SAMSUNG
Sat Jan 01, 12:02 AM
postmarketOS demos
weston-presentation-shm (Animation)
weston-simple-damage (Animation)
Wayland Terminal

postmarketOS: Aiming for a 10 year life-cycle for smartphones

2017-05-26 / 8 min

Introduction post to postmarketOS, a touch-optimized, pre-configured Alpine Linux with own packages, that can be installed on smartphones. (Not usable for most people yet!)

Update: Alpine Linux developers do not see their distribution as "GNU/Linux", so I won't be using this term for Alpine or pmOS anymore. One of these developers, ^7heo, helped me to reword this article and other pages on this blog accordingly, thank you!

Index

Minimalistic Linux distributions run fine on ten year old PCs.

It is 2017. Pick an average PC from 2007 and install a minimal Linux based operating system. You will be able to do basic computing tasks (eg. surfing the web, reading E-Mails, listening to music, chatting) just like on an *expensive* modern PC. You will even get security updates, so your old computer is protected, just like as a new one.

Why are Android/Linux phones different?

Androids architecture is based on **forking** (one might as well say *copy-pasting*) **the entire code-base for each and every device *and* Android version.** And then working on that independent, basically instantly incompatible version. Especially adding device-specific drivers plays an important role.

This workflow makes it next to impossible to patch all Android devices with security updates in time or at all (*Stagefright* vulnerabilities for example rendered one billion devices vulnerable). And even if the vendor provides updates, it will only be for a limited time and then you must buy a new device to get security updates or the latest Android version. How convenient!

Alright, so there is the LineageOS community, which provides weekly updates for an impressive number of smartphones. They provide a practical solution today, and I am very grateful for that. However, such Android based projects will always run behind Google and the phone industry, fixing only symptoms but never the root-cause.

This is just the tip of the iceberg. Android has way more problems, read Cascardo's GNU on Smartphones (part II) for more nightmares.

We can fix this as a community.

Here is the solution: Bend an existing Linux distribution to run on smartphones. Apply all necessary changes as small patches and upstream them, where it makes sense.

Of course I am not the only one, that came to this conclusion - especially in the last few weeks with the Halium project rising *(greetings!)*. I am all-in for working together — sharing udev rules, merging Android kernels together, whatever makes sense!

On-screen keyboard
is not implemented yet.
Plug in a USB cable and
run on your PC:
telnet 192.168.2.15

postmarketOS architecture

I'm working on an Alpine Linux based distribution called postmarketOS where each phone will have **only one unique package** — all other packages are shared among all devices.

These device-$vendor-$name packages contain a so-called /etc/deviceinfo file, which describes what makes the device special: SD card availability, which flash software to use and other information. The file format is not stable yet, and once we have common kernels for multiple devices, I'd like to include the required modules and dtb name.

And just to make it clear, postmarketOS does not fit the Halium model, as it avoids the Android build system entirely and does *not* run any part of the Android userspace next to its more or less typical

Linux userspace. *(At least not in the regular install, but it could come at some point in the future as optional compatibility layer for Android applications if someone wants to work on it. Personally, I'd rather have native Linux applications (in the case of Alpine: linked against musl, dynamically or statically) than Android apps on my phone.)*

Prototype

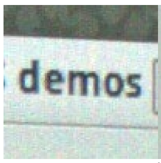
So much for the theory. Practically I can show you a sophisticated chroot/build/flash tool called **pmbootstrap** which should allow **fast and clean development progress**, both in porting to new phones and in implementing hardware support for the existing ports *(more on pmbootstrap below)*. That is also where most of the time went during development so far, so don't expect too much of postmarketOS. Most drivers don't work so you can't make phone calls or use the WiFi. Nevertheless, here is the current state of postmarketOS.

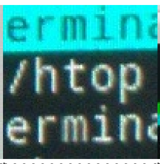
Devices:

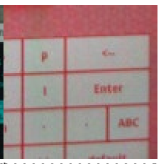
- Samsung Galaxy SII (samsung-i9100)
- Google Nexus 4 (lg-mako)

Features:

- Encrypted root file system (password needs to be typed in via USB telnet right now)
- Installation on SD card or internal memory
- Weston with working touchscreen
- Weston demos work performantly (*without* proprietary 3D acceleration!)
- SSH access via USB
- Kernels compiled from LineageOS source
- Clean package manager based installation







Generally speaking, the samsung-i9100 port works better than the lg-mako port. The latter has some strange driver bugs. The boot splash images only appear for a second, XWayland does not work there and the colors in Weston are all red. I even had to patch Weston before it worked at all. Without the patch it assumes it should draw with 0 Hz — in other words: Never ;)

It should be noted, that the Replicant, the free software clone of Android, also targets the samsung-i9100 and has open source user-space drivers for the modem, which I plan to package for postmarketOS/Alpine Linux.

pmbootstrap

Technical details incoming! If you're not into that, skip this section.

Alpine Linux is *really small*. A base installation is only about 6 MB in size and takes not more than a few seconds to extract! Thanks to this characteristic, I was able to write a bootstrap program that abstracts everything in chroots and therefore basically runs on top of any Linux distribution (GNU-based or not), which has Python 3 and the openssl command line program available.

Consequently, the host system does not get touched when installing the required programs (fastboot etc.) and your distribution doesn't even need to have them packaged.

Quick feature rundown:

- Chroot setup (with distro-independent QEMU user emulation):
  - x86_64* (building, flashing, ...)
  - armhf* (building)
  - armhf* (target rootfs)
- Clean chroot shutdown (umount) and zapping
- Build software as packages:
  - Wraps abuild, the "light version of makepkg"
  - Alpine Linux' APKBUILDs are very similar to Arch Linux' PKGBUILDs
- Cross-compile **all** armhf-packages:
  - Linux Kernel: build with cross-compiler in x86_64 chroot
  - Other: build in armhf chroot, use cross-compiler with distcc from x86_64 chroot (alarm-style)
  - Use Alpine Linux' shipped modern gcc, no pre-built Android toolchain
- Effective caching out of the box (survives chroot zaps):
  - ccache (also works with distcc/cross-compiler)
  - Alpine Linux package cache
- Installation targets:
  - Raw image file (flash as "system" partition)
  - SD card
- Flasher abstraction:
  - fastboot
  - heimdall
  - ... really easy to add more!
- Logging:
  - all shell commands executed are logged in an extra file
  - readable overview is displayed on the screen
- Security:
  - Initial package manager download
    - Signature verification with openssl against keys shipped with pmbootstrap
    - Minimum installed version check (for downloaded package and version reported by the extracted binary)
  - All executed shell commands get properly escaped with Python's built-in shlex
  - Working testcases for the above two points
  - Only using root rights where necessary (through sudo)
  - No default passwords in the installation: The install action asks for the *user's* and for the *root partition* password.

* x86_64/armhf: Example architectures for host/target. The code is generic, so it should work with any architectures

supported by Alpine Linux.

Future goals and where *you* could help

Rough direction of where I'd like postmarketOS to head to. In case you're a hacker who wants to help, feel free to do so. But please write into the tracker before starting serious work. This way we can make sure, that we do not have redundant work.

Devices

Pick an old Android device, that you don't need anymore and start porting postmarketOS for it. It should be pretty straight forward. One device-* package, one for the kernel, calibrate the touchscreen, and the demos should more or less work already!

If you're feeling adventurous, try a non-Android device. How about iPhones? I'll probably be working on a port for at least one non-Android device myself for demonstration purposes.

Drivers

Fix the lg-mako screen colors and/or make the following peripherals work:

- WiFi
- Audio
- Modem (Phone calls, mobile internet)
- Hardware buttons (Volume keys, home button)
- Camera
- ...

In most cases, the drivers are already provided by the Android/LineageOS kernels, that we currently use and only need to be configured in the userspace (for example with udev rules).

The long time goal is using the mainline kernel.

Phone interface

Package a Linux-based phone interface. Plasma mobile *seems* to be the most complete one right now (although still not stable). But I'd also be interested in ubports (fork of the discontinued Ubuntu Touch) once it matures. Or maybe writing a minimal Android-like interface based on libweston.

From what I understand, the SailfishOS interface is closed source, so that will not be an option.

postmarketOS is developed in the spirit of regular Linux distributions, so there's no problem in having multiple phone interfaces (just like KDE/Gnome/XFCE/...) and let the user choose.

Security

Great care has been taken to make pmbootstrap safe, as it will run on productive systems of postmarketOS developers. This is not the case for postmarketOS in its current proof-of-concept state (Weston runs as root, ...) so we must work on that before it can be used in real life. Even better would be privilege separation throughout the entire OS.

There is so much more...

...but the blog post is long enough as it is, so I'll wrap it up. Thanks for reading, thanks to my friends who reviewed earlier versions of this blog post. Thanks to Replicant, LineageOS, Halium. Together, we can make the vision of long-lasting, open source smartphone operating systems a reality!

Tell your friends!

If you want to read more, make sure to subscribe to the RSS feed and/or watch the project on GitHub.

Comments: reddit, Hacker News