A WSN middleware for security and localization services

Speaker
Marco Santic

Center of Excellence DEWS
University of L'Aquila
Italy
Overview

→ Introduction
  ● Concept of a Middleware for WSN
  ● Useful services for a WSN-MW

→ LightWeight Localization

→ Security services

→ Agilla2: a renewed MW

→ Conclusions
Introduction
A Middleware (MW) is a software layer ideally located between the operating system and user-level applications.

- Its main goal is to provide advanced services (with respect to the OS ones) to upper layers.
- These services usually allow applications to communicate without taking care of the underlying HW and SW heterogeneity.
- In this way, developers can focus on application logic without worrying (too much) about architecture-specific details.
Moreover, the use of a MW normally allows an effective way to maintain and integrate distributed applications

- In the Wireless Sensor Networks (WSN) field, a MW is meant to give a network-oriented view to the developers, providing architecture and topology oriented high level API
- This allows to integrate the WSN in complex systems and eventually to handle more WSNs as a single entity

Advanced services
- Localization and security services, built on modules, can be provided to upper layers and enrich the high level API
Lightweight Localization
Lightweight Localization 1/5

Motivations

- Localization often requires a considerable amount of time and efforts in deployment and setup of parameters
  - It is desirable an easy deployable algorithm
- Often techniques and approaches proposed in literature require a heavy computational effort
  - It is desirable a computationally light solution, usable on resource limited devices
- Complex mathematical approaches are belittled if there is an underlying RSSI-based ranging approach
  - Simple operations are desirable, avoiding floating point operations

Goal

- Lightweight approach, feasible for a WSN node, accepting lower accuracy as trade off
Proposed solution

- RSSI-based
  - No extra hardware needed

- Handset based (blind node localizes itself)
  - Beacon signals from anchor nodes (sending coordinates), on board data processing

- The algorithm performs an exploration of a grid of candidate positions
  - It can also work along single axes to reduce the number of candidate positions (especially in 3D)

- Evaluation of a quality metric derived from signal propagation model

- Lower accuracy, but simple operations and faster execution
Simulations

- Scenarios
  - 4 and 9 anchors
  - 10m x 10m
  - +/- 20% search area
- Grid steps
  - 0.1m – 0.25m – 0.5m

- Chosen random position O
- Supposed position X
- Quality metric representation
Validation with real data

- Some simple indoor and outdoor scenarios (different dims.)
- Data collected in real scenario (Ambient Assisted Living prj.)
- First implementation in C language on PC
Implementation on real devices

- Application for TinyOS2
- Preliminary tests on Memsic MicaZ (indoor and outdoor)
Security services
Security services

➲ Motivations

- It is desirable, in WSN for monitoring, sensing and actuation:
  - data integrity,
  - authentication,
  - secure transmissions

➲ Solutions

- Secure Cryptographic Scheme, light in computation and aimed to resource constrained devices
- Intrusion Detection System
Topological Authenticated Key Scheme (TAKS)

- Cryptographic scheme that can offer passive security at Data Link layer
- Hybrid approach for the generation and distribution of keys among the nodes of the WSN
- The keys are function of the network topology (the nodes belong to)
- Every node has a Local Key (private role) and a Transmitted Key (public role)
- Possible secure key generation for point-to-point (pair-wise) or point-to-multipoint (cluster-wise) communication
Every node has then a Local Configuration Data (LCD)
- Local Key Component (LKC)
- Transmitted Key Component (TKC)
- Local Planned Topology (LPT) containing Topology Vectors (TV)

These data are used in encryption/decription scheme
An Intrusion Detection System (WIDS)

- Based on Weak Process Model, used to model possible threats

- Realized a TinyOS component integrated with 802.15.4 stack
Agilla as Mobile-Agent based MW for WSN
Agilla as Agent-Based MW for WSN

Mobile-Agent based MW

- A Mobile-Agent is an object, composed by code and some supporting data structure, that can move (migrate) among different nodes of the network.

Mobile-Agent based technologies are subject of many studies and thus they are in constant evolution.

- Every day, issues like power consumption, flexibility and portability are faced with improved methodologies, granting better performances.
- An agent-based MW allows us to achieve better resilience (by adding code and data redundancy) and scalability (by offering a dynamic way to remotely program new devices).
- It is the only MW category that allows WSN reconfiguration at run-time without loosing continuity of service.
In such a MW domain, after a detailed comparison, Agilla resulted the most suitable one. Other MW have been discarded mainly due to the limitations on multitasking and target platforms.

- So, Agilla has been selected for some European research projects and ported to TinyOS 2.x to fully exploit new features and platforms.
The goal of the work was to port the original Agilla MW to a new version of TinyOS

- Since we are targeting TinyOS 2.x we have called it Agilla2

In order to support such a porting and, more in general, to allow future maintenance, a prior modeling activity has been considered of critical importance

- In fact, the availability of a human-readable model of Agilla is quite a must for a complete overview of Agilla inner architecture and a valuable help during the porting
- Also, such a model could be useful in future, for maintainability and further improvements to Agilla itself
- So, the modeling of Agilla inner components has been performed by using UML and MagicDraw
Then, to get a TinyOS 2.x compatible version of Agilla, a full porting of the original Agilla source code was needed

- The first step has been to identify the main differences between TinyOS 1.x and TinyOS 2.x and to define the strategies to be used to make Agilla compatible with the latter

- Main differences between TinyOS versions
  - Tasks
  - Booting sequence
  - Timers
  - Communications
  - Error Codes
The validation has been based on several famous Agilla applications (e.g. Oscilloscope) and some custom ones.

- The goal has been to run such applications correctly on Agilla2, using all the functionalities previously analyzed, such as sensor readings, communications, and migration of agents.

- To achieve this result, we installed Agilla2 on the sensor nodes (MicaZ and Iris) and then we injected the proper agents through the AgentInjector, which correctly started to retrieve and forward data.
Conclusions
Results

- We presented an innovative **approach for a computationally lightweight localization**; the approach has been firstly analysed by means of simulations, then it has been validated by means of the execution with real data and by *implementing it on real devices*

- A **cryptographic scheme** providing security modules has been developed (**TAKS**). It has been implemented and its performances benchmarked

- It has been realized an **IDS multi-platform in TinyOS2**, independent from the stack implementation, that can be used in 802.15.4 beacon-enabled networks. The IDS has been introduced in Agilla(2) and **agents that use it has been tested**

- The **Agilla porting** to TinyOS 2.x has been performed in the context of some European research projects (ERC VISION and ECSEL SAFECOP). The porting has been successfully completed and the new code, called **Agilla2**, is online available at DEWS website
Work in Progress: localization and security services in Agilla2

- Currently we are working on a localization and security-oriented extension for Agilla2, in order to obtain a MW able to provide proper mechanisms for simple localization and to support data integrity, authentication techniques and secure transmissions.

Future works

- A new MW will be developed as a re-design/re-factoring of Agilla2, to minimize the overheads introduced by some services, to reduce the final footprint of the code, in order to have more space for different communication protocols (e.g. IEEE 802.15.4(e), OpenZB, TinyAODV, 6LowPAN, etc.) and the application layer, and to reduce energy consumption.
References


- A Middleware Approach for IEEE 802.15.4 Wireless Sensor Networks Security; Stefano Marchesani, Luigi Pomante, Marco Pugliese, Fortunato Santucci; EAI Endorsed Transactions on Ubiquitous Environments, Volume 2, Number 5, July 2015

- A Renovated mobile agents middleware for WSN: Porting of Agilla to the TinyOS2.x Platform; Luigi Pomante, Marco, Walter Tiberti, Stefano Marchesani, Lorenzo Corradetti and Daniele Gregori (University of L’Aquila, Italy); RTSI 2016, Bologna
Thank you!