6LoWPAN: An Open IoT Networking Protocol

Samsung Open Source Conference 2015
Seoul, Korea

Stefan Schmidt
Samsung Open Source Group
stefan@osg.samsung.com

Scope

6LoWPAN: An Open IoT Networking Protocol

- **Open:** Specified by the IETF
  - Specifications available without any membership or license fees
  - Designed and developed in public
- **IoT:** Making “Things” Internet-aware
  - Usage of IPv6 to make use of internet protocols
  - Leverage on the success of open protocols in contrast to proprietary solutions
- **Networking:** Stopping at layer 3
  - Application layer protocols are flexible and can vary
  - Often used together with CoAP, MQTT, etc

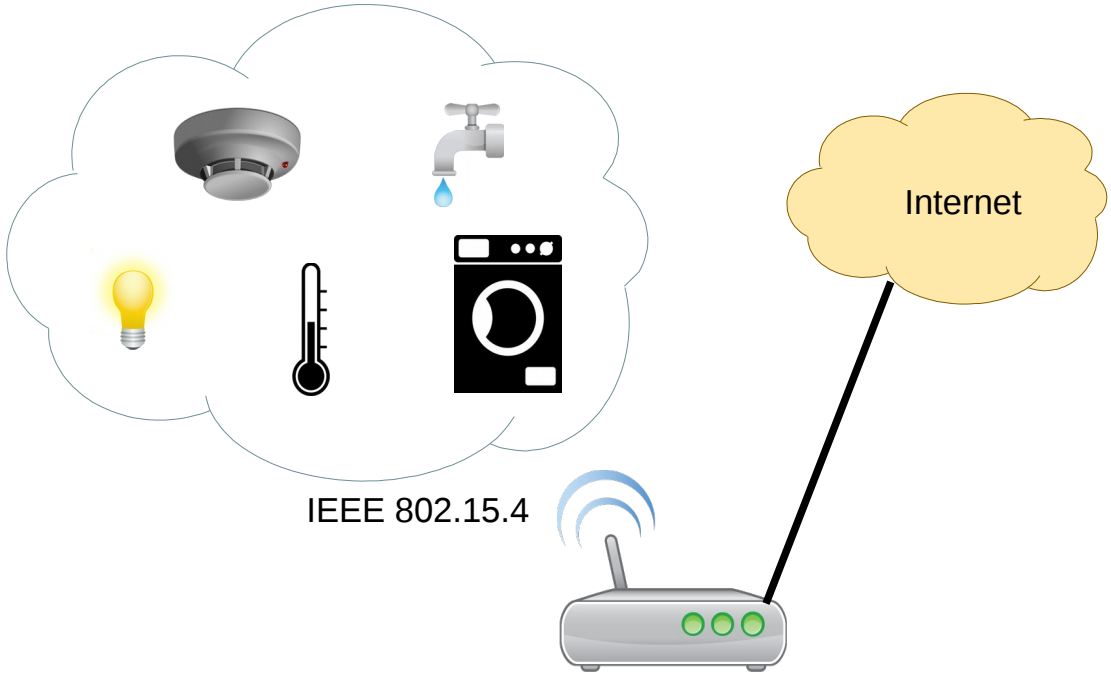
Agenda

- Motivation
- 6LoWPAN
- 6LoWPAN Header Compression
- Linux-wpan Status
- Future Work

Motivation

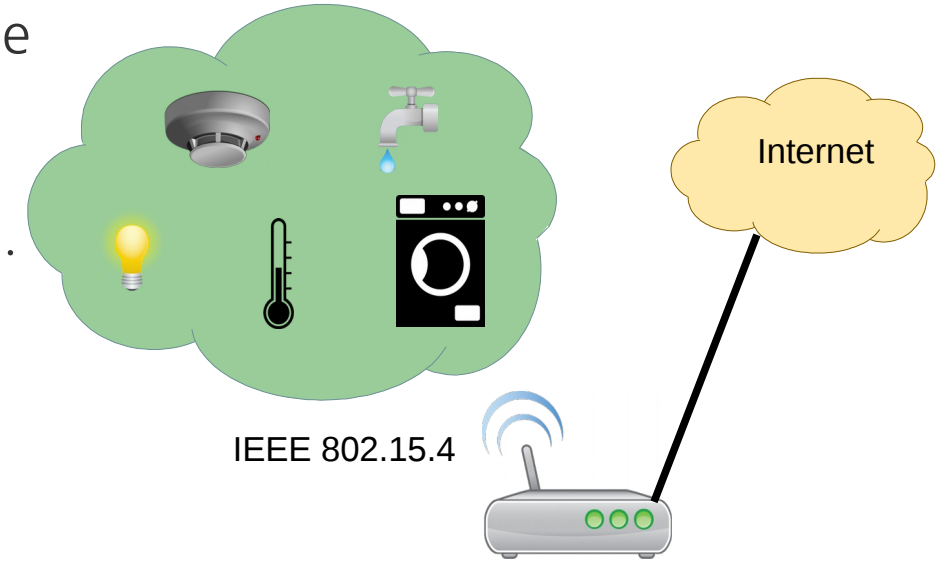
Use Cases

- Battery powered sensors (temp, smoke, etc)
- Main powered appliances (washing machine)
- WiFi accesspoints as Border Router / Gateways
- Home use
- Industrial use



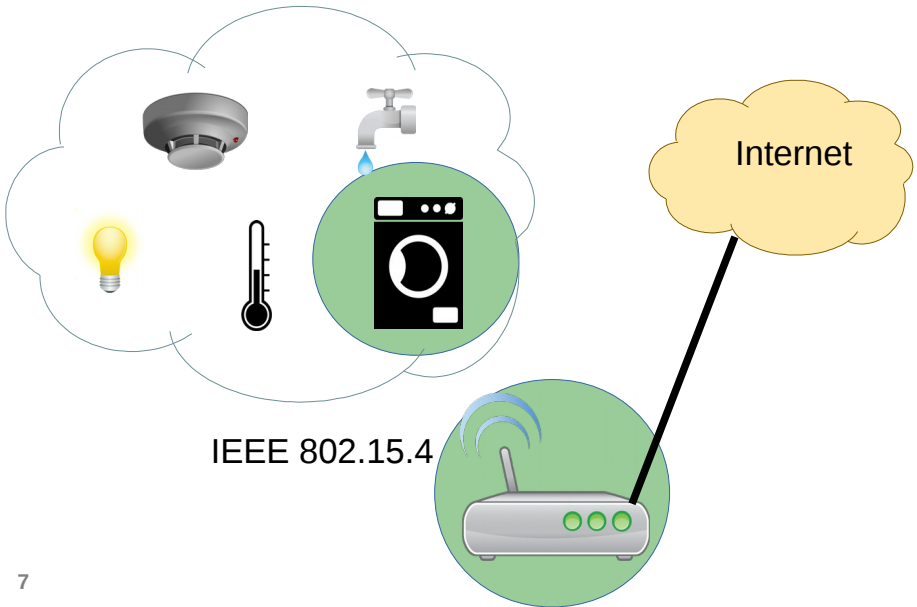
Motivation 6LoWPAN

- Sensors are likely to have restricted wireless connectivity
- Using IPv6 instead of something proprietary allows the usage of existing and proven protocols driving the internet
- A full unmodified TCP/IP stack might clash with hardware limitations (which are useful for power savings)
- Sensor only need to transfer little data, compared to the usage scenarios of a Smartphone, PC ..



Motivation Linux-wpan

- Battery powered sensors might not run Linux but choose a smaller OS
- Linux is the OS of choice for embedded products
- Main powered appliances might run Linux already and would benefit from native 6LoWPAN support
- IEEE 802.15.4 chips could easily be integrated in WiFi accesspoints or routers which already run Linux
- Thus a real benefit to have a 802.15.4 subsystem ready in the Linux kernel



Movement Towards IP

- A lot protocols are moving towards IP
- Often started out with their own networking stack
- Swichting to make use of the success of IP as protocol
- The name Internet of Things already implies that it should be modeled after the success of the Internet
  - Direct addressing of nodes
  - Re-usage of proven protocols
- But TCP/IP is not one size fits all
  - Adaptations needed for MTU size
  - Reduce of header overhead
  - UDP (DTLS) instead of TCP to avoid latencies

6LoWPAN

ZigBee Relations

- IEEE 802.15.4 is often mixed up with ZigBee
- It uses the PHY and MAC layers defined by IEEE 802.15.4
- Everything above Layer 2 was proprietary
- ZigBee IP seems to have switched to 6LoWPAN and keeping the rest on top of it
- ZigBee licensing seems incompatible with the GPL, no ZigBee support for the Linux Kernel

IEEE 802.15.4 / LoWPAN

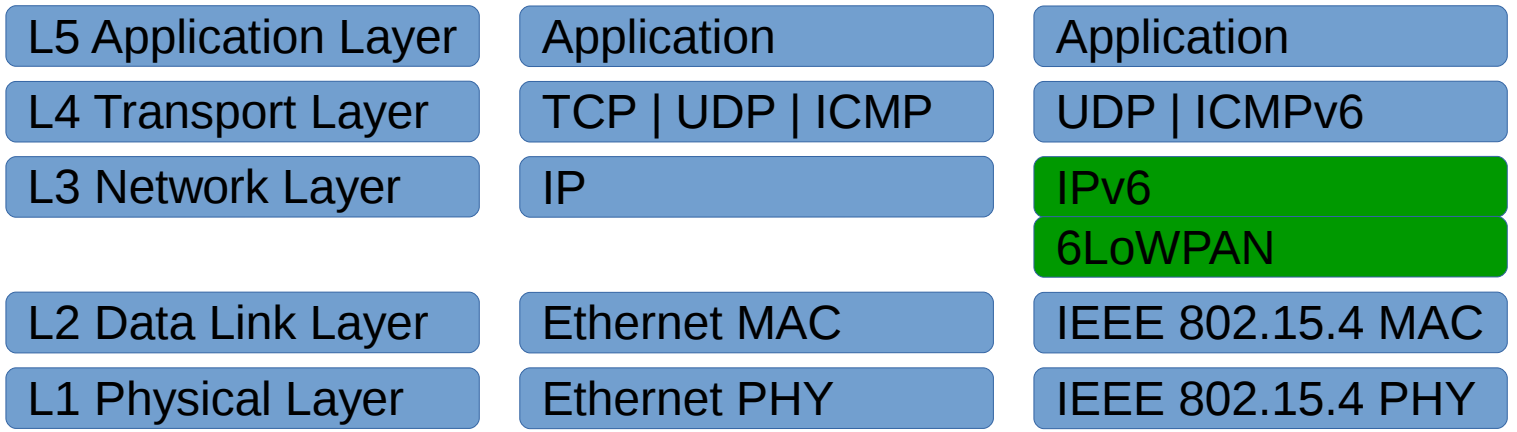
- IEEE 802.15.4 specifications, starting in 2003
- Low-Rate and low power Wireless Personal Area Networks
- Specifies the physical and the MAC layer
- Simple addressing but no routing
- Star and Peer-to-Peer topologies supported
- Mesh topologies need some layers on top of these
- Applications are small battery powered devices like sensors and actors in automation

6LoWPAN

- A series of IETF specifications, starting in 2007
- IPv6 over LoWPAN (IEEE 802.15.4)
- Direct IP addressing of nodes
- Adaptation layer between Data-Link and Network layer (RFC4944)
- Autoconfiguration with neighbor discovery (RFC4944)
- Header and payload compressions (RFC4944, RFC6282, RFC7400)
- Updates and extensions in other RFC's (see references at the end)

6LoWPAN Adaptation Layer

- The 6LoWPAN adaptation layer sits between Data-link and original Network layer
- It effectively becomes part of the Network layer, but only on the specified Data-Link layers

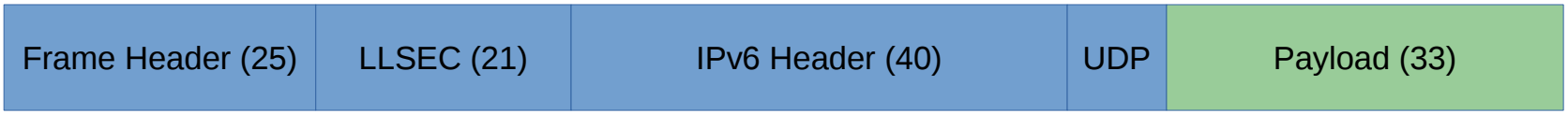


6LoWPAN Fragmentation

- IPv6 allows for a maximum packet size of 1280 bytes
- This is impossible to handle in the 127 byte MTU of IEEE 802.15.4 (other PHY's will vary here)
- Therefore 6LoWPAN defines a fragmentation scheme to allow such packets
- The 11 bit fragmentation header allows for 2048 byte packet size with fragmentation
- But fragmentation can still lead to bad performance in loosy networks
- Best to avoid big packet sizes
- 11 000xxx as FRAG1 dispatch value, 11 100xxx for FRAGN

The Header Size Problem

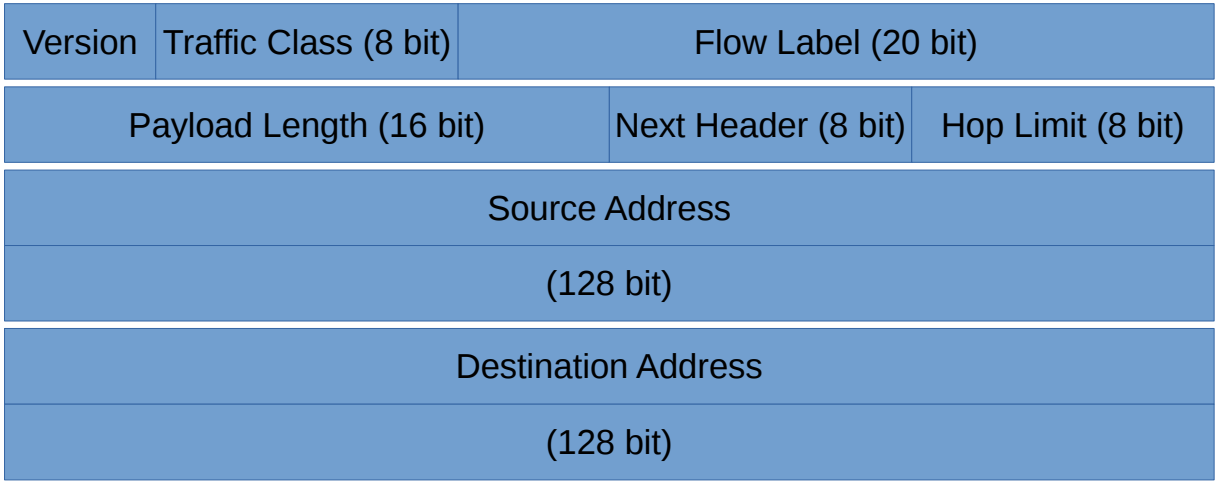
- Worst-case scenario calculations
- Maximum frame size in IEEE 802.15.4: 127 byte
- Reduced by the max. frame header (25 byte): 102 byte
- Reduced by highest link-layer security (21 byte): 81 byte
- Reduced by standard IPv6 header (40 byte): 41 byte
- Reduced by standard UDP header (8 byte): 33 byte
- This leaves only **33 byte** for actual payload
- The rest of the space is used by headers



IPv6 Header Compression (IPHC)

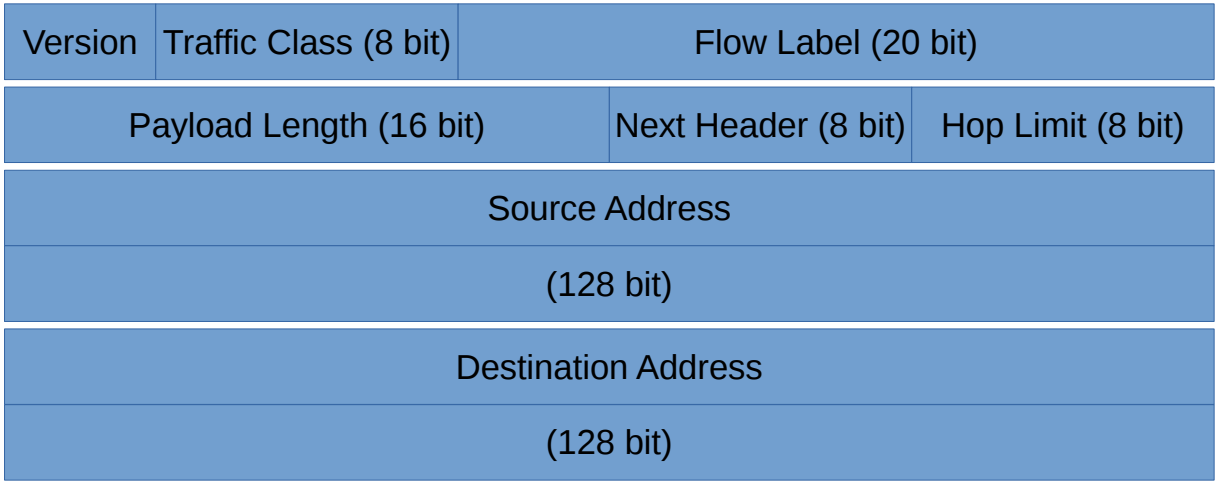
- Defining of some default values in IPv6 header
  - Version == 6, traffic class & flow-label == 0, hop-limit only well-known values (1, 64 or 255)
  - Remove the payload length (available in 6LoWPAN fragment header or data-link header)
  - Addresses (link-local, global, multicast)
- Re-usage of the L2 address for IPv6
  - Eliding the IPv6 prefix (global known by network, link-local defined by compression)
  - Using the EUI-64 L2 address
  - Using the short address in following format PAN_ID:16 bit zero:SHORT_ADDRESS

IPv6 Header (40 bytes)



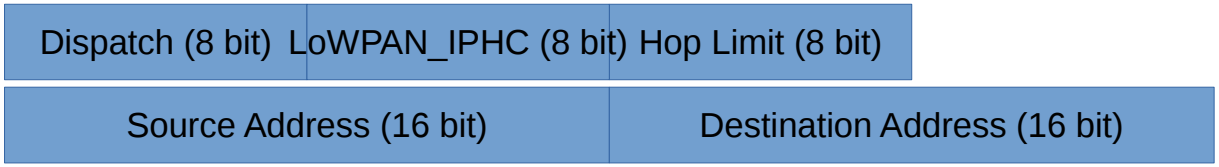
Header Comparison

IPv6 Header (40 bytes)



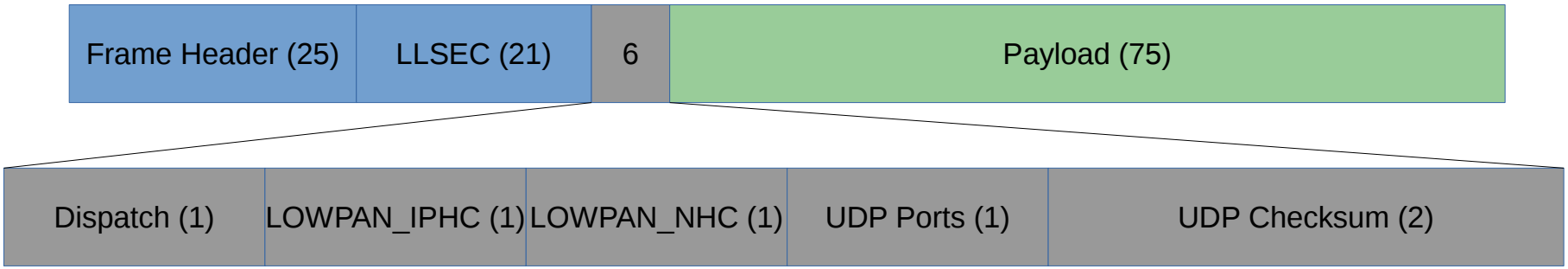
6LoWPAN Header IPHC link-local (2 bytes)

6LoWPAN Header IPHC multi-hop (7 bytes)



The Header Size Solution

- Calculations with plain 6LoWPAN usage
- IPv6 with link-local and UDP on top
- IPHC with NHC for UDP
- The fixed 48 byte IPv6 + UDP header could in the best cases be reduced to 6 bytes



Stateless Address Autoconfiguration

- Autoconfiguration based on layer 2 address
  - EUI-64 hardware address use as is
  - Pseudo 48bit address based on short address:
    16_bit_PAN:16_zero_bits:16_bit_short_address
- Link-local addresses use the FE80::/64 prefix
- Neighbor Discovery for 6LoWPAN is specified in RFC6775

6LoWPAN Header Compression

6LoWPAN Compressions

- Started with HC1 and HC2 compressions
- Updated / deprecated by IPHC and NHC
- Extended by GHC
- More NHC schemes to come, e.g. EAP
- Possible to invent your own scheme if you have repeating usage patterns in your use case

LOWPAN_HCx

- LOWPAN_HC1 and LOWPAN_HC2 came with the original adaptation layer specification (RFC4944)
- Best savings in link-local communication, e.g. neighbor discovery
- The more common case with routable addresses needed the prefix to be carried in line which makes the compression less efficient
- UDP compression support

Next Header Compression

- RFC6282
- LOWPAN_IPHC
  - Better compression for global and multicast addresses not only link-local
  - Compress header fields with common values: version, traffic class, flow label, hop-limit
- NHC IPv6 Extension Header compression
  - Hop-by-Hop
  - Routing Header
  - Fragment Header
  - Destination Options Header
  - Mobility Header
- NHC UDP Header compression
  - Compressing ports range to 4 bits
  - Allows to elide the UDP checksum for cases where upper layers handle message integrity

Generic Header Compression

- RFC7400
- A new scheme had to be defined for each new header which should be compressed
- Plugging into NHC
- Adding a vastly more general, but slightly less efficient scheme
- LZ-77 style compression with bytecode for
  - Appending zeroes
  - Backreferencing to a static dictionary
  - Copy data as is
- Indicating GHC capability over ND option 6CIO for bootstrapping

Bluetooth LE Relationship

- IETF specification for IPv6 over Bluetooth LE
- Still in draft phase (draft-ietf-6lo-btle)
- No fragmentation but usage of compression methods
- Common code is thus shared between the wpan and Bluetooth subsystems in the Linux kernel

More 6LoWPAN Adaptations

- Specifications being prepared for other L2 technologies
- NFC
- DECT Ultra Low Energy
- PowerLine (PLC)
- 802.11ah lower energy consumption, many stations, long distance, sub 1GHz
- 6loBAC: Token passing network for the RS-485 physical layer

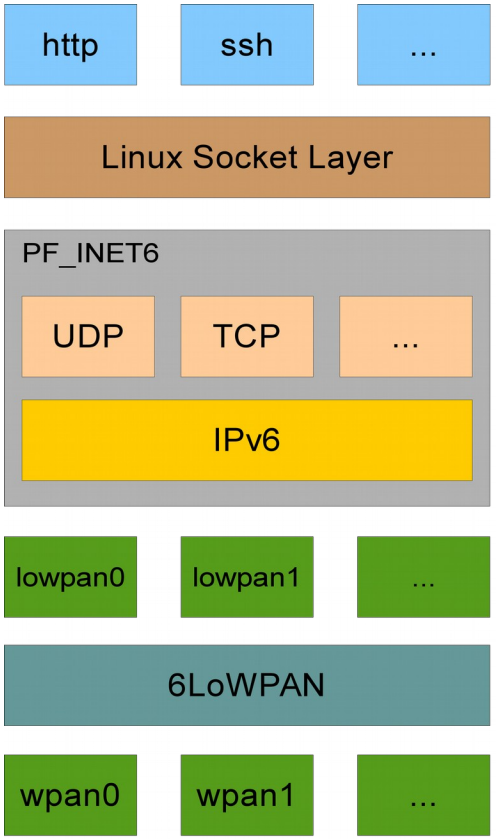
Linux-wpan Status

Project

- IEEE 802.15.4 and 6LoWPAN support in the Linux kernel
- Started in 2008 as linux-zigbee project on Sourceforge
- The first steps of mainlining around 2012
- New project name to avoid confusion: Linux-wpan
- New maintainer Alexander Aring, Pengutronix
- Mailinglist moved to vger like most other Kernel lists
- Patches are now handled on the list and picked up through the Bluetooth-next tree
- http://wpan.cakelab.org, releases, docs

Architecture

- ieee802154 handles the MAC layer and drivers (wpan0 interface)
- 6LoWPAN sits on top of the wpan devices and acts as adaptation layer to be used by the normal IPv6 kernel stack (lowpan0 interface)
- 6LoWPAN transparently handles the fragmentation and reassembly between the different MTU's (127 vs 1280) as well as compressions



Source: Alexander Aring

Current Mainline Status

- ieee802154 layer with drivers for various chips (at86rfxxx, mrf24j40, cc2520, atusb)
- Link Layer Security
- 6LoWPAN implementation
- LOWPAN_IPHC
- NHC for UDP
- GHC being worked on
- Connection between Linux devices works
- Connection to Contiki devices works
- Connection to RIOT devices works

Source: qi-hardware.com

Source: openlabs.ca

Next Header Compression

- 6LoWPAN Next Header Compression (NHC)
- Kernel framework allows for different modules to handle one compression and decrompression format each
- Mix and match different modules/formats
- Only NHC UDP is fully implemented right now
- Runtime configuration interface missing
- GHC capability indication via ND (6CIO) not yet supported

Future Work

Linux-wpan

- Implement missing parts of the spec
  - Beacon and MAC command frame support
  - Coordinator support in MAC layer and wpan-tools
  - Scan for available PANs
- Improve existing drivers and add support for new hardware
- Implement more NHC modules for other compression schemes
- Neighbor Discovery Optimization for 6LoWPAN (RFC6775)

Interoperability

- Linux-wpan most of the time tested against Linux-wpan only
- Basic tests with Contiki
  - IEEE 802.15.4 connections
  - 6LoWPAN with LOWPAN_IPHC and NHC UDP
- RIOT OS developers also tested against Linux-wpan
  - Lead to fixes on both sides
- Attending a formal Plugtest is still on the agenda
  - There is an ETSI plugtest during IETF94 in Yokohama, Japan next week
  - Hopefully some attendee with a Linux-wpan stack
  - Maybe another Plugtest in Europe next year which we might attend

Miscellaneous

- Routing Protocol for Low-Power and Lossy Networks (RFC6550)
  - unstrung, linux-rpl as current implementations
- Thread uses parts of 6LoWPAN for their protocol
  - Potential for cooperation and interop testing

References

- RFC4919: 6LoWPAN Problem Statement
- RFC4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks
- RFC6282: Compression Format for IPv6 Datagrams
- RFC6550: RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks
- RFC6775: Neighbor Discovery Optimization for 6LoWPAN
- RFC7400: 6LoWPAN-GHC: Generic Header Compression for 6LoWPAN

Thank you.

We are hiring.

jobs@osg.samsung.com