SoC Vendor Panel

Prpl Summit 2019
Berlin
2019-10-23

Introduction

prpl

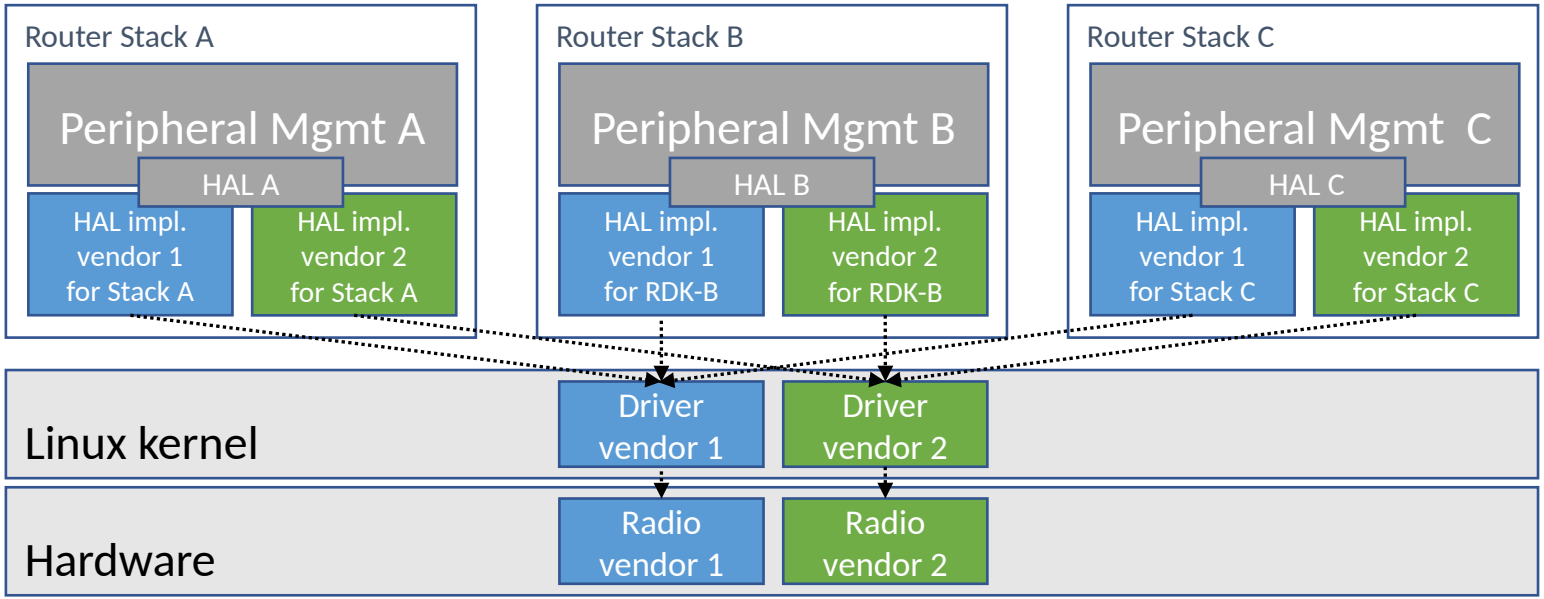
Low Level API

History and Problem Statement

LL API: Problem

Problem: incompatibility between device drivers

Mitigation: creation of a driver specific, software stack specific Hardware Abstraction Layer (HAL)

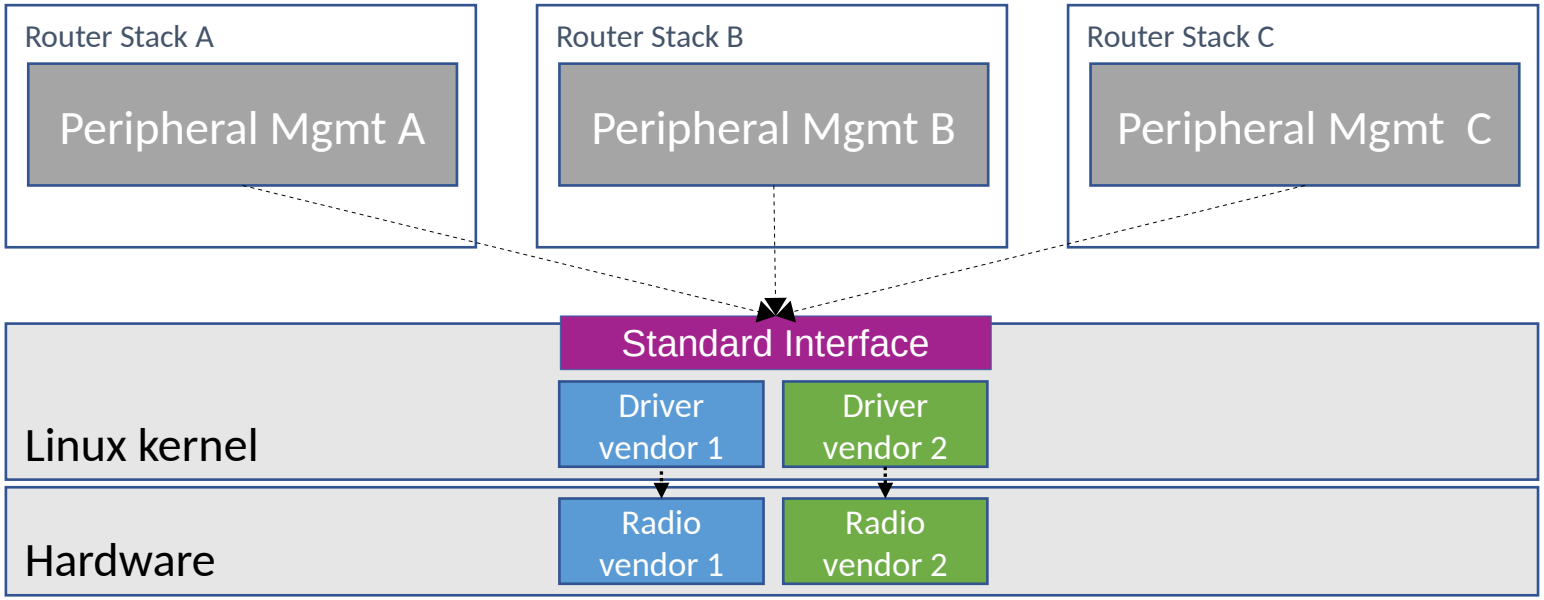


LL API: Solution

Solution: standard Linux peripheral interfaces

HAL interfaces disappear

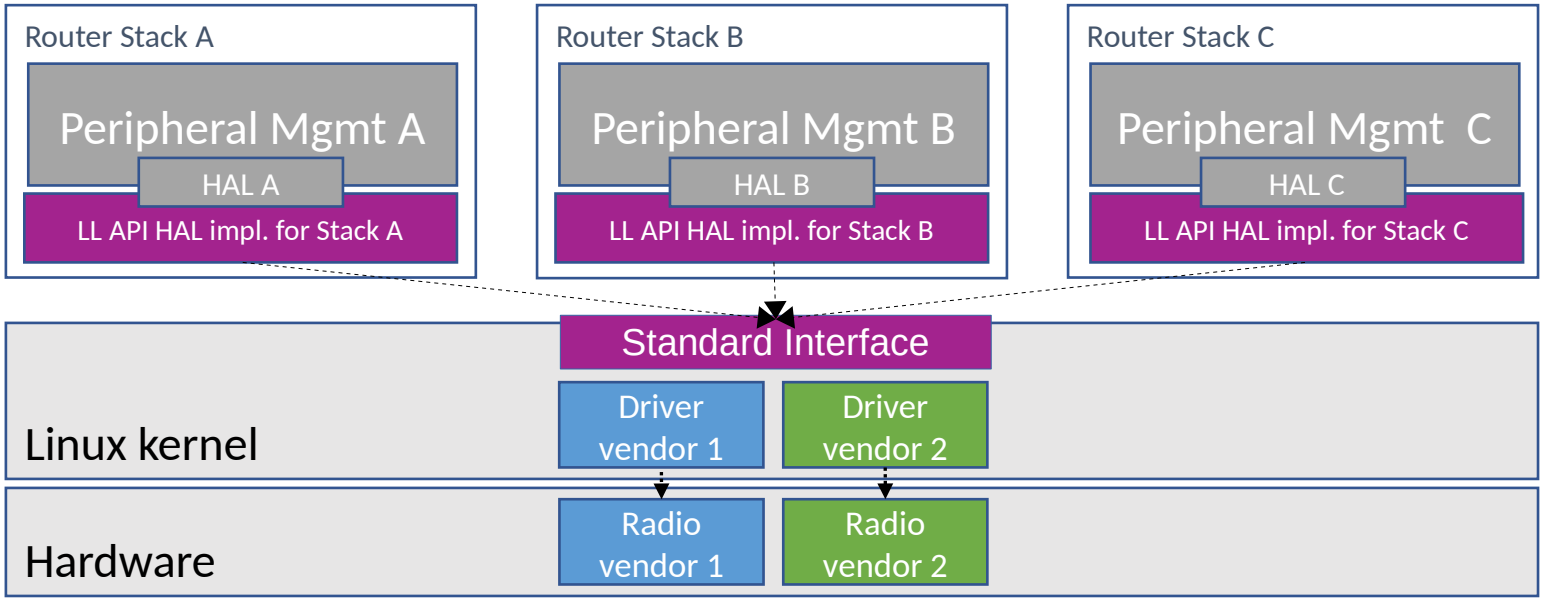
Less effort for silicon vendors, OEM's, software developers, system integrators → everybody wins



Transition Strategy

Router stack vendor develops new HAL implementation → extra effort

Over time, as the need to support old drivers disappears, the HAL *may* be removed



Low Level Interfaces under Definition

Simple Peripherals

GPIO

LED

PWM

pushbutton

I²C

SPI

sensors

Storage

NOR/NAND Flash

CPU Features

CPU frequency scaling to reduce power consumption

Network SoC Features

hardware packet steering

flowtable offload

hardware packet offload

multicast packet offload

QoS queue configuration

QoS packet classification

A/VDSL, G.fast

VoIP

Network Peripherals

Ethernet switch

switch VLAN support

wireless LAN – 2017

Security Features

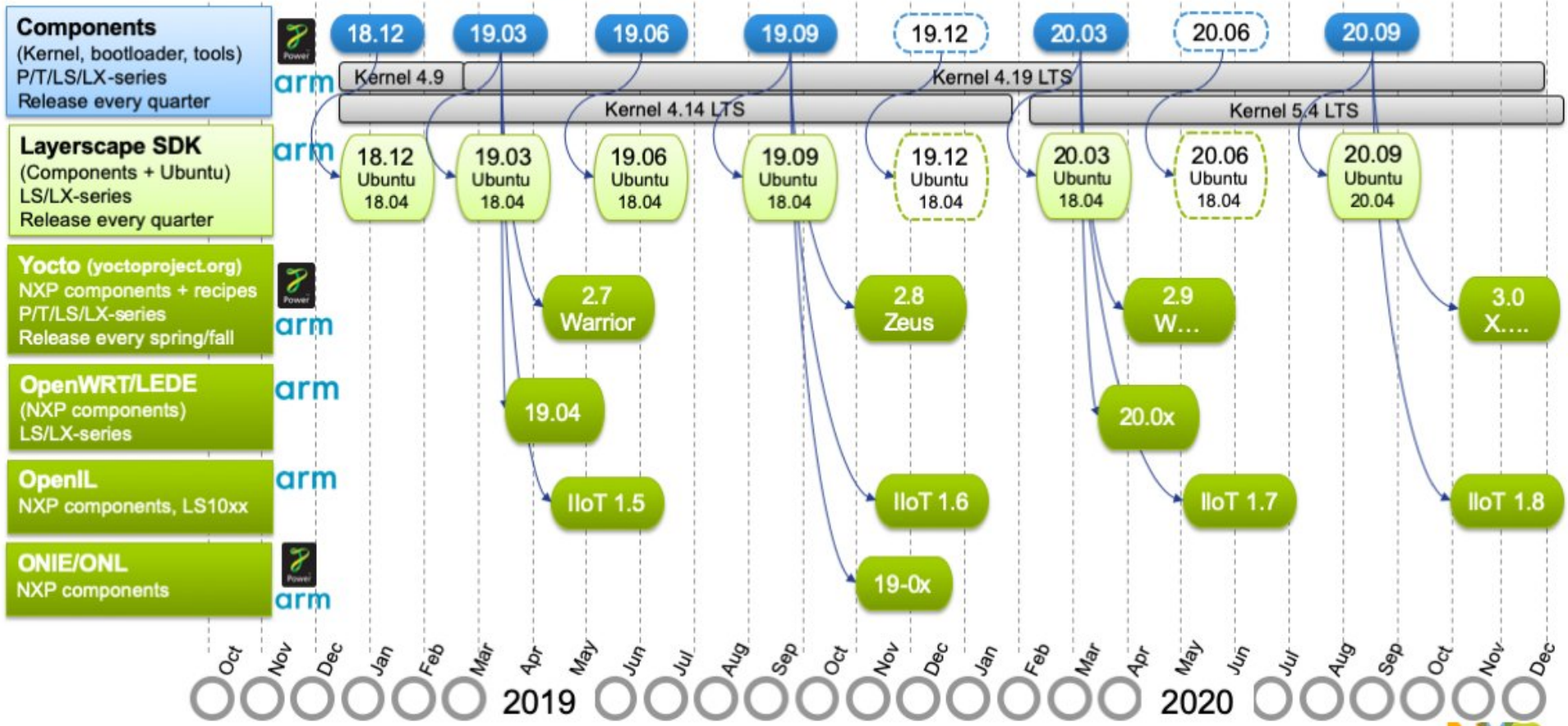
hardware crypto

hardware random number generation

Linux containers

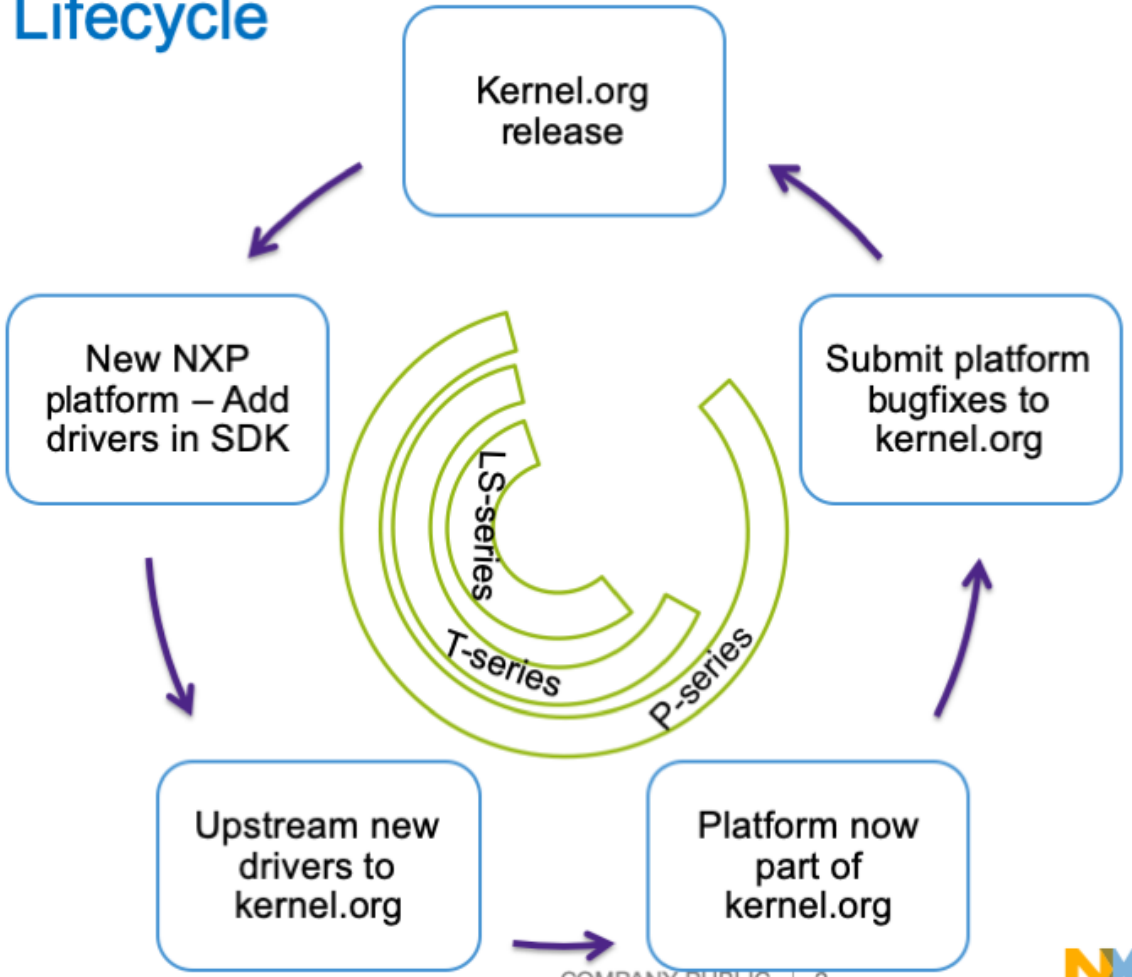
prpl

Linux Release Strategy for ARM and Power



Platform Upstreaming Lifecycle

- Upstreaming is ongoing process.
- For a product family upstreaming may be 2-3 years to completion.
- Meanwhile, NXP provides platform patches via the SDK or via partners.
- When complete, customers encouraged to use `kernel.org`.
- Maintenance provided via `kernel.org` – NXP upstreams fixes



prplMesh

History and Problem Statement

prplMesh Project

- Goals:
  - Open source reference implementation of Wi-Fi Alliance CERTIFIED EasyMesh™.
  - Standardised wireless management across router stacks, using a standard API.
  - Initial primary goal: vehicle to map out wireless LAN low-level API interactions for certification.
- Benefits:
  - Consistency of behaviour.
  - Carrier-grade out of the (open source) box.

prplWrt

prplWrt Project

- Showcase to bring all the Prpl Foundation projects together.
  - Low-level API
  - SSI API
  - prplMesh
  - LCM
  - Security
- OpenWrt support for carrier grade, top of the line silicon.

Questions

Closing Statements

Thank you

Low Level API Goal in Text

The problem to be solved: the lack of a common API as defined by Linux. This results in incompatible, vendor-specific interfaces, and the need for every integrator (OEM, router software stack developer) to define and implement its own hardware abstraction layers. Prpl's role is to provide that common API by working with the open source community (Linux, OpenWrt, RDK-B) to provide those API's, and to help silicon vendors to support it in their drivers.

Proposed Charter for LL API WG

1. **Identify** points of divergence of **software interfaces** to functionalities, implemented in hardware, microcode or firmware, embedded in network SoC's and peripheral hardware (referred to as **low-level API's**) on **CPE gateway** and wireless access point devices.
2. Work with silicon vendors, software and hardware CPE integrators and the Open Source community, specifically the Linux community, to **define common interfaces**. The guiding principles are the following.
    1. The interfaces SHOULD be **in the Linux kernel** to reduce complexity of integration and to work towards convergence with the desktop, server, core networking equipment and other embedded niches such as IoT and set-top boxes.
    2. The interfaces MUST be **open**, allowing any silicon vendor to deliver, possibly proprietary, driver code that can plug into the interface's framework.
    3. Where suitable interfaces already exist, any **gaps** must be identified and **addressed through implementation**.
    4. New interfaces MUST be **upstreamed** to the appropriate projects (e.g. the Linux kernel), or, if none exist, published in the open by the Prpl Foundation.

Proposed Charter for LL API WG

3. Document and **publish** the interfaces and implementation guidelines, at a high level to improve understanding, and at a detailed level.
4. Verify and report **compliance**.
5. Align architecture and implementations with the LL API through **collaboration** with
    - other Prpl WG's, such as prplWrt and prplMesh,
    - **open source router communities**, such as OpenWrt and RDK-B, and with proprietary software stack vendors,
    - silicon vendors.
6. Work with **stakeholders** to define a **roadmap** to full compliance, and provide **assistance** with API backporting to older Linux kernel versions.
7. Promote the development of compliant interfaces through advocacy. Strive to have the LL API **integrated in Service Provider specifications**.