Software Defined Wireless Networks (SDWN): Unbridling SDNs

Salvatore Costanzo, Laura Galluccio, Giacomo Morabito, Sergio Palazzo

Dipartimento di Ingegneria Elettrica, Elettronica, e Informatica
University of Catania

October 25, 2012

Outline

Introduction and motivations

Motivations and contribution

- Increasing number of wireless and mobile communications companies are investing in SDN.
  - Examples: Verizon, Nokia Siemens Networks, Ericsson, and Netgear are members of OpenNetworking Foundation.
- However, large effort is still required:
  - what are the advantages of SDN in the most common wireless networking scenarios?
  - how should the SDN concept be expanded to suit the characteristics of wireless and mobile communications?
- In this paper we
  - analyze the advantages of the SDN approach in low rate wireless personal area networks (LR-WPAN)s. Example: IEEE 802.15.4
  - identify the differences in requirements between traditional wired networks and LR-WPANs.
  - present **Software Defined Wireless Network (SDWN)** –
    - ⋆ partially implemented

SDN in the wireless domain

Advantages of SDN in LR-WPANs

- SDN is about simplification and evolvability
  - This applies to any networking environment
- Almost universal consensus on the first two layers of the protocol stack – *IEEE 802.15.4*, but several alternative candidates – often incompatible – for the higher layers
  - For example: 6LOWPAN vs. ZigBee
  - Note: *Same discussions valid for other relevant domains: vehicular networks and mesh networks, for example*

  $\rightarrow$ difficult for a node to move from one network to another

  SDN solves this problem allowing to change the behavior of a node *on the fly*
- In most implementations SDN applies a centralize management – through the controller

  $\rightarrow$ easier optimization in the use of network resources
  - $\rightarrow$ crucial problems: which element(s) run the network control operations, how to communicate with such element(s)

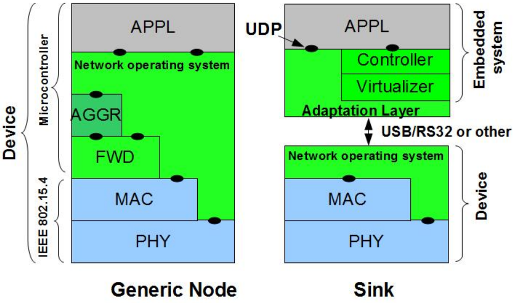
Requirements for SDN solutions in LR-WPANs

- In traditional wired networks: velocity major performance measure → rigid definition of rules
- In LR-WPANs scenarios priorities change: **energy efficiency** primary requirement for a Software Defined Wireless Network (SDWN) solution
- Accordingly, SDWN must support
  - **Duty cycles**: An appropriate *action*[1] must be defined
  - **In-network data aggregation:** An appropriate module must be introduced in the protocol architecture and a new action must be defined
  - **flexible definition of the rules**: this is achieved in two ways:
    - Any byte of the packet can be considered by the rule
    - Other relation operators (not just equivalence) can be considered for the matching of rules
  - Other **obvious** requirements: for example, support of node mobility, deal with link unreliability, robustness to the failure of generic nodes and the control node

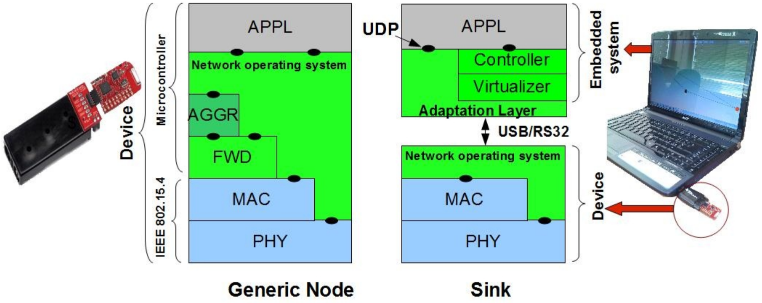
[1] We use the OpenFlow approach and notations

Software Defined Wireless Networks (SDWN)

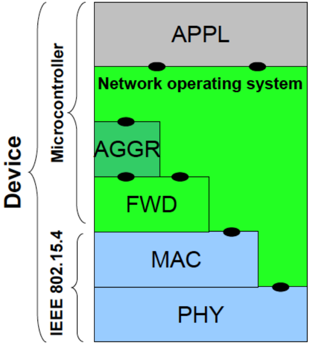
Architecture



Architecture - Example of our implementation



Generic node



Generic Node

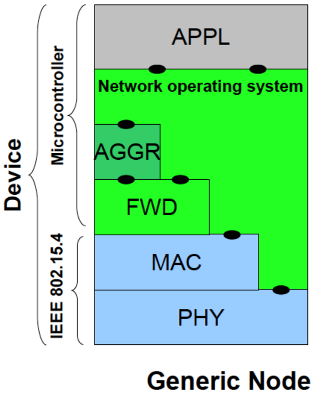
- Standard IEEE 802.15.4 (layers 1 and 2)
- **Forwarding layer**: Treats arriving packets as specified in the *flow table*...
  - *Control packet*: given to the Networking Operating System (NOS)
  - *Data packet*: Two cases:
    - Match in the flow table → treat accordingly
    - No match → sent to the NOS
- **Aggregation layer**: Aggregates information flowing through the network. Currently,
  - One of the possible actions is to include the packet in an *aggregation equivalent flow* (AEF)
  - Concatenates packets of the same AEF
- **Network Operating System (NOS) layer**:
  - Collects and sends local information to the Controller
  - Controls the behavior of all layers as specified by the Controller

Flow table and flow table entries

- A flow table is composed by several *flow table entries*
- A flow table entry contains the following information
  - A rule: description of the characteristics of all packets belonging to a *flow* and that must be treated by the node in the same way
  - An action: operation be executed to all packets of the flow
  - Some statistic information: number of packets received for the flow

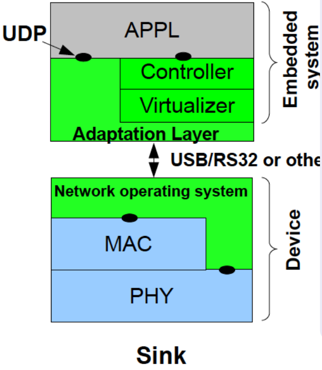
Further details later...

Generic node



- Standard IEEE 802.15.4 (layers 1 and 2)
- **Forwarding layer**: ttreats arriving packets as specified in the *flow table*...
  - *Control packet*: given to the Networking Operating System (NOS)
  - *Data packet*: Two cases:
    - Match in the flow table → treat accordingly
    - No match → sent to the NOS
- **Aggregation layer**: Aggregates information flowing through the network. Currently,
  - One of the possible actions is to include the packet in an *aggregation equivalent flow* (AEF)
  - Concatenates packets of the same AEF
- **Network Operating System (NOS) layer**:
  - Collects and sends local information to the Controller
  - Controls the behavior of all layers as specified by the Controller

Sink node



Embedded system

- **Adaptation layer**: formats messages as needed by the device
- **Virtualizer**: uses the local information collected by the generic nodes to build a consistent representation of network state
  - Topology, nodes battery level, link quality, etc
  - Currently it is implemented as a Java map

  The Virtualizer allows several logical networks to run over one physical network.
- **Controller(s)**: implements the network management policy desired

  In our implementation Controllers interact with the Virtualizer through a UDP socket

Device

Same as a generic node...

Some design and implementation details

Collection of topology information

- NOS in generic nodes need to communicate with the NOS in the sink → *how reach the sink without the controller intervention?*
  - The sink periodically generates a beacon packet, which is broadcast in the network
  - The beacon packet contains information about the current distance (number of hops) from the sink and the battery level of the last relay node
  - Generic nodes use information in the beacon to decide the most convenient next hop to reach the sink
- Information contained in the beacon is used to build the neighbors table
  - It contains the list of neighbors and the corresponding RSSI values
  - It is periodically sent to the sink through a report packet
- The sink receives the neighbor tables by all nodes and infers the global network topology (at the *Virtualizer* layer)

Specification of rules and actions

- In our scenario
  - Major goal: flexibility
  - Major limitation: small memory
- A rule operates on (up to) three windows of 1-2 bytes of the incoming packets
- Each flow table entry contains three group of blocks:
  1. Window blocks: related to the three windows
     - For each window: size, operator, position of the first byte of the window, value
  2. Action block: two fields *Action type* (e.g., forward, modify, drop, aggregate, turn off radio) and the *Action value* (meaning depends on the type of action)
  3. Counter: number of packets received so far satisfying the rule

Window I				**Window II**				**...**	**Action**		**Stats**
Size	*Op*	*Addr*	*Value*	*Size*	*Op*	*Addr*	*Value*	...	*Type*	*Value*	*Count*
2	=	2	170.24	2	≠	4	170.11	..	Forward	170.23	17
2	=	2	170.16	1	=	1	3	..	Drop	1	3
2	≠	2	170.24	1	=	7	25	..	Modify	7/26	3
2	=	2	170.17	0	=	0	0	..	Forward	170.21	11

Figure: Exemplary table.

- Example: consider Entry 1
  - if bytes 2 and 3 are = 170 and 24 and bytes 3 and 4 are ≠ 170 and 11 → forward to node 170.23
  - 17 packets have been classified with this rule

Packet format

byte	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	Packet Length								Network ID							
2	Source Address															
4	Destination Address															
6	Type of Packet								Time To Live							
8	Address Next Hop															

Entry 1: All packets generated by node 170.24 and that are not directed to node 170.11 must be forwarded to node 170.23.

- Packet types:
  - **Type 0 - Data packet**: packet generated by the application layer
  - **Type 1 - Beacon packet**: broadcast periodically by the sink
  - **Type 2 - Report packet**: generated periodically by generic nodes and sent to the sink
  - **Type 3 - Rule/action request**: generated by a generic node and sent to the sink upon receiving a packet not matching with any flow table entry
  - **Rule/action Response**: Generated by the sink in response to a Rule/action request

An empiric assessment

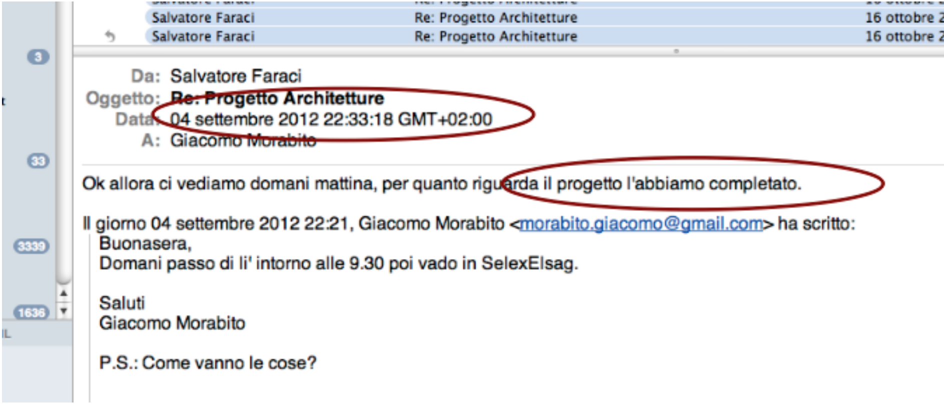
Challenging three students

- A simple assignment: write the SDWN controller for a sensor network imposing the following policy
  - If the payload value is lower than x deliver the packet to node A
  - If the payload value is higher than x deliver the packet to node B
  - If the payload value is equal to x deliver the packet to B but avoid routes passing through C
- The challengers: *Good* Computer Engineering MSc students
  - Two lessons about SDN and OpenFlow
  - Good Java programming skills
- The rule: complete the assignment within 24 hours $\rightarrow$
  - *maximum vote* (i.e., 30/30), *no further exam*!

Challenging three students

- A simple assignment: write the SDWN controller for a sensor network imposing the following policy
  - If the payload value is lower than x deliver the packet to node A
  - If the payload value is higher than x deliver the packet to node B
  - If the payload value is equal to x deliver the packet to B but avoid routes passing through C
- The challengers: *Good* Computer Engineering MSc students
  - Two lessons about SDN and OpenFlow
  - Good Java programming skills
- The rule: complete the assignment within 24 hours $\rightarrow$
  - *maximum vote* (i.e., 30/30), *no further exam*!

Started on September 4, 2012 at 9.30 and...!



Salvatore Faraci	Re: Progetto Architetture	16 ottobre 2
Salvatore Faraci	Re: Progetto Architetture	16 ottobre 2

Da: Salvatore Faraci
Oggetto: **Re: Progetto Architetture**
Data: 04 settembre 2012 22:33:18 GMT+02:00
A: Giacomo Morabito

Ok allora ci vediamo domani mattina, per quanto riguarda il progetto l'abbiamo completato.

Il giorno 04 settembre 2012 22:21, Giacomo Morabito <morabito.giacomo@gmail.com> ha scritto:

> Buonasera,
> Domani passo di li' intorno alle 9.30 poi vado in SelexElsag.
>
> Saluti
> Giacomo Morabito
>
> P.S.: Come vanno le cose?

Challenging three students

- A simple assignment: write the SDWN controller for a sensor network imposing the following policy
  - If the payload value is lower than x deliver the packet to node A
  - If the payload value is higher than x deliver the packet to node B
  - If the payload value is equal to x deliver the packet to B but avoid routes passing through C
- The challengers: *Good* Computer Engineering MSc students
  - Two lessons about SDN and OpenFlow
  - Good Java programming skills
- The rule: complete the assignment within 24 hours $\rightarrow$
  - *maximum vote* (i.e., 30/30), *no further exam*!

Started on September 4, 2012 at 9.30 am and completed around 6.15 pm of the same day!

Conclusions and future work

Conclusions and future work

Contributions

- First attempt to analyze the opportunities and challenges of SDN in LR-WPANs
- Definition and implementation of a prototype

Future work

- Robustness against sink failure (multi-sink architectures?)
- Performance evaluation through simulation or experimentation
- Optimal setting of some protocol parameters (for example, size of tables, period of beacon and report generation, etc.)
- Implementation of a simulator of the generic nodes and interaction with a real controller
  - Extension to the the hardware-in-the-loop simulation approach