the LibraryBox distribution are either harder to find, or potential FCC regulations may cause them to become locked down. While this is unlikely in the long run, there is no "simple" way to install LibreBox on other embedded devices such as the Raspberry Pi.

Requested by the team is

Working with advice from the leads of both the Piratebox and Librarybox projects, the first goal will be to leverage the existing codebase onto the Raspberry Pi 3, complete with build instructions and installer that will enable novice users to build their own.

From this, I will build an installer script that guides a novice to advanced user through the process of setting up their own LibraryBox from scratch on a Raspberry Pi 3's suggested Debian installation.

Detail

Fundamentally, LibraryBox can be implemented as "stuff tossed on top of Debian" -- Much of the configuration can be done on top of a sparse Debian root filesystem. To facilitate this, I will:

- Develop a simple installer that installs prerequisites for the system
 - Will be able to interactively configure and install

- Allows the user to choose how to store files: On the SD card (e.g. on F2FS) or on an external storage device (e.g. USB disk)
- Build a toolchain that generates images for the Raspberry Pi 3 (possibly 2?)
 - installation should be as simple as "burn this to your SD card and it will work"
 - two flavors of pre-baked image possible: One that is intended for BIG SD cards (64GB or so) and uses F2FS, another that prompts for a device during configuration?
- Develop a mechanism to upgrade the debootstrap-based environment
 - OpenWRT uses their custom "sysupgrade" system. Debian doesn't have anything, but it should be possible to overwrite the contents of the SD card live with a new root filesystem

ArchLinuxArm vs. Debian (... or Raspbian)

The PirateBox project has chosen ArchLinuxArm on their side. While ALARM is very spiffy and has a wide support for a large number of ARM chipsets, it has a definite downside: Rolling releases and embedded just don't jive for me: they present too moving a target. Debian's official team for ARM has a history of providing good quality packages, with the Debian project overall focusing on stability and "just works" ness.

There are two options for support on the Raspberry Pi:

- Debian mainline; Lots of work bringing images up to speed
- Sitting on top of Raspbian: We'll have to mangle the root filesystem ourselves.
- Sitting on top of ArchLinuxArm: We'll have to mangle the rootfs. Rolling release + embedded = bad time.

Debian Mainline has mixed support: For the pi3, there's still some work to get everything completely set up < <https://people.debian.org/~stapelberg/2016/11/24/raspberry-pi-3.html> and <https://people.debian.org/~stapelberg//2017/03/22/raspberry-pi-3.html> > and still lacks support for the Wifi/Bluetooth

However...

Bluetooth and 802.11 are apparently supported through the use of < <https://github.com/drtyhlpr/rpi23-gen-image> >. This does come with the cost of requiring a complete kernel compile, but gives us access to the Debian mainline ARM repos, as opposed to the second-hand repos from Raspbian.

Modification of an existing rootfs makes the expandability to other platforms very simple; for example, the CHIP from NextThingCo would be easy to bring up with only minor modifications for hostapd and storage.

Goals

Pre-Midterm

- Getting LibraryBox on the RPi 3 done "DIY" -- Feature-complete with the TP-Link devices. (1wks)
- Write an automated installer ("stage1") that guides the user through the steps necessary to install from a clean Raspbian install (2-3wks)
- re-build the configuration frontend ("stage2") to be a little nicer (1-2wks)

The automated installer will handle

- Installing prerequisite packages for a Debian system + getting LibraryBox scripts set up ("stage 1")
- Configuration of ("Stage 2")
 - Configuring storage settings
 - Configuring wireless settings (hostap, etc.)
 - Adding any custom branding/static content

Stage 2 can be either terminal based (via `dialog`) or web-based (built on Flask?). Stage 2 would replace the existing configuration tool for sync/FTP/etc configuration.

At this point, if there's time, I'd like to add support for iBeacon advertisement and do cleanup to make things more polished

Post-Midterm

- Pre-baked images which offer support for a variety of SD card configurations (e.g. "frozen" releases, "moddable" releases) (3-4wks)
 - 2-3wks "Modify the rootfs of an existing device in an automated way". See previous comment about NextThingCo CHIP.
 - 1wk to attempt to build a working rootfs from rpi23-gen-image + modify it.
- Full documentation on building, modding, and otherwise extending everything. (2wks)

The pre-baked environment should: *Have the environment installed to the point of hitting Stage2* Have an SSH user that guides the user through stage 2 *Be upgradable via SSH and later via a web interface* Be "Burn and run" ready: A user should have to only use a single tool such as Resin.io's burning tool to do the magic.

Things I'd like to get done:

- iBeacons; There is a script to do it < <https://github.com/dburr/linux-ibeacon> > and the official Google Eddystone toolchest has an implementation < <https://github.com/google/eddystone/blob/master/eddystone-url/implementations/linux/advertise-url> > but which I'm not too familiar with. iBeacons aren't my forte, but I'd like to at least see basic support tossed in.

Maintainability

The topic of maintainability came up on Gittr by MaStr. However, after conversation with Griffey, this project doesn't need to keep it totally in mind; I however would like to, with the help of the LibraryBox and PirateBox team, help develop it further afterwards.