Introduction to Linux-wpan and Potential Collaboration

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

- Relationship to RIOT
- Linux-wpan Project
- Current Collaboration
- Future
Relationship to RIOT
Relationship

- Over the air protocol interoperation
  - IEEE 802.15.4 and 6LoWPAN
  - Independent implementations
  - Interoperability testing
- RIOT native on Linux
Linux-wpan Project
Motivation

- Platforms already running Linux could benefit from native 15.4 and 6LoWPAN subsystems
- 15.4 transceivers could easily be added to existing designs
- Battery powered sensors on the other hand are more likely to run a OS like RIOT
- Example: Google OneHub AP which already comes with, de-activated, 15.4 hardware
- Example: Ci40 Creator board as home IoT hub
Development Boards

- Development boards with 15.4 hardware
- Ci40 Creator (Keno should have mentioned the 6LoWPAN clickers)
- ARTIK 5/10
- Raspberry Pi with openlabs shield
- Various transceivers hooked up via SPI
- EOL atusb USB transceiver
Linux-wpan Project

- IEEE 802.15.4 and 6LoWPAN support in the mainline Linux kernel
- Started in 2008 as linux-zigbee project on SourceForge
- First steps of mainlining in 2012
- New project name to avoid confusion: linux-wpan
- New maintainer: Alexander Aring, Pengutronix
- Normal kernel development model
- Patches are posted and reviewed on the mailing list
- Accepted patches find their way through bluetooth-next towards Linus tree
Linux-wpan Community

- Small community: 1-2 core devs and ~4 additional people for specific drivers
- #linux-wpan on Freenode (~25 people)
- http://vger.kernel.org/vger-lists.html#linux-wpan (~85 people)
- https://github.com/linux-wpan (no PR model)
- http://wpan.cakelab.org used for wpan-tools releases and docs
Current Status

- ieee802154 layer with driver for various transceivers (at86rf2xx, mrf24j40, cc2520, atusb, adf7242)
- Link Layer Security
- 6LoWPAN with fragmentation and reassembly (RFC 4944)
- LOWPAN_IPHC and NHC for UDP (RFC 6282)
- Testing between Linux, RIOT and Contiki
Current Collaboration
Development

- Developer discussions about protocol interpretations
- Test counterpart for new developments e.g. neighbour discovery
- RIOT native on Linux
Testing

- Linux-wpan on the Raspberry Pi to test as boarder router for RIOT nodes
- Part of tests run before every RIOT release
- Packet sniffing
Future
raw802154

- PR #5582, currently reviewed
- With RIOT native on Linux this can be used for virtual testing on the 15.4 level
- RIOT will use a fake loop-back interface of the Linux-wpan subsystem through the raw802154 driver
- Virtual 15.4 communication between RIOT, Linux and maybe later OpenThread
raw802154 Details

- The RIOT driver will use raw packets over AF_PACKET
- The Linux loop-back interface will be used in promiscuous mode to avoid address filter problems
- Acknowledgements are not handled
- Several virtual interfaces can be created with RIOT and Linux
- Could also be used to run on a real transceiver interface, not recommended :-(
## Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Linux</th>
<th>RIOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE 802.15.4: data and ACK frames</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>IEEE 802.15.4: beacon and MAC command frames</td>
<td>✘</td>
<td>✘</td>
</tr>
<tr>
<td>IEEE 802.15.4: coordinator scenario: scanning, joining, PAN coordinator</td>
<td>✘</td>
<td>✘</td>
</tr>
<tr>
<td>IEEE 802.15.4: link layer security</td>
<td>✔️</td>
<td>✘</td>
</tr>
<tr>
<td>6LoWPAN: frame encapsulation, fragmentation, addressing (RFC 4944)</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>6LoWPAN: IP header compression (RFC 6282)</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>6LoWPAN: next header compression, UDP only (RFC 6282)</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>6LoWPAN: generic header compression (RFC 7400)</td>
<td>Partial</td>
<td>✘</td>
</tr>
<tr>
<td>6LoWPAN: neighbour discovery optimizations (RFC 6775)</td>
<td>Partial</td>
<td>✔️</td>
</tr>
<tr>
<td>Mesh link establishment draft</td>
<td>✘</td>
<td>✘</td>
</tr>
</tbody>
</table>
Linux-wpan Future

- Implement missing parts of the 15.4 specification
  - Beacon and MAC command frame support
  - Coordinator support in MAC layer and wpan-tools
  - Scanning
- Improve existing drivers and add support for new hardware
- Neighbour Discovery Optimizations (RFC 6775), started
- Evaluate running OpenThread on top of linux-wpan
RIOT Future

- Only suggestions from my side :-)  
- Link Layer Security implementation  
- Beacon and MAC command frame support  
- PAN scanning and joining  
- Support more transceivers  
- Simply hook them up to a board over SPI  
- Learn from what we had to figure out e.g. mrf24j40 problems with security enabled frames
Together

- Automate RIOT vs. Linux testing with the raw802154 driver in RIOT
- Compatible short address handling
- The mesh link establishment draft might be interesting to implement (OpenThread, ZigBee IP)
Thank you!
6LoWPAN

- Physical and MAC layer defined by IEEE 802.15.4 from 2003 onwards
- Series of IETF specifications from 2007 onwards (RFCs 4944, 6282, 7400, etc)

<table>
<thead>
<tr>
<th>L5 Application Layer</th>
<th>Application</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>L4 Transport Layer</td>
<td>TCP</td>
<td>UDP</td>
</tr>
<tr>
<td>L3 Network Layer</td>
<td>IP</td>
<td>IPv6</td>
</tr>
<tr>
<td>L2 Data Link Layer</td>
<td>Ethernet MAC</td>
<td>6LoWPAN</td>
</tr>
<tr>
<td>L1 Physical Layer</td>
<td>Ethernet PHY</td>
<td>IEEE 802.15.4 MAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IEEE 802.15.4 PHY</td>
</tr>
</tbody>
</table>
Architecture

- ieee802154 and mac802154 handle the driver and MAC layer (wpan0 interface)
- 6LoWPAN sits on top of the wpan devices and acts as an adaptation layer to be used by the normal IPv6 kernel stack (lowpan0 interface)
- The 6LoWPAN subsystem transparently handles fragmentation and reassembly between the different MTUs as well as compression