Cross-Layer Interactions in IP Wireless Sensor Networks

Andrzej Duda
Grenoble INP - Ensimag
Grenoble Informatics Laboratory

duda@imag.fr

http://duda.imag.fr
Overview

- Wireless Sensor Networks and Internet of Things
- IP protocol stack
- L2 - MAC issues: saving energy
- L3 - IP and routing
- L2 - L3 Layer interactions
- Conclusions
Wireless Sensor Networks and Internet of Things
WSN – Wireless Sensor Networks

- WSNs:
  - multiple proprietary technologies (Wavenis, DUST, EMBER, Z-WAVE, Crossbow, many others...)

sensors or actuators

connect to sink and get/send data

user

sink

Physical World
Connect WSNs to Internet

- sink = gateway
- proprietary heterogeneous L2/L3
- many gateways for each L2/L3
IETF July 2007

Most promoters of non-IP solutions have understood that IP was a MUST: they call this "IP convergence": A protocol translation gateway! Or Tunneling…

courtesy of JP Vasseur
Integration with the Internet?

- end-to-end IP
- sink = enhanced IP router
  - header compression, data aggregation, security
- nodes are IP enabled, endpoints with IP addresses
Target Applications

- Smart Grid
  - precise metering of electricity distribution
  - monitoring of energy consumption
  - acting upon devices
- Urban/Environmental Sensing
  - generic platform for sensing applications
- Environmental Sensing
  - low cost wireless communications
- Home and building automation
  - heating/air conditioning, light control
Internet of Things and IP protocol stack
Internet of Things

- Application (HTTP like)
  - CoAP (Constrained Application Protocol)

- Transport
  - lightweight, chosen functions

- Network – Routing
  - adaptation (header compression)

- MAC
  - Low Radio Duty Cycle

- PHY
  - 802.15.4, LP 802.11, Wavenis
Internet of Things

- **CoAP:**
  - simplified HTTP, RESTful (no state, asynchronous request/response)
  - point-to-point, client initiated
  - on top of UDP

- **RPL:**
  - Distance Vector routing protocol
  - builds Directed Acyclic Graph (multiple parents)

- **6LoWPAN:**
  - adaptation layer for IPv6 over IEEE 802.15.4
  - fragmentation/header compression
## Contiki (SICS)

<table>
<thead>
<tr>
<th>Layer</th>
<th>UDP</th>
<th>TCP</th>
<th>End-to-end reliability, application addressing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network (IPv6, RPL)</td>
<td></td>
<td></td>
<td>Host addressing, routing</td>
</tr>
<tr>
<td>Adaptation (6lowpan, ...)</td>
<td></td>
<td></td>
<td>Header compression, fragmentation</td>
</tr>
<tr>
<td>Medium Access Control (CSMA/CA, ...)</td>
<td></td>
<td></td>
<td>Medium access control</td>
</tr>
<tr>
<td>Radio Duty Cycling (ContikiMAC, X-MAC, ...)</td>
<td></td>
<td></td>
<td>Low-power operation</td>
</tr>
<tr>
<td>Link (802.15.4, 802.11, ...)</td>
<td></td>
<td></td>
<td>Radio connectivity</td>
</tr>
</tbody>
</table>
L2 - MAC Issues: Saving Energy
PHY Layer

- Energy consumption
  - most energy spent for radio transmission and reception!
Basic MAC access methods

- Preamble Sampling

- Scheduled Listening
ContikiMAC – X-MAC

Send data packets until ack received

Sender

Receiver

Transmission detected

Send first data packet when receiver is known to listen

Sender

Receiver

Transmission detected

Receiver
ContikiMAC - broadcast

Send data packets during entire period

Sender: D D D D D D D

Receiver: D

Transmission detected

Reception window: D

Data packet: D

To allow low-power wireless devices to actively participate in a low-power wireless network while maintaining a low power consumption, the radio transceiver must be duty-cycled. With radio duty cycling, the radio is switched off most of the time, but switched on often enough to allow the device to receive transmissions from other nodes. Over the years, many different duty cycling schemes have been designed. To illustrate the concept of duty cycling, we look at ContikiMAC, the default duty cycling mechanism in Contiki [7]. The principles of ContikiMAC is illustrated in Figure 7, Figure 8, and Figure 9. In ContikiMAC, nodes periodically wake up to check for a transmission from a neighbor. To transmit a message, the sender repeatedly transmits the packet until an acknowledgment is received from the receiver. After a successful transmission, the sender has learned the wake-up phase of the receiver, and subsequently needs to send fewer transmissions. A broadcast transmission must wake up all neighbors. The sender therefore extends the packet train for a full wake-up period.
ContikiMAC - broadcast

Figure 6: The energy consumption of the individual ContikiMAC operations.

Now be optimized to start at the expected wake-up time of the neighbor as seen in Figure 7, which shows how the number of transmissions are reduced because of the phase-lock optimization.

By computing the areas under the graphs in Figure 7 through Figure 7, we can compute the energy consumption of each operation. The result is shown in Figure 8. We see that the cost of a broadcast transmission is many orders of magnitude higher than the cost of the wake-up.

This is good: the wake-up is the most common operation in ContikiMAC—executed many times per second—and therefore should be significantly less expensive than the other operations.

Armed with the information in Figure 8, we can now compare the cost of the ContikiMAC wake-up operation with the wake-up operation of other duty cycling mechanisms. Table 1 shows the cost of a wake-up in ContikiMAC in the Contiki X0MAC implementation and the duty cycling mechanism by Hui and Culler.

4.2 Network Power Consumption

To evaluate the network power consumption of ContikiMAC and the efficiency of its optimizations, we run a set of simulations in the Contiki simulation environment.

The Contiki simulation environment consists of the Cooja network simulator and the MSPsim device emulator. MSPsim provides a cycle-accurate Tmote Sky emulation with a symbol-accurate emulation of the CC2500.
802.15.4

- TDMA controlled by a PAN Coordinator
  - **Beacons** – common time base
  - define Superframe structure
- Devices may sleep for extended periods over multiple beacons
- Slotted CSMA in beaconed PANs
- Unslotted CSMA in non-beaconed PANs
- Low duty cycle requires **beaconed enabled mode** and **slotted CSMA**
Superframe

\[ SD = a_{BaseSuperframeDuration} \times 2^{50} \text{ symbols} \]

\[ BI = a_{BaseSuperframeDuration} \times 2^{50} \text{ symbols} \]
CSMA/CA

- Backoff period: time unit=20 symbols
- BE: Backoff Exponent
- Backoff: random interval in $[0, 2^{BE}-1]$* Backoff period
- CW: Contention Window – the number of units to perform CCA (Clear Channel Assessment) after random backoff
- NB: Number of Backoffs (initial 0)
- Default values: minBE=3, maxBE=5, limit=4
Slotted CSMA/CA

- Initialization: \(NB=0, \ CW=2, \ \text{Backoff}=[0, 2^{BE}-1]\)
- Busy: \(NB=NB+1, \ CW=2, \ BE=\min[BE+1, MaxBE]\)
Contention in 802.15.4

![Graph showing PDR vs Number of nodes for 802.15.4 with and without hidden nodes.](image)
Topology in 802.15.4

- Cluster Tree
  - nodes associate with a coordinator
  - coordinator sends beacons
- Multi-hop forwarding
- Complex topology at L2 required for synchronized operation
Data pull in 802.15.4

- Data pending flag in a beacon
- Broadcast flag
  - coordinator to devices
- No broadcast from devices
Beacon Scheduling in 802.15.4

- Received Beacon
- Transmitted Beacon
- SD
- StartTime
- BI

- Incoming Active Period (received)
- Inactive
- Outgoing Active Period (transmitted)
- Inactive

Super trame N° 1
Super trame N° 2
Super trame N° 3

- Outgoing Beacon
- Incoming Beacon
- Sleep
No optimization for Upward and Downward Traffic!

- Previous slide:
  - One BO for all nodes (standard)
  - Upward Latency: BI (1 Superframe)
  - Downward Latency: N-hops * BI
IP and routing
Key IPv6 Contributions

- Large simple address (128 bits, 16 bytes)
  - Network ID + Interface ID
  - Plenty of addresses, good for Things!
  - Subnetwork has to carry at least 1280 bytes

- Autoconfiguration and Management
  - ICMPv6:
    - Neighbor Solicitation (NS)
    - Neighbor Advertisement (NA)

- Protocol options framework
  - Header extensions
**IPv6 over 802.15.4?**

- Large IP Address & Header  =>  16 bit short address / 64 bit EUID
- Minimum Transfer Unit     =>  Fragmentation
- Short range & Embedded    =>  Multiple Hops

**IPv6 packet**

- transport header
- application payload

- 40 B + options
- cls flow len hops NH src IP dst IP  
  16 B  16 B  
- net payload

**802.15.4 frame**

- ctrl len src UID dst UID  
  link payload  
- 1280 Bytes MIN
- 128 Bytes MAX

**UDP datagram or TCP segment**
IPv6 and WSN

- Addresses in WSN
  - same subnet, 0 prefix (::/64)
  - Border Router adds prefix
6LoWPAN

- IPv6 packets over 802.15.4 networks
  - 802.15.4 frame - 127 bytes, fragmentation
  - compress headers (derive addresses from L2)
  - Neighbor Discovery

6LoWPAN UDP/IPv6

<table>
<thead>
<tr>
<th>MAC hdr</th>
<th>lowpan</th>
<th>cIP</th>
<th>cUDP</th>
</tr>
</thead>
</table>

127 B Frame

108 B Payload

FCS
RPL (IPv6 Routing Protocol for Low power and Lossy Networks)

- Directed Acyclic Graph (DAG) - a directed graph with no cycles exist.
- Destination Oriented DAG (DODAG) - a DAG rooted at a single destination.
Rank

Defines a node’s relative position within a DODAG with respect to the DODAG root.
Routing messages

RPL defines a new ICMPv6 message with three possible types:

- **DAG Information Object (DIO)** - carries information that allows a node to discover an RPL Instance, learn its configuration parameters and select DODAG parents

- **DAG Information Solicitation (DIS)** - solicit a DODAG Information Object from a RPL node

- **Destination Advertisement Object (DAO)** - used to propagate destination information upwards along the DODAG.
DODAG construction

- Metrics
  - hops
  - ETX (number of retransmissions)
**Trickle Timer**

- Periodically send link-local multicast DIO messages, nodes listen to DIOs (overhearing)
- Routing inconsistencies influence the rate of DIO messages:
  - either every node that hears the message finds its data is consistent with their own state ➔ double the timer
  - a recipient detects an inconsistency ➔ reset the timer
- Consequence:
  - expects always-on nodes!
  - decoupled from beacon period!
  - link quality changes result in trickle reset
L2 – L3 layer interactions
RPL on top of 802.15.4?

- DODAG construction requires L2 connectivity for sending DIO
  - DIO provides the information about parents
  - association with a parent at L2 should be done before!
  - tree constructed before at L2
- DODAG construction on a Tree = Tree
- Parent choice based on DIO reception – L3 metric (no check for a symmetric link)
- DIO message may exceed the available 79 octets causing fragmentation
- Trickle ineffective because of beaconed mode
Need for cross-layer

- Node needs to associate with a coordinator at L2
  - it cannot send any data frame before association
- Node waits for beacons, but it may only make the association decision based on the L3 information coming from RPL

```
RPL

connection

PHY
```

Neighbor Discovery

MAC

in sync

topology information
Possible scheme for topology construction

L2 \rightarrow L3 : request information (implicit DIS)

Coord.1
Coord.2
Coord.3
RFD 1

scan duration

Making a choice at L3

L2\rightarrow L3 : request association

L2-L3 joint management
collecting DIO information to choose L3 network
associate to the corresponding parent at L2

Collecting neighborhood information to choose parent

L2 \rightarrow L3 : request information

Coord.1
Coord.2
Coord.3
RFD 1

beacon request used to trigger DIO transmission
DIS is not possible since no L2 available yet

802.15.4 MAC beacon-enabled

MAC management

RPL Beacon Request

7.5.2.1.2 Active channel scan
If a coordinator of a beacon-enabled PAN receives the beacon request command, it shall ignore the command and continue transmitting its periodic beacons as usual. If a coordinator of a nonbeacon-enabled PAN receives this command, it shall transmit a single beacon frame using unslotted CSMA-CA.

41
RPL on top of ContikiMAC?

- Expensive broadcasts, needs data to keep sync
- Convergence to low Duty Cycle based on Trickle
  - works if no dynamic changes (not realistic)
- Good - no topology binding at L2

![Number of trickle interval resets vs Node ID](chart.png)
RPL on top of 6LoWPAN?

- Neighbor Discovery? – Link Maintenance already done at MAC/PHY layer
- Neighbor Solicitation/Advertisement
  - needed for mapping L2 addresses to L3
  - are NS/NA really necessary?
    - timeouts before next beacon and active period?
    - DAD not mandatory as addresses are based on EUI-64
    - L2 address already known from L3
- No need for using IPv6 Neighbor Discovery (no NS/NA)
Conclusions

- Future Internet of Things
  - relies on IP networking
  - promising approach to unifying sensor networks

- Still many issues to address
  - topology construction: how to choose a parent and associate at L2? Topology at L2 vs. L3
  - better link metric estimation (take into account asymmetric links)
  - neighborhood maintenance coupled with duty cycled L2

- Upper layer issues
  - transport (TCP) over duty cycled networks
Questions?
# Size of Super Frame

<table>
<thead>
<tr>
<th>SO</th>
<th>Size of Slot (symbols)</th>
<th>SD duration 2,4/2,485 GHz</th>
<th>SD duration 902/928 MHz</th>
<th>SD duration 868/868,6 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>60</td>
<td>15,36 ms</td>
<td>24 ms</td>
<td>48 ms</td>
</tr>
<tr>
<td>1</td>
<td>120</td>
<td>30,72 ms</td>
<td>48 ms</td>
<td>96 ms</td>
</tr>
<tr>
<td>2</td>
<td>240</td>
<td>61,44 ms</td>
<td>96 ms</td>
<td>192 ms</td>
</tr>
<tr>
<td>3</td>
<td>480</td>
<td>122.88 ms</td>
<td>192 ms</td>
<td>384 ms</td>
</tr>
<tr>
<td>4</td>
<td>960</td>
<td>245.76 ms</td>
<td>384 ms</td>
<td>768 ms</td>
</tr>
<tr>
<td>14</td>
<td>983040</td>
<td>251,6 s</td>
<td>393,2 s</td>
<td>786,4 s</td>
</tr>
</tbody>
</table>
Slotted CSMA/CA

1. **NB=0, CW=2, BE=minBE**
2. **R=rand[0, 2^BE - 1]**
3. Locate backoff period boundary
4. Backoff for R units
5. **CCA on backoff period boundary**
6. **Busy**
7. **NB++, CW=2, BE=min(BE+1, maxBE)**
8. **NB>limit?**
   - No
   - Yes: **Failure**
   - Yes
   - Idle
8. **CW=CW-1**
9. **CW=0?**
   - No
   - Yes: **Success**
Constrained Application Protocol (CoAP)

- RESTful Web services for networked embedded devices
  - Idealized architectural style of the Web
  - HTTP for the Internet of Things

GET /sensors/temperature
CoAP on top of RPL?

- Needs Downward routes
  - DAO from leaves to sink
  - standard does not specify when to send DAO
- Storing nodes
  - routing table – one entry per IP address (ad hoc)
- Multiple sinks?
  - multi-homing problem for a sensor node