Abstract

Nowadays, Internet of things is witnessing a tremendous evolution due to the increasing growth in communication technologies, weather and environmental sensing, health care sensing. Indeed, sensors are being a kind of intelligent mobile agent able to perceive its environment and transmit information to the cloud for processing, This way of perception allow the development of several kinds of applications to enhance human capacity to understand their environment and make appropriate decision. However, developing such advanced applications relies heavily on the quality of the communication between sensors and between sensors and the infrastructure, therefore, such communication can be realized only with the help of a secure data collection and efficient data treatment and analysis.

Data collection in a vehicular network has been always a real challenge due to the specific characteristics of these highly dynamic networks (frequent changing topology, vehicles speed and frequent fragmentation), which lead to opportunistic and non long-lasting communications. Security, remains another weak aspect in these wireless networks since they are by nature vulnerable to various kinds of attacks aiming to falsify collected data and affect their integrity. Furthermore, collected data are not understandable by themselves and could not be interpreted and understood if directly shown to a driver or sent to other nodes in the network. They should be treated and analyzed to extract meaningful features and information to develop reliable applications. In addition, developed applications always have different requirements regarding quality of service (QoS). Several research investigations and projects have been conducted to overcome the aforementioned challenges. However, they still did not meet perfection and suffer from some weaknesses. For this reason, we focus our efforts during this thesis to develop a platform for a secure and efficient data collection and exploitation to provide vehicular network users with efficient applications to ease their travel with protected and available connectivity. Therefore, we first propose a solution to deploy an optimized number of data harvesters to collect data from an urban area. Then, we propose a new secure intersection based routing protocol to relay data to a destination in a secure manner based on a monitoring architecture able to detect and evict malicious vehicles. This protocol is after that enhanced with a new intrusion detection and prevention mechanism to decrease the vulnerability window and detect attackers before they persist their attacks using Kalman filter. In a second part of this thesis, we concentrate on the exploitation of collected data by developing an application able to calculate the most economic itinerary in a refined manner for drivers and fleet management companies. This solution is based on some information that may affect fuel consumption, which are provided by vehicles and other sources in Internet accessible via specific APIs, and targets to economize money and time. Finally, a spatio-temporal mechanism allowing to choose the best available communication medium is developed. This latter is based on fuzzy logic to assess a smooth and seamless handover, and considers collected information from the network, users and applications to preserve high quality of service.

Résumé

ⵖⵉⴽⴽⴰⴷ

ⵖⵉⴽⴽⴰⴷ ⵙ ⵎⵔⴰⵡ ⵉⵙⴳⴳⵯⴰⵙⵏ ⵔⴰⴷ ⵢⵓⵜⵉ ⵍⵍⴱⴰⵙⵜⵉⴽ ⵉⵙⵍⵎⴰⵏ ⵖ ⵢⵉⵍⵍ

ملخص

الأفكار الخضراء عديمة اللون تنام بغضب

Acknowledgements

Dedication

Contents

List of Tables

List of Figures

List of Abbreviations

List of Nomenclatures

BS Base Station

CCI Co-channel Interference

DC Duty cycle

Jit Jitter

Mob Mobility

PDR Packet delivery ratio

PLR Packet loss rate

PS Payload size

RTT Round time trip

SC Service Cost

SINR Signal-to-interference & noise ratio

SIR Signal-to-Interference Ratio

SL Sleep time

TC Traffic congestion

Th Throughput

ToA Time on Air

Tx Transmission Energy

List of Publications

"There's no absolutely reliable way to achieve a great citation. However, hardworking could be fruitful" - Eraldo Banovac

International Conferences

National Conferences

Journals

Survey

Part I

IoT

1 | Introduction

"The secret of a good sermon is to have a good beginning and a good ending, then having the two as close as possible" - George Burns

Contents

1.1 Context & motivation

The exponential growth of 5G networks and the development of IoT that will greatly come with it, would considerably raise the number of Smart Cities applications. The aim of such technology is mainly to improve the comfort and the safety of users through wireless IoT networks. Wireless Sensor Networks (WSNs) are the source of sensed data of cities things, i.e. roads, cars, pedestrians, houses, parking, etc. The cloud is the entity that collects the sensed data and allows users and machines to do data analysis and improve services. For Smart Cities, one objective is improving the welfare of citizens as well as its safety getting real-time information about the city infrastructure. One application would be the transportation systems, and traffic lights control having as an objective avoids congestion and dangerous situations. A static cycle of traffic lights has a direct impact on traffic jams. The long period at red or green light could impact the fluidity of the city traffic. The Internet of Things (IoT) would give an answer to the required interoperability between heterogeneous wireless networks. Our objective is to model, prototype and evaluate a traffic control system. Indeed, different infrastructures have different purposes and technologies, this means that it is not possible to state communication between two infrastructures following a Device-to-Device approach. However, thinking of an indirect or Device-to-Cloud communication between infrastructures seems useful when every connected system has its own technologies, e.g. Zigbee, LoRa, SigFox, ITS-G5. Consequently, IP stack would be the suitable mediator for interconnecting these networks. It removes the barriers of rigid standard specifications of the hardware despite the overhead of the extra network configuration. Furthermore, we want to have a scalable solution not limited only on the traffic light management system. We can deploy sensors and actuators to measure noise or air pollution via panels or roads and offer new services, e.g. where and when jogging is better. To implement our Urban Traffic Light Control based on an IoT network architecture (IoT-UTLC), we setting a real IEEE 802.15.4 WSN devices that would act as actuators and sensors. All these small traffic light devices are driven by a Border Router (BR) which is a gateway to the Internet. This BR

1.2 Methodology and contributions

1.3 Organization of the thesis

2 | State of the art

"Given one hour to save the planet, I would spend 59 minutes understanding the problem and one minute resolving it." - Albert Einstein

Contents

2.1 IoT Applications requirements [1]

2.1.1 Transportation and logistics

2.1.2 Healthcare

2.1.3 Smart environnement

2.1.4 personal and social

2.1.5 Futuristic

Use cases			
Health Monitoring			
Water Distribution			
Electricity Distribution			
Smart Buildings			
Intelligent Transportation			
Surveillance			
Environmental Monitoring			

Table 2.1: Use cases [**hancke_role_2012**]

Callenges-Applications	Gids	EHealth	Transportations	Cities	**Building**
Ressources cinstraints	+	+++	-	++	+
Mobility	+	++	+++	+++	-
Heterogeneity	++	++	++	+++	+
Scalability	+++	++	+++	+++	++
QoS cinstraints	++	++	+++	+++	+++
Data management	++	+	+++	+++	++
Lack of standardization	++	++	++	++	+++
Amount of attacks	+	+	+++	+++	+++
Safety	++	++	+++	++	+++

Table 2.2: Main IoT challenges[2] + [3]

voir [4]

Smart systems in smart cities [6]

- Smart Mobility
- Smart semaphores controle
- Smart Red Swarm
- Smart panels
- Smart bus scheduling

Figure 1: Use cases.

Use Case	Packet rate () [packet/day]	Minimum success rate (Ps,min)	Grouping
Wearables	10	90	
Smoke Detectors	2	90	
Smart Grid	10	90	Group A PL = 10/20B
White Goods	3	90	
Waste Management	24	90	
VIP/Pet Tracking	48	90	
Smart Bicycle	192	90	
Animal Tracking	100	90	
Environmental Monitoring	5	90	
Asset Tracking	100	90	Group B PL = 50B
Smart Parking	60	90	
Alarms/Actuators	5	90	
Home Automation	5	90	
Machinery Control	100	90	
Water/Gas Metering	8	90	
Environmental Data Collection	24	90	
Medical Assisted Living	8	90	
Microgeneration	2	90	
Safety Monitoring	2	90	Group C PL = 100/200B
Propane Tank Monitoring	2	90	
Stationary Monitoring	4	90	
Urban Lighting	5	90	
Vending Machines Payment	100	90	
Vending Machines General	1	90	Group D PL = 1KB

Table 2.3: A PPLICATION REQUIREMENTS FOR THE USE CASES OF INTEREST[5] [3].

- Smart EV management
- Smart surface parking
- Smart signs
- Smart energy systems
- Smart lighting
- Smart water jet systems
- Smart residuals gathering

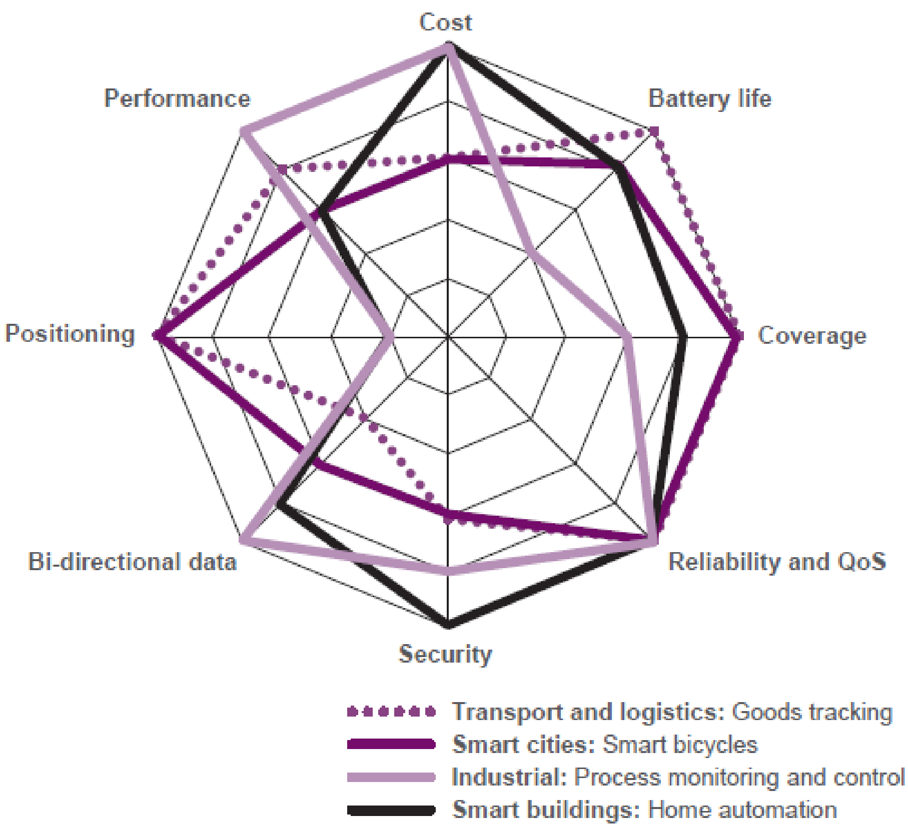


Figure 2: Use cases.

- Smart building construction
- Smart tourism
- Smart QRinfo
- Smart monitoring
- Smart hawkeye

2.1.6 Summary and discussion

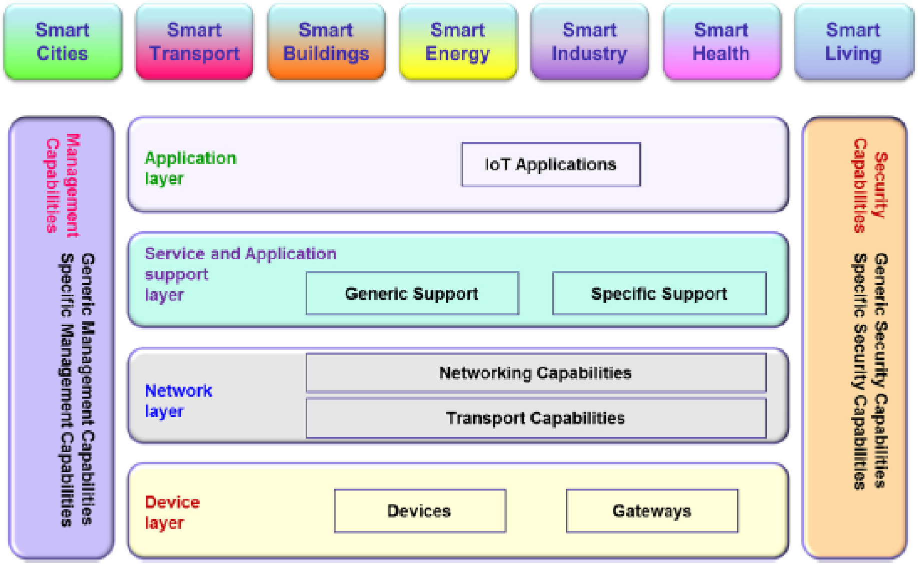


Figure 3: Use cases.

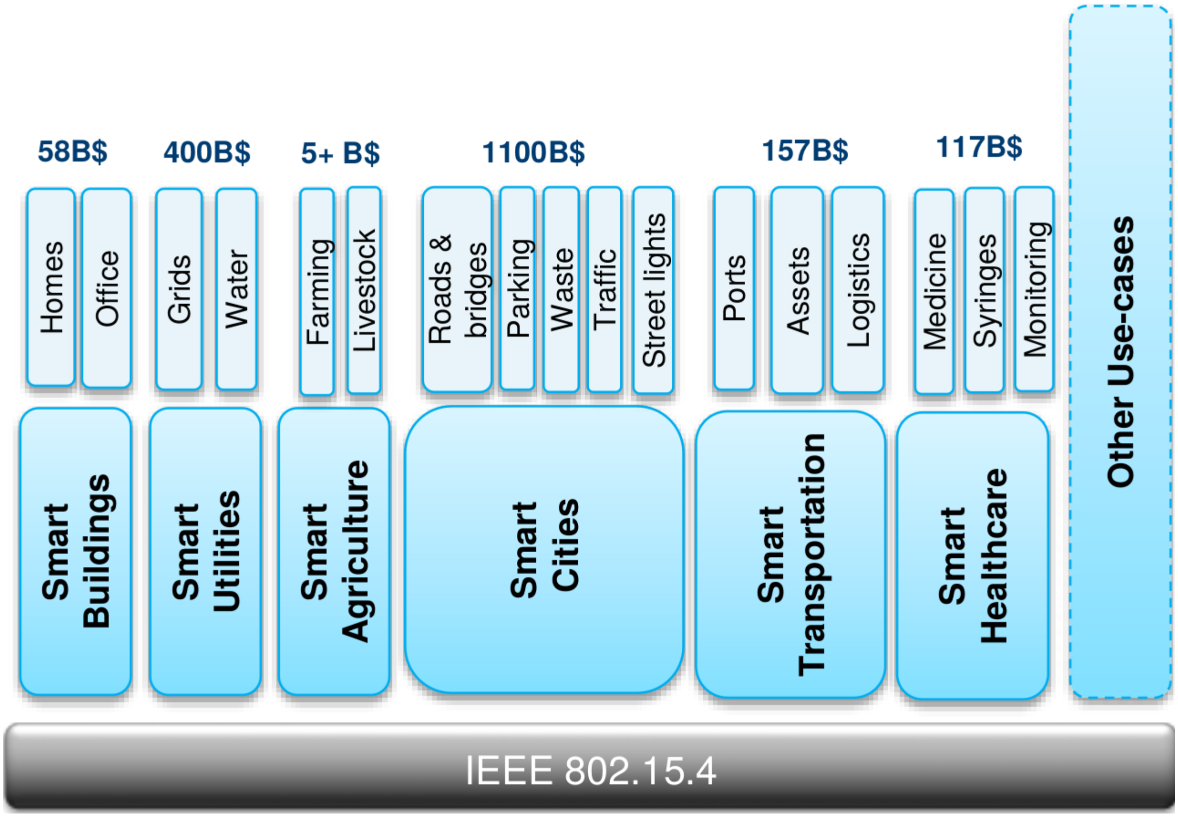


Figure 4: 802.15.4 use cases [**sarwar_iot_**].

2.2 IoT Sensors

2.2.1 OS platform

The operating system is the foundation of the IoT technology as it provides the functions for the connectivity between the nodes. However, different types of nodes need different levels of OS complexity; a passive node generally only needs the communication stack and is not in need of any threading capabilities, as the program can handle all logic in one function. Active nodes and border routers need to have a much more complex OS, as they need to be able to handle several running threads or processes, e.g. routing, data col-

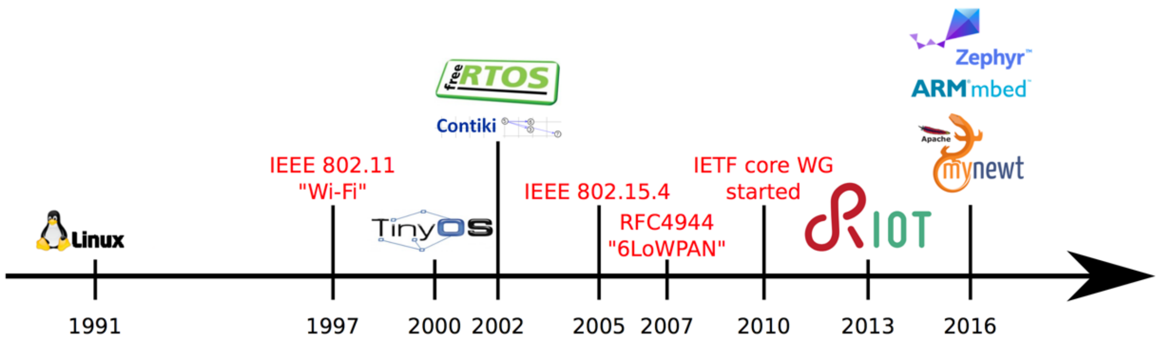


Figure 5: .

lection and interrupts. To qualify as an OS suitable for the IoT, it needs to meet the basic requirements: Low Random-access memory (RAM) footprint Low Read-only memory (ROM) footprint Multi-tasking Power management (PM) Soft real-time These requirements are directly bound to the type of hardware designed for the IoT. As this type of hardware in general needs to have a small form factor and a long battery life, the on-board memory is usually limited to keep down size and energy consumption. Also, because of the limited amount of memory, the implementation of threads is usually a challenging task, as context switching needs to store thread or process variables to memory. The size of the memory also directly affects the energy consumption, as memory in general is very power hungry during accesses. To be able reduce the energy consumption, the OS needs some kind of power management. The power management does not only let the OS turn on and off peripherals such as flash memory, I/O, and sensors, but also puts the MCU itself in different power modes. As the nodes can be used to control and monitor consumer devices, either a hard or soft real-time OS is required. Otherwise, actions requiring a close to instantaneous reaction might be indefinitely delayed. Hard real-time means that the OS scheduler can guarantee latency and execution time, whereas Soft real-time means that latency and execution time is seen as real-time but can not be guaranteed by the scheduler. Operating systems that meet the above requirements are compared in table 2.1 and 2.2.

OS	Architecture	Multi threading	Scheduling	Dynamic Memory	Memory protection	Network Stack	Virtualization and Completion
Contiki/Contiki-ng	Modular	✓	Interrupts execute w.r.t.	✓	✗	uIP Rime	Serialized Access
MANTIS	Modular	✗	Priority classes	✓	✗	At Kernel COMM layer	Semaphores.
Nano-RK	Layered	✓	Monotonic harmonized	✗	✗	Socket abstraction	Serialized access semaphores
LiteOS	Monolithic	✓	Round Robin	✓	✓	File	Synchronization primitives

Table 2.4: Common operating systems used in IoT environment [7]

2.2.1.1 Contiki

Contiki is a embedded operating system developed for IoT written in C [12]. It supports a broad range of MCUs and has drivers for various transceivers. The OS does not only support TCP/IPv4 and IPv6 with the uIP stack [9], but also has support for the 6LoWPAN stack and its own stack called RIME. It supports threading with a thread system called Photothreads [13]. The threads are stack-less and thus use only two bytes of memory per thread; however, each thread is bound to one function and it only has permission to control its own execution. Included in Contiki, there is a range of applications such as a HTTP, Constrained Application Protocol (CoAP), FTP, and DHCP servers, as well as other useful programs and tools. These applications can be included in a project and can run simultaneously with the help of Photothreads. The limitations to what applications can be run is the amount of RAM and ROM the target MCU provides. A standard system with IPv6 networking needs about 10 kB RAM and 30 kB ROM but as applications are added the requirements tend to grow.

Contiki is an open source operating system for the Internet of Things. Contiki connects tiny low-cost, low-power micro-controllers to the Internet.

2k RAM, 60k ROM; 10k RAM, 48K ROM Portable to tiny low-power micro-controllers I386 based, ARM, AVR, MSP430, ... Implements uIP stack IPv6 protocol for Wireless Sensor Networks (WSN) Uses the protothreads abstraction to run multiple process in an event based kernel. Emulates concurrency Contiki has an event based kernel (1 stack) Calls a process when an event happens

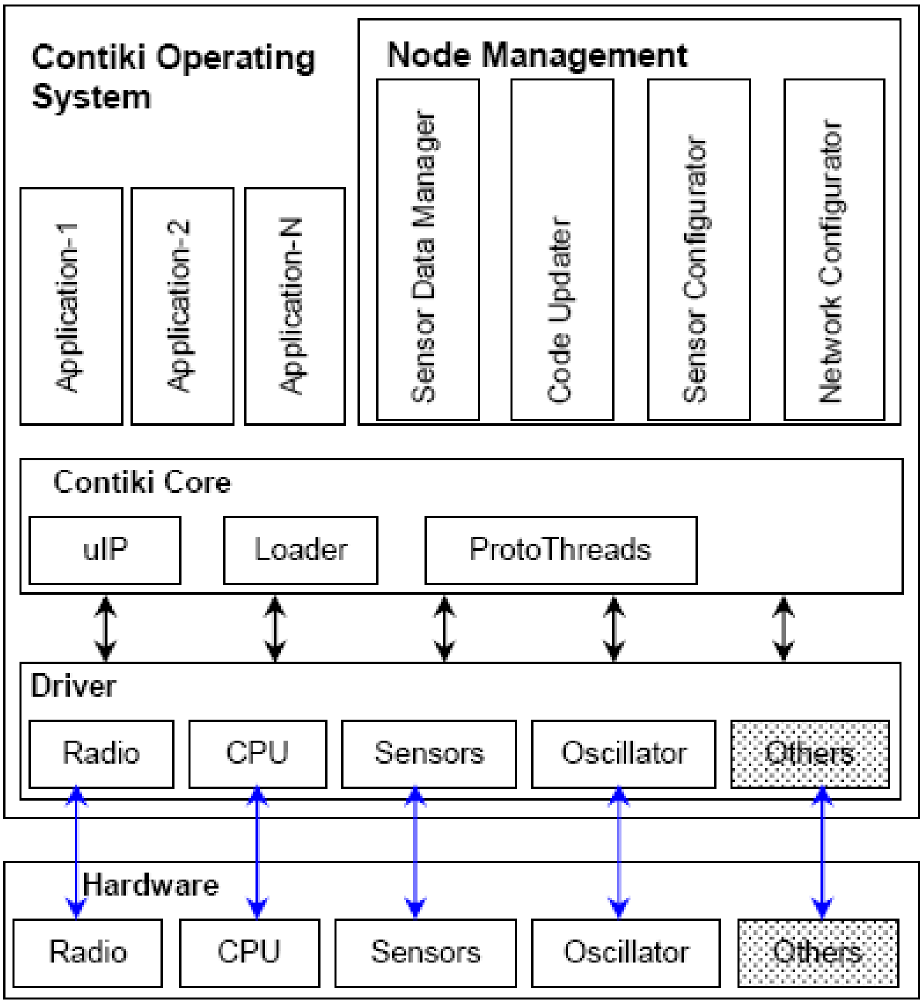


Figure 6: contiki .

Contiki size One of the main aspect of the system, is the modularity of the code. Besides the system core, each program builds only the necessary modules to be able to run, not the entire system image. This way, the memory used from the system, can be reduced to the strictly necessary. This methodology makes more practical any change in any module, if it is needed. The code size of Contiki is larger than that of TinyOS, but smaller than that of the Mantis system. Contiki's event kernel is significantly larger than that of TinyOS because of the different services provided. While the TinyOS event kernel only provides a FIFO event queue scheduler , the Contiki kernel supports both FIFO events and poll handlers with priorities. Furthermore, the flexibility in Contiki requires more run-time code than for a system like TinyOS, where compile time optimization can be done to a larger extent.

The documentation in the doc folder can be compiled, in order to get the html wiki of all the code. It needs doxygen installed, and to run the command make html. This will create a new folder, doc/hmtl, and in the index.html file, the wiki can be opened.

Contiki Hardware Contiki can be run in a number of platforms, each one with a different CPU. Tab.7 shows the hardware platforms currently defined in the Contiki code tree. All these platforms are in the platform folder of the code.

Kernel structure

2.2.1.2 RIOT

RIOT is a open source embedded operating system supported by Freie Universität Berlin, INIRA, and Hamburg University of Applied Sciences [14]. The kernel is written in C but the upper layers support C++ as well. As the project originates from a project with real-time and reliability requirements, the kernel supports hard real-time multi-tasking scheduling. One of the goals of the project is to make the OS completely POSIX compliant. Overhead for multi-threading is minimal with less than 25 bytes per thread. Both IPv6 and 6LoWPAN is supported together with UDP, TCP, and IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL); and CoAP and Concise Binary Object Representation (CBOR) are available as application level communication protocols.

2.2.1.3 TinyOS

TinyOS is written in Network Embedded Systems C (nesC) which is a variant of C [15]. nesC does not have any dynamic memory allocation and all program paths are available at compile-time. This is manageable thanks to the structure of the language; it uses modules and interfaces instead of functions [16]. The modules use and provide interfaces and are interconnected with configurations; this procedure makes up the structure of the program. Multitasking is implemented in two ways: trough tasks and events. Tasks, which focus on computation, are non-preemptive, and run until completion. In contrast, events which focus on external events i.e. interrupts, are preemptive, and have separate start and stop functions. The OS has full support for both 6LoWPAN and RPL, and also have libraries for CoAP.

2.2.1.4 freeRTOS

One of the more popular and widely known operating systems is freeRTOS [17]. Written in C with only a few source files, it is a simple but powerful OS, easy to overview and extend. It features two modes of scheduling, pre-emptive and co-operative, which may be selected according to the requirements of the application. Two types of multitasking are featured: one is a lightweight Co-routine type, which has a shared stack for lower RAM usage and is thus aimed to be used on very small devices; the other is simply called Task, has its own stack and can therefore be fully pre-empted. Tasks also support priorities which are used together with the pre-emptive scheduler. The communication methods supported out-of-the-box are TCP and UDP.

2.2.1.5 SDN platforms

Plan de controle	**Plan de gestion**	**Plan de doonées**
Controle d'admission	Controle et supervision de QoS	Controle du trafic
Réservation de ressources	Gestion de contrats	Façonnage du trafic
Routage	QoS mapping	Controle de congestion
Signalisation	Politique de QoS	Classification de paquets
		Marquage de paquets
		Ordonnancements des paquets
		Gestion de files d'attente

Table 2.5: An example table.

Sensor OpenFlow [20,21] SDWN [60] Smart [14] SDN-WISE [78] SDCSN [88] TinySDN [69,118] Virtual Overlay [59,87,90] Multi-task [122] SDWSN-RL [123] Wireless power transfer [126] Function alternation [65] Statistical machine learning [24] Context-based [91,92] Soft-WSN [9]

[8] Many studies have identified SDN as a potential solution to the WSN challenges, as well as a model for heterogeneous integration.

[8] This shortfall can be resolved by using the SDN approach.

[9] SDN also enhances better control of heterogeneous network infrastructures.

[9] Anadiotis et al. define a SDN operating system for IoT that integrates SDN based WSN **(SDN-WISE)**. This experiment shows how heterogeneity between different kinds of SDN networks can be achieved.

[9] In cellular networks, OpenRoads presents an approach of introducing SDN based heterogeneity in wireless networks for operators.

[10] There has been a plethora of (industrial) studies synergising SDN in IoT. The major characteristics of IoT are low latency,wireless access, mobility and heterogeneity.

[10] Thus a bottom-up approach application of SDN to the realisation of heterogeneous IoT is suggested.

[10] Perhaps a more complete IoT architecture is proposed, where the authors apply SDN principles in IoT heterogeneous networks.

[11] it provides the SDWSN with a proper model of network management, especially considering the potential of heterogeneity in SDWSN.

[11] We conjecture that the SDN paradigm is a good candidate to solve the heterogeneity in IoT.

Management architecture	Management feature	Controller configuration	Traffic Control	Configuration and monitoring	Scapability and localization	Communication management
[12] Sensor Open Flow	SDN support protocol	Distributed	in/out-band	✓	✓	✓
[13] SDWN	Duty sycling, aggregation, routing	Centralized	in-band	✓		
[14] SDN-WISE	Programming simplicity and aggregation	Distributed	in-band		✓	
[degante_smart_2014] Smart	Efficiency in resource allocation	Distributed	in-band		✓	
SDCSN	Network reliability and QoS	Distributed	in-band		✓	
TinySDN	In-band-traffic control	Distributed	in-band		✓	
Virtual Overlay	Network flexibility	Distributed	in-band		✓	
Context based	Network scalability and performance	Distributed	in-band		✓	
CRLB	Node localization	Centralized	in-band			
Multi-hope	Traffic and energy control	Centralized	in-band			✓
Tiny-SDN	Network task measurement	-	in-band			

Table 2.6: SDN-based network and topology management architectures. [10]

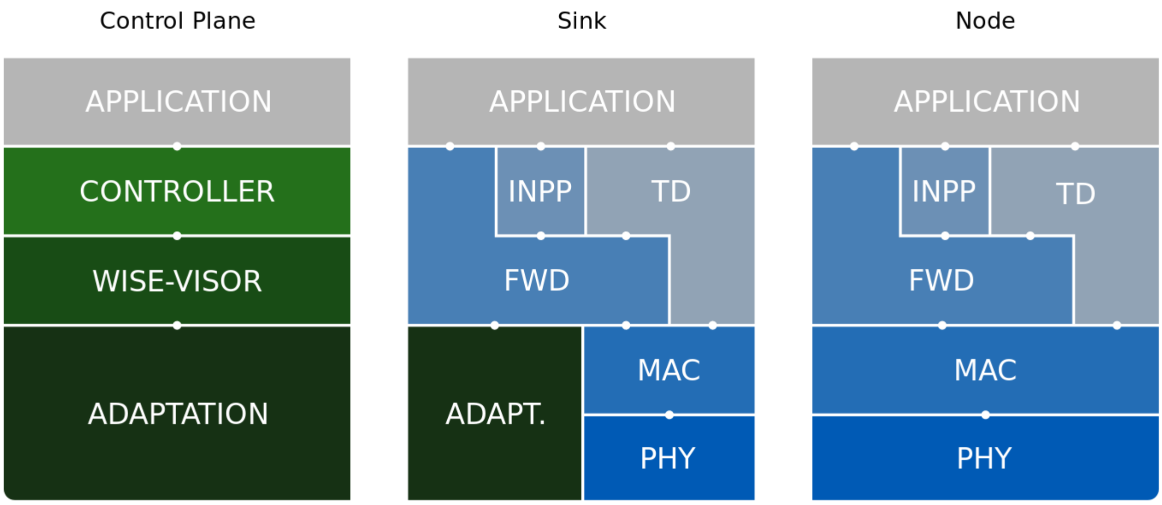


Figure 7: LPWAN connectivity.

2.2.1.6 Summary and conclusion

	LiteOS	Nano-RK	MANTIS	Contiki
Architecture	Monolithic	Layered	Modular	Modular
Scheduling Memory	Round Robin	Monotonic harmonized	Priority classes	Interrupts execute w.r.t.
Network	File	Socket abstraction	At Kernel COMM layer	uIP, Rime
Virtualization and Completion	Synchronization primitives	Serialized access semaphores	Semaphores	Serialized, Access
Multi threading	✓	✓	✗	✓
Dynamic protection	✓	✗	✓	✓
Memory Stack	✓	✗	✗	✗

Table 2.7: Common operating systems used in IoT environment [7]

2.2.2 Processing Unit

Even though the hardware is in one sense the tool that the OS uses to make IoT possible, it is still very important to select a platform that is future-proof and extensible. To be regarded as an extensible platform, the hardware needs to have I/O connections that can be used by external peripherals. Amongst the candidate interfaces are Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I 2 C), and Controller Area Network (CAN). These interfaces allow developers to attach custom-made PCBs with sensors for monitoring or actuators for controlling the environment. The best practice is to implement an extension socket

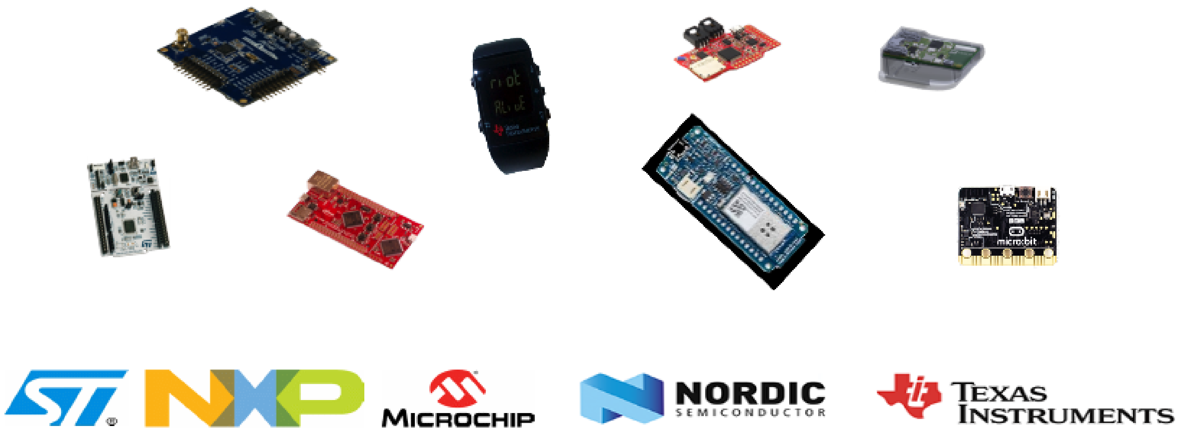


Figure 8: .

with a well-known form factor. A future-proof device is specified as a device that will be as attractive in the future as it is today. For hardware, this is very hard to achieve as there is constant development that follows Moores Law [4]; however, the most important aspects are: the age of the chip, its expected remaining lifetime, and number of current implementations i.e. its popularity. If a device is widely used by consumers, the lifetime of the product is likely to be extended. One last thing to take into consideration is the product family; if the chip belongs to a family with several members the transition to a newer chip is usually easier.

2.2.2.1 OpenMote

OpenMote is based on the Ti CC2538 System on Chip (SoC), which combines an ARM Cortex-M3 with a IEEE 802.15.4 transceiver in one chip [18, 19]. The board follows the XBee form factor for easier extensibility, which is used to connect the core board to either the OpenBattery or OpenBase extension boards [20, 21]. It originates from the CC2538DK which was used by Thingsquare to demo their Mist IoT solution [22]. Hence, the board has full support for Contiki, which is the foundation of Thingsquare. It can run both as a battery-powered sensor board and as a border router, depending on what extension board it is attached to, e.g OpenBattery or OpenBase. Furthermore, the board has limited support but ongoing development for RIOT and also full support for freeRTOS.

2.2.2.2 MSB430-H

The Modular Sensor Board 430-H from Freie Universität Berlin was designed for their ScatterWeb project [23]. As the university also hosts the RIOT project, the decision to support RIOT was natural. The main board has a Ti MSP430F1612 MCU [24], a **Ti CC1100 transceiver**, and a battery slot for dual AA batteries; it also includes a SHT11 temperature and humidity sensor and a MMA7260Q accelerometer to speed up early development. All GPIO pins and buses are connected to external pins for extensibility. Other modules with new peripherals can then be added by making a PCB that matches the external pin layout.

2.2.2.3 Zolertia

As many other Wireless Sensor Network (WSN) products, the Zolertia Z1 builds upon the MSP430 MCU [25, 26]. The communication is managed by the Ti CC2420 which operates in the 2.4 GHz band. The platform includes two sensors: the SHT11 temperature and humidity sensor and the MMA7600Q accelerometer. Extensibility is ensured with: two connections designed especially for external sensors, an external connector with USB, Universal asynchronous **receiver/transmitter (UART)**, SPI, and I 2 C.

2.2.3 Radio Unit

2.2.3.1 Lora Tranceiver

To limit the complexity of the radio unit:

- limiting message size: maximum application payload size between 51 and 222 bytes, depending on the spreading factor
- using simple channel codes: Hamming code
- supporting only half-duplex operation

- using one transmit-and-receive antenna
- on-chip integrating power amplifier (since transmit power is limited)

limiting message size: maximum application payload size between 51 and 222 bytes, depending on the spreading factor using simple channel codes: Hamming code supporting only half-duplex operation using one transmit-and-receive antenna on-chip integrating power amplifier (since transmit power is limited)

Ref	Module	Frequency MHz	Tx power	Rx power	Sensitivity	Channels	Distance
[_waspmote_]	**Semtech SX1272**	863-870 (EU) 902-928 (US)	14 dBm	dBm	-134 dBm	8 13	22+ km
[_waspmote_]	**rn2483**						
[_waspmote_]	**rn2903**						
[_waspmote_]	**rak811**						
[_waspmote_]	**Semtech sx1276**						
[_waspmote_]	**rfm95**						
[_waspmote_]	**CMWX1ZZABZ-078**						
[_waspmote_]	**LoPy4**						
[_waspmote_]	**mDot**						
[_waspmote_]	**xDot**						
[_waspmote_]	**Laird RM192**						
[_waspmote_]	**Laird RM186**						
[_waspmote_]	**CMWX1ZZABZ-078**						
[_waspmote_]	**Also Laird RM1xx**						
[_waspmote_]	**iMST iM88x/iM98x**						
[_waspmote_]	**Mic SAM RN34/35**						
[_waspmote_]	**Semtech SX1278**						

Table 2.8

2.2.4 Sensing Unit

2.2.5 Summary and discussion

2.3 IoT Communication protocols

Application protocol	DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	HTTP
Service discovery		mDNS			DNS-SD		
Transport				UDP/TCP			
Network		IPv6 RPL			IPv4/IPv6		
		6LowPan			RFC 2464		RFC 5072
MAC		IEEE 802.15.4	IEEE 802.11 (Wi-Fi)		IEEE 802.3 (Ethernet)		2G, 3G, LTE
		2.4GHz, 915, 868MHz	2.4, 5GHz				
		DSS, FSK, OFDM	CSMA/CA		CUTP, FO		

Table 2.9: Standardization efforts that support the IoT

2.3.1 Application

2.3.1.1 LwM2M

2.3.1.2 CBOR

2.3.1.3 DTLS

2.3.1.4 OSCOAP

2.3.1.5 COAP (COnstrained Application Protocol)

The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks in the Internet of Things. More detailed information about the protocol is given in the Contiki OS CoAP section.

1) Overview Like HTTP, CoAP is a document transfer protocol. Unlike HTTP, CoAP is designed for the needs of constrained devices. The packets are much smaller than HTTP TCP flows. Packets are simple to generate and can be parsed in place without consuming extra RAM in constrained devices. CoAP runs over UDP, not TCP. Clients and servers communicate through connectionless datagrams. Retries and reordering are implemented in the application stack. It follows a client/server model. Clients make requests to servers, servers send back responses. Clients may GET, PUT, POST and DELETE resources. CoAP implements the REST model from HTTP, with the primitives GET, POST, PUT and DELETE.

2) Coap Methods CoAP extends the HTTP request model with the ability to observe a resource. When the observe flag is set on a CoAP GET request, the server may continue to reply after the initial document has been transferred. This allows servers to stream state changes to clients as they occur. Either end may cancel the observation. CoAP defines a standard mechanism for resource discovery. Servers provide a list of their resources (along with metadata about them) at /.well-known/core. These links are in the application/link-format media type and allow a client to discover what resources are provided and what media types they are.

3) Coap Transactions

4) Coap Messages The CoAP message structure is designed to be simpler than HTTP, for reduced transmission data. Each field responds to a specific purpose.

- Constrained Application Protocol
- The IETF Constrained RESTful Environments
- CoAP is bound to UDP
- CoAP can be divided into two sub-layers
  - messaging sub-layer
  - request/response sub-layer
    a) Confirmable.
    b) Non-confirmable.
    c) Piggybacked responses.
    d) Separate response
- CoAP, as in HTTP, uses methods such as:
  - GET, PUT, POST and DELETE to
  - Achieve, Create, Retrieve, Update and Delete
  - Ex: the GET method can be used by a server to inquire the clients temperature

<table>
<tr><td>Ver</td><td>T</td><td>TKL</td><td>Code</td><td>Message ID</td></tr>
<tr><td colspan="5">Token</td></tr>
<tr><td colspan="5">Options</td></tr>
<tr><td colspan="3">11111111</td><td colspan="2">Payload</td></tr>
</table>

CoAP Header

Ver: is the version of CoAP
T: is the type of Transaction
TKL: Token length
Code: represents the request method (1-10) or response code (40-255).
- Ex: the code for GET, POST, PUT, and DELETE is 1, 2, 3, and 4, respectively.

Message ID: is a unique identifier for matching the response.
Token: Optional response matching token.

2.3.1.6 MQTT

- Message Queue Telemetry Transport
- Andy Stanford-Clark of IBM and Arlen Nipper of Arcom

    - Standardized in 2013 at OASIS
- MQTT uses the publish/subscribe pattern to provide transition flexibility and simplicity of implementation
- MQTT is built on top of the TCP protocol
- MQTT delivers messages through three levels of QoS
- Specifications
    - MQTT v3.1 and MQTT-SN (MQTT-S or V1.2)
    - MQTT v3.1 adds broker support for indexing topic names
- The publisher acts as a generator of interesting data.

Message Type	UDP	QoS Level	Retain
Remaining length			
Variable length header			
Variable length message payload			

CoAP Header

Message type: CONNECT (1), CONNACK (2), PUBLISH (3), SUBSCRIBE (8) and so on
DUP flag: indicates that the massage is duplicated
QoS Level: identify the three levels of QoS for delivery assurance of Publish messages
Retain field: retain the last received Publish message and submit it to new subscribers as a first message

2.3.1.7 XMPP

- Extensible Messaging and Presence Protocol
- Developed by the Jabber open source community
- An IETF instant messaging standard used for:
    - multi-party chatting, voice and telepresence
- Connects a client to a server using a XML stanzas
- An XML stanza is divided into 3 components:
    - message: fills the subject and body fields
    - presence: notifies customers of status updates
    - iq (info/query): pairs message senders and receivers
- Message stanzas identify:
    - the source (from) and destination (to) addresses
    - types, and IDs of XMPP entities

2.3.1.8 AMQP

- Advanced Message Queuing Protocol
- Communications are handled by two main components
    - exchanges: route the messages to appropriate queues.
    - message queues: Messages can be stored in message queues and then be sent to receivers
- It also supports the publish/subscribe communications.
- It defines a layer of messaging on top of its transport layer.
- AMQP defines two types of messages
    - bare massages: supplied by the sender
    - annotated messages: seen at the receiver
- The header in this format conveys the delivery parameters:
    - durability, priority, time to live, first acquirer & delivery count.
- AMQP frame format

  Size the frame size.
  DOFF the position of the body inside the frame.
  Type the format and purpose of the frame.
    * Ex: 0x00 show that the frame is an AMQP frame
    * Ex: 0x01 represents a SASL frame.

2.3.1.9 DDS

- Data Distribution Service
- Developed by Object Management Group (OMG)

Application protocol	RestFull	Transport	Publish/Subscribe	Request/Response	Security	QoS	Header size (Byte)
COAP	✓	UDP	✓	✓	DTLS	✓	4
MQTT	✗	TCP	✓	✗	SSL	✓	2
MQTT-SN	✗	TCP	✓	✗	SSL	✓	2
XMPP	✗	TCP	✓	✓	SSL	✗	-
AMQP	✗	TCP	✓	✗	SSL	✓	8
DDS	✗	UDP TCP	✓	✗	SSL DTLS	✓	-
HTTP	✓	TCP	✗	✓	SSL	✗	-

Table 2.10: Application protocols comparison

- Supports 23 QoS policies:
  - like security, urgency, priority, durability, reliability, etc
- Relies on a broker-less architecture
  - uses multicasting to bring excellent Quality of Service
  - real-time constraints
- DDS architecture defines two layers:

  DLRL Data-Local Reconstruction Layer
  - serves as the interface to the DCPS functionalities

  DCPS Data-Centric Publish/Subscribe
  - delivering the information to the subscribers
- 5 entities are involved with the data flow in the DCPS layer:
  - Publisher:disseminates data
  - DataWriter: used by app to interact with the publisher
  - Subscriber: receives published data and delivers them to app
  - DataReader: employed by Subscriber to access received data
  - Topic: relate DataWriters to DataReaders
- No need for manual reconfiguration or extra administration
- It is able to run without infrastructure
- It is able to continue working if failure happens.
- It inquires names by sending an IP multicast message to all the nodes in the local domain
  - Clients asks devices that have the given name to reply back
  - the target machine receives its name and multicasts its IP @
  - Devices update their cache with the given name and IP @

2.3.1.10 mDNS

- Requires zero configuration aids to connect machine
- It uses mDNS to send DNS packets to specific multicast addresses through UDP
- There are two main steps to process Service Discovery:
  - finding host names of required services such as printers
  - pairing IP addresses with their host names using mDNS
- Advantages
  - IoT needs an architecture without dependency on a configuration mechanism
  - smart devices can join the platform or leave it without affecting the behavior of the whole system
- Drawbacks
  - Need for caching DNS entries

2.3.2 Network

2.3.2.1 6TiSCH

2.3.2.2 OLSRv2

2.3.2.3 AODVv2

2.3.2.4 LoRaWAN

2.3.2.5 ROHC

2.3.2.6 IPHC

2.3.2.7 SCHC

2.3.2.8 NHC

2.3.2.9 ROLL

2.3.2.10 RPL

RPL is a Distance Vector IPv6 routing protocol for LLNs that specifies how to build a Destination Oriented Directed Acyclic Graph (DODAG) using an objective function and a set of metrics/constraints. The objective function operates on a combination of metrics and constraints to compute the best path.

An RPL Instance consists of multiple Destination Oriented Directed Acyclic Graphs (DODAGs). Traffic moves either up towards the DODAG root or down towards the DODAG leafs. The graph building process starts at the root or LBR (LowPAN Border Router). There could be multiple roots configured in the system. The RPL routing protocol specifies a set of ICMPv6 control messages to exchange graph related information. These messages are called DIS (DODAG Information Solicitation), DIO (DODAG Information Object) and DAO (DODAG Destination Advertisement Object). The root starts advertising the information about the graph using the DIO message. The nodes in the listening vicinity (neighbouring nodes) of the root will receive and process DIO messages potentially from multiple nodes and makes a decision based on certain rules (according to the objective function, DAG characteristics, advertised path cost and potentially local policy) whether to join the graph or not. Once the node has joined a graph it has a route toward the graph (DODAG) root. The graph root is termed as the parent of the node. The node computes the rank of itself within the graph, which indicates the coordinates of the node in the graph hierarchy. If configured to act as a router, it starts advertising the graph information with the new information to its neighbouring peers. If the node is a leaf node, it simply joins the graph and does not send any DIO message. The neighbouring peers will repeat this process and do parent selection, route addition and graph information advertisement using DIO messages. This rippling effect builds the graph edges out from the root to the leaf nodes where the process terminates. In this formation each node of the graph has a routing entry towards its parent (or multiple parents depending on the objective function) in a hop-by-hop fashion and the leaf nodes can send a data packet all the way to root of the graph by just forwarding the packet to its immediate parent. This model represents a MP2P (Multipoint-to-point) forwarding model where each node of the graph has reach-ability toward the graph root. This is also referred to as UPWARD routing. Each node in the graph has a rank that is relative and represents an increasing coordinate of the relative position of the node with respect to the root in graph topology. The notion of rank is used by RPL for various purposes including loop avoidance. The MP2P flow of traffic is called the up direction in the DODAG.

The DIS message is used by the nodes to proactively solicit graph information (via DIO) from the neighbouring nodes should it become active in a stable graph environment using the poll or pull model of retrieving graph information or in other conditions. Similar to MP2P or up direction of traffic, which flows from the leaf towards the root there is a need for traffic to flow in the opposite or down direction. This traffic may originate from outside the LLN network, at the root or at any intermediate nodes and destined to a (leaf) node. This requires a routing state to be built at every node and a mechanism to populate these routes. This is accomplished by the DAO (Destination Advertisement Object) message. DAO messages are used to advertise prefix reachability towards the leaf nodes in support of the down traffic. These messages carry prefix information, valid lifetime and other information about the distance of the prefix. As each node joins the graph it will send DAO message to its parent set. Alternately, a node or root can poll the sub-dag for DAO message through an indication in the DIO message. As each node receives the DAO message, it processes the prefix information and adds a routing entry in the routing table. It optionally aggregates the prefix information received from various nodes in the subdag and sends a DAO message to its parent set. This process continues until the prefix information reaches the root and a complete path to the prefix is setup. Note that this mode is called the storing mode of operation where intermediate nodes have available memory to store routing tables. RPL also supports another mode called non-storing mode where intermediate node do not store any routes.

2.3.2.11 6LowPAN

6LoWPAN is a networking technology or adaptation layer that allows IPv6 packets to be carried efficiently within a small link layer frame, over IEEE 802.15.4 based networks. As the full name implies, IPv6 over Low-Power Wireless Personal Area Networks, it is a protocol for connecting wireless low power networks using IPv6.

As the full name implies, IPv6 over Low-Power Wireless Personal Area Networks, it is a protocol for connecting wireless low power networks using IPv6.

1) Characteristics

- Compression of IPv6 and UDP/ICMP headers
- Fragmentation / reassembly of IPv6 packets

- Mesh addressing
- Stateless auto configuration
-

2) Encapsulation Header format All LowPAN encapsulated datagrams are prefixed by an encapsulation header stack. Each header in the stack starts with a header type field followed by zero or more header fields.

3) Fragment Header The fragment header is used when the payload is too large to fit in a single IEEE 802.15.4 frame. The Fragment header is analogous to the IEEE 1394 Fragment header and includes three fields: Datagram Size, Datagram Tag, and Datagram Offset. Datagram Size identifies the total size of the unfragmented payload and is included with every fragment to simplify buffer allocation at the receiver when fragments arrive out-oforder. Datagram Tag identifies the set of fragments that correspond to a given payload and is used to match up fragments of the same payload. Datagram Offset identifies the fragments offset within the unfragmented payload and is in units of 8-byte chunks.

4) Mesh addressing header The Mesh Addressing header is used to forward 6LoWPAN payloads over multiple radio hops and support layer-two forwarding. The mesh addressing header includes three fields: Hop Limit, Source Address, and Destination Address. The Hop Limit field is analogous to the IPv6 Hop Limit and limits the number of hops for forwarding. The Hop Limit field is decremented by each forwarding node, and if decremented to zero the frame is dropped. The source and destination addresses indicate the end-points of an IP hop. Both addresses are IEEE 802.15.4 link addresses and may carry either a short or extended address.

5) Header compression (RFC4944) RFC 4944 defines HC1, a stateless compression scheme optimized for link-local IPv6 communication. HC1 is identified by an encoding byte following the Compressed IPv6 dispatch header, and it operates on fields in the upper-layer headers. 6LoWPAN elides some fields by assuming commonly used values. For example, it compresses the 64-bit network prefix for both source and destination addresses to a single bit each when they carry the well-known link-local prefix. 6LoWPAN compresses the Next Header field to two bits whenever the packet uses UDP, TCP, or ICMPv6. Furthermore, 6LoWPAN compresses Traffic Class and Flow Label to a single bit when their values are both zero. Each compressed form has reserved values that indicate that the fields are carried inline for use when they dont match the elided case. 6LoWPAN elides other fields by exploiting cross-layer redundancy. It can derive Payload Length which is always elided from the 802.15.4 frame or 6LoWPAN fragmentation header. The 64-bit interface identifier (IID) for both source and destination addresses are elided if the destination can derive them from the corresponding link-layer address in the 802.15.4 or mesh addressing header. Finally, 6LoWPAN always elides Version by communicating via IPv6.

The HC1 encoding is shown in Figure 11. The first byte is the dispatch byte and indicates the use of HC1. Following the dispatch byte are 8 bits that identify how the IPv6 fields are compressed. For each address, one bit is used to indicate if the IPv6 prefix is linklocal and elided and one bit is used to indicate if the IID can be derived from the IEEE 802.15.4 link address. The TF bit indicates whether Traffic Class and Flow Label are both zero and elided. The two Next Header bits indicate if the IPv6 Next Header value is 7UDP, TCP, or ICMP and compressed or carried inline. The HC2 bit indicates if the next header is compressed using HC2. Fully compressed, the HC1 encoding reduces the IPv6 header to three bytes, including the dispatch header. Hops Left is the only field always carried inline.

RFC 4944 uses stateless compression techniques to reduce the overhead of UDP headers. When the HC2 bit is set in the HC1 encoding, an additional 8-bits is included immediately following the HC1 encoding bits that specify how the UDP header is compressed. To effectively compress UDP ports, 6LoWPAN introduces a range of wellknown ports (61616 61631). When ports fall in the well-known range, the upper 12 bits may be elided. If both ports fall within range, both Source and Destination ports are compressed down to a single byte. HC2 also allows elision of the UDP Length, as it can be derived from the IPv6 Payload Length field.

The best-case compression efficiency occurs with link-local unicast communication, where HC1 and HC2 can compress a UDP/IPv6 header down to 7 bytes. The Version, Traffic Class, Flow Label, Payload Length, Next Header, and linklocal prefixes for the IPv6 Source and Destination addresses are all elided. The suffix for both IPv6 source and destination addresses are derived from the IEEE 802.15.4 header.

However, RFC 4944 does not efficiently compress headers when communicating outside of link-local scope or when using multicast. Any prefix other than the linklocal prefix must be carried inline. Any suffix must be at least 64 bits when carried inline even if derived from a short 802.15.4 address. As shown in Figure 8, HC1/HC2 can compress a link-local multicast UDP/IPv6 header down to 23 bytes in the best case. When communicating with nodes outside the LoWPAN, the IPv6 Source Address prefix and full IPv6 Destination Address must be carried inline.

6) Header compression Improved (draft-hui-6lowpan-hc-01) To provide better compression over a broader range of scenarios, the 6LoWPAN working group is standardizing an improved header compression encoding format, called HC. The format defines a new encoding for compressing IPv6 header, called IPHC. The new format allows Traffic Class and Flow Label to be individually compressed, Hop Limit compression when common values (E.g., 1 or 255) are used, makes use of shared-context to elide the prefix from IPv6 addresses, and supports multicast addresses most often used for IPv6 ND and SLAAC. Contexts act as shared state for all nodes within the LoWPAN. A single context holds a single prefix. IPHC identifies the context using a 4-bit index, allowing IPHC to support up to 16 contexts simultaneously within the LoWPAN. When an IPv6 address matches a contexts stored prefix, IPHC compresses the prefix to the contexts 4-bit identifier. Note that contexts are not limited to prefixes assigned to the LoWPAN but can contain any arbitrary prefix. As a result, share contexts can be configured such that LoWPAN nodes can compress the prefix in both Source and Destination addresses even when communicating with nodes outside the LoWPAN.

The improved header compression encoding is shown in Figure 8. The first three bits (011) form the header type and indicate the use of IPHC. The TF bits indicate whether the Traffic Class and/or Flow Label fields are compressed. The HLIM bits indicate whether the Hop Limit takes the value 1 or 255 and compressed, or carried inline.

Bits 8-15 of the IPHC encoding indicate the compression methods used for the IPv6 Source and Destination Addresses. When the Context Identifier (CID) bit is zero, the default context may be used to compress Source and/or Destination Addresses. This mode is typically when both Source and Destination Addresses are assigned to nodes in the same LoWPAN. When the CID bit is one, two additional 4-bit fields follow the IPHC encoding to indicate which one of 16 contexts is in use for the source and destination addresses. The Source Address Compression (SAC) indicates whether stateless compression is used (typically for link-local communication) or stateful context-based compression is used (typically for global communication). The Source Address Mode (SAM) indicates whether the full Source Address is carried inline, upper 16 or 64-bits are elided, or the full Source Address is elided. When SAC is set and the Source Addresses prefix is elided, the identified context is used to restore those bits. The Multicast (M) field indicates whether the Destination Address is a unicast or multicast address. When the Destination Address is a unicast address, the DAC and DAM bits are analogous to the SAC and SAM bits. When the Destination Address is a multicast address, the DAM bits indicate different forms of multicast compression. HC also defines a new framework for compressing arbitrary next headers, called NHC. HC2 in RFC 4944 is only capable of compressing UDP, TCP, and ICMPv6 headers, the latter two are not yet defined. Instead, the NHC header defines a new variable length Next Header identifier, allowing for future definition of arbitrary next header compression encodings. HC initially defines a compression encoding for UDP headers, similar to that defined in RFC 4944. Like RFC 4944, HC utilizes the same well-known port range (61616-61631) to effectively compress UDP ports down to 4-bits each in the best case. However, HC no longer provides an option to carry the Payload Length in line, as it can always be derived from the IPv6 header. Finally, HC allows elision of the UDP Checksum whenever an 10upper layer message integrity check covers the same information and has at least the same strength. Such a scenario is typical when transportor application-layer security is used. As a result, the UDP header can be compressed down to two bytes in the best case.

Routing protocol	Control Cost	Link Cost	Node Cost
OSPF/IS-IS	✗	✓	✗
OLSRv2	?	✓	✓
RIP	✓	?	✗
DSR	✓	✗	✗
RPL	✓	✓	✓

Table 2.11: Routing protocols comparison [**_rpl2_**]

- Routing over low-power and lossy links (ROLL)
- Support minimal routing requirements.
  - like multipoint-to-point, point-to-multipoint and point-to-point.
- A Destination Oriented Directed Acyclic Graph (DODAG)
  - Directed acyclic graph with a single root.
  - Each node is aware of ts parents
  - but not about related children
- RPL uses four types of control messages
  - DODAG Information Object (DIO)
  - Destination Advertisement Object (DAO)

- ➟ DODAG Information Solicitation (DIS)
- ➟ DAO Acknowledgment (DAO-ACk)

➠ Standard topologies to form IEEE 802.15.4e networks are

Star contains at least one FFD and some RFDs

Mesh contains a PAN coordinator and other nodes communicate with each other

Cluster consists of a PAN coordinator, a cluster head and normal nodes.

➠ The IEEE 802.15.4e standard supports 2 types of network nodes

FFD Full function device: serve as a coordinator

- ✳ It is responsible for creation, control and maintenance of the net
- ✳ It store a routing table in their memory and implement a full MAC

RFD Reduced function devices: simple nodes with restricted resources

- ✳ They can only communicate with a coordinator
- ✳ They are limited to a star topology

Routing protocol	Control Cost	Link Cost	Node Cost
OSPF/IS-IS	✗	✓	✗
OLSRv2	?	✓	✓
RIP	✓	?	✗
DSR	✓	✗	✗
RPL	✓	✓	✓

Table 2.12: Routing protocols comparison [**_rpl2_**]

2.3.3 MAC

Channel based	FDMA	OFDMA WDMA SC-FDMA		
	TDMA	MF-TDMA STDMA		
	CDMA	W-CDMA TD-CDMA TD-SCDMA DS-CDMA FH-CDMA MC-CDMA		
	SDMA	HC-SDMA		
Packet-based	Collision recovery	ALOHA Slotted ALOHA R-ALOHA AX.25 CSMA/CD		
	Collision avoidance	MACA MACAW CSMA CSMA/CA DCF PCF HCF CSMA/CARP		
	Collision-free	Token ring Token bus MS-ALOHA		
Duplexing methods	Delay and disruption tolerant	MANET VANET DTN Dynamic Source Routing		

Table 2.13

2.3.3.1 Sharing the channel

1) TDMA, FDMA, CDMA, TSMA

2.3.3.2 Transmitting information

1) TFDM, TDSSS, TFHSS

2.3.4 Radio

2.3.4.1 Digital modulation

1) ASK, APSK, CPM, FSK, MFSK, MSK, OOK, PPM, PSK, QAM, SC-FDE, TCM WDM

2.3.4.2 Hierarchical modulation

1) QAM, WDM

2.3.4.3 Spread spectrum

1) SS, DSSS, FHSS, THSS

2.3.5 Summary and discussion

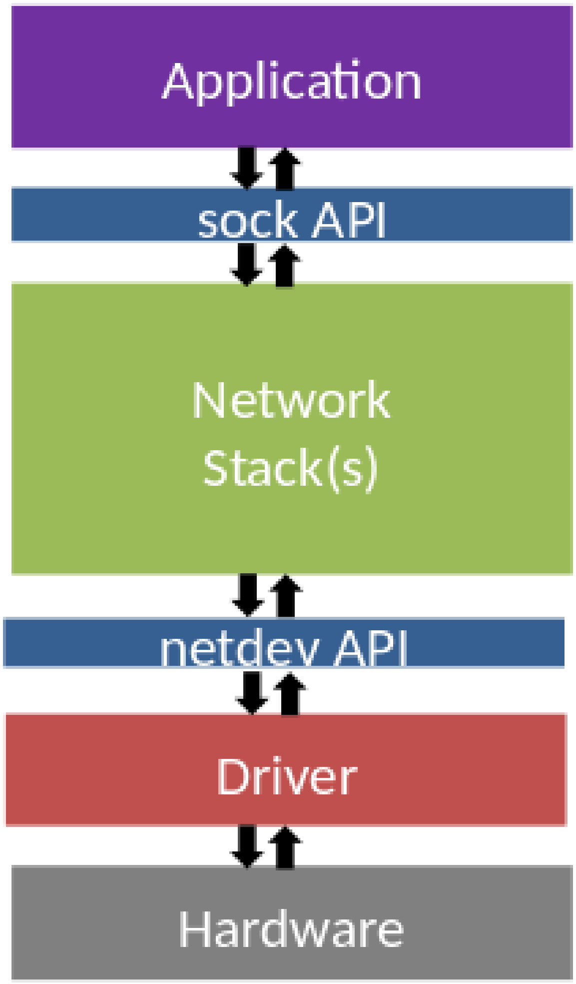


Figure 9: LPWAN.

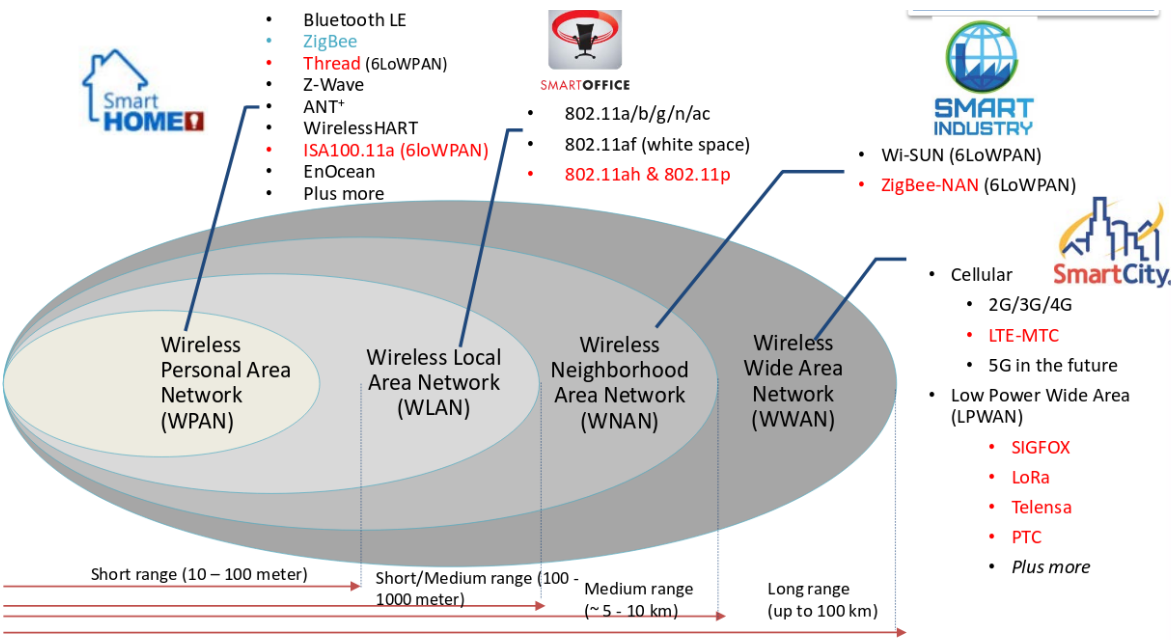


Figure 10: LPWAN.

2.4 IoT Norms & Standards

Several different wireless communication protocols, such as Wireless LAN (WLAN), BLE, 6LoWPAN, and ZigBee may be suitable for IoT applications. They all operate in the 2.4GHz frequency band and this, to-

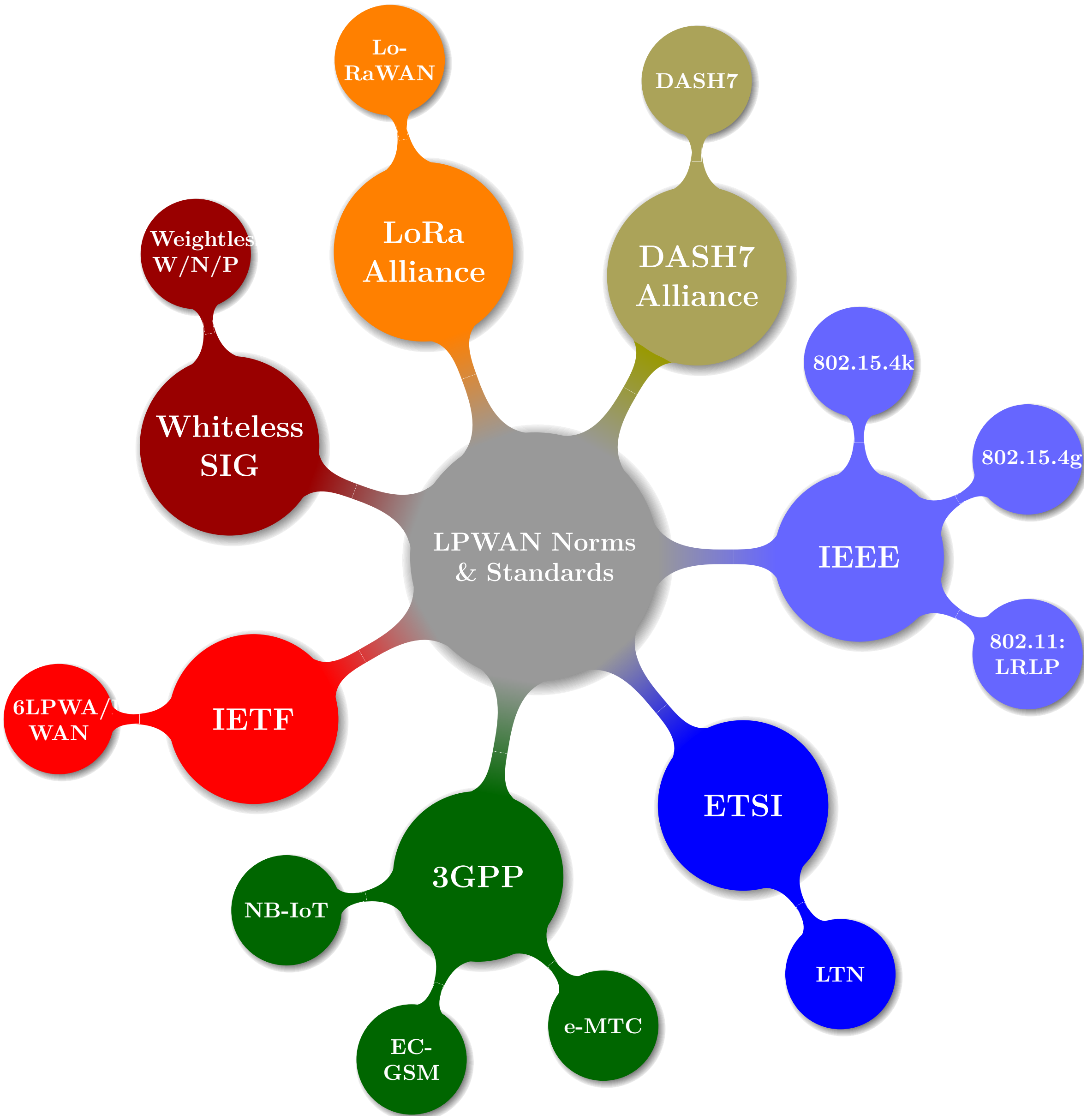


Figure 11: LPWAN.

gether with the limited output power in this band, means that they all have a similar range. The main differences are located in the MAC, PHY, and network layer. WLAN is much too power hungry as seen in table 2.6 and is only listed as a reference for the comparisons.

2.4.1 Divers

2.4.1.1 IPLC

2.4.1.2 BACnet

2.4.1.3 Z-WAze

2.4.1.4 Bluetooth LE

BLE is developed to be backwards compatible with Bluetooth, but with lower data rate and power consumption [28]. Featuring a data rate of 1Mbit/s with a peak current consumption less than 15mA, it is a very efficient protocol for small amounts of data. Each frame can be transmitted 47bytes in 1Mbit/s = 376Mus; thanks to the short transmission time, the transceivers consumes less power as the transceiver can be in receive mode or completely off most of the time. BLE uses its own addressing methods and as the MAC frame size (figure 2.6) is only 39bytes, thus IPv6 addressing is not possible.

Starting from Bluetooth version 4.2, there is support for IPv6 addressing with the Internet Protocol Support Profile; the new version allows the BLE frame to be variable between 2 257 bytes. The network set-up is controlled by the standard Bluetooth methods, whereas IPv6 addressing is handled by 6LoWPAN as specified in IPv6 over Bluetooth Low Energy [29].

2.4.2 SigFox

2.4.3 IETF

2.4.3.1 6LoWPAN

is a relatively new protocol that is maintained by the Internet Engineering Task Force (IETF) [7, 6]. The purpose of the protocol is to enable IPv6 traffic over a IEEE 802.15.4 network with as low overhead as possible; this is achieved by compressing the IPv6 and UDP header. A full size IPv6 + UDP header is 40+8 bytes which is tild 38% of a IEEE 802.15.4 frame, but with the header compression this overhead can be reduced to 7 bytes, thus reducing the overhead to tild 5%, as seen in figures 2.3 and 2.4.

2.4.4 3GPP

2.4.4.1 NB-IoT

2.4.4.2 EC-GSM

2.4.4.3 e-MTC

2.4.5 IEEE

2.4.5.1 IEEE 802.11

2.4.5.2 IEEE 802.15.4

At present days, there are several technology standards. Each one is designed for a specific need in the market. For the Wireless Sensor Networks, the aim is to transmit little information, in a small range, with a small power consumption and low cost. The IEEE 802.15.4 standard offers physical and media access control layers for low-cost, low-speed, low-power Wireless Personal Area Networks (WPANs)

1) Physical Layer The standard operates in 3 different frequency bands: - 16 channels in the 2.4GHz ISM band - 10 channels in the 915MHz ISM band - 1 channel in the European 868MHz band

2) Definitions Coordinator: A device that provides synchronization services through the transmission of beacons. PAN Coordinator: The central coordinator of the PAN. This device identifies its own network as well as its configurations. There is only one PAN Coordinator for each network. Full Function Device (FFD): A device that implements the complete protocol set, PAN coordinator capable , talks to any other device. This type of device is suitable for any topology. Reduced Function Device (RFD): A device with a reduced implementation of the protocol, cannot become a PAN Coordinator. This device is limited to leafs in some topologies.

3) Topologies Star topology: All nodes communicate via the central PAN coordinator , the leafs may be any combination of FFD and RFD devices. The PAN coordinator usually uses main power.

Peer to peer topology: Nodes can communicate via the central PAN coordinator and via additional point-to-point links . All devices are FFD to be able to communicate with each other.

Combined Topology: Star topology combined with peer-to-peer topology. Leafs connect to a network via coordinators (FFDs) . One of the coordinators serves as the PAN coordinator .

IEEE 802.15.4

The IEEE 802.15.4 standard defines the PHY and MAC layers for wireless communication [6]. It is designed to use as little transmission time as possible but still have a decent payload, while consuming as little power as possible. Each frame starts with a preamble and a start frame delimiter; it then continues with the MAC frame length and the MAC frame itself as seen in figure 2.2. The overhead for each PHY packet is only 4+1+1 133 tild 4.5%; when using the maximum transmission speed of 250kbit/s, each frame can be sent 133byte in 250kbit/s = 4.265ms. Furthermore, it can also operate in the 868MHz and 915MHz bands, maintaining the 250kbit/s transmission rate by using Offset quadrature phase-shift keying (O-QPSK).

Several network layer protocols are implemented on top of IEEE 802.15.4. The two that will be examined are 6LoWPAN and ZigBEE.

2.4.5.3 ZigBee

is a communication standard initially developed for home automation networks; it has several different protocols designed for specific areas such as lighting, remote control, or health care [27, 6]. Each of these protocols uses their own addressing with different overhead; however, there is also the possibility of direct IPv6 addressing. Then, the overhead is the same as for uncompressed 6LoWPAN, as seen in figure 2.5.

A new standard called ZigBee 3.0 aims to bring all these standards together under one roof to simplify the integration into IoT. The release date of this standard is set to Q4 2015.

2.4.6 LoaraWAN

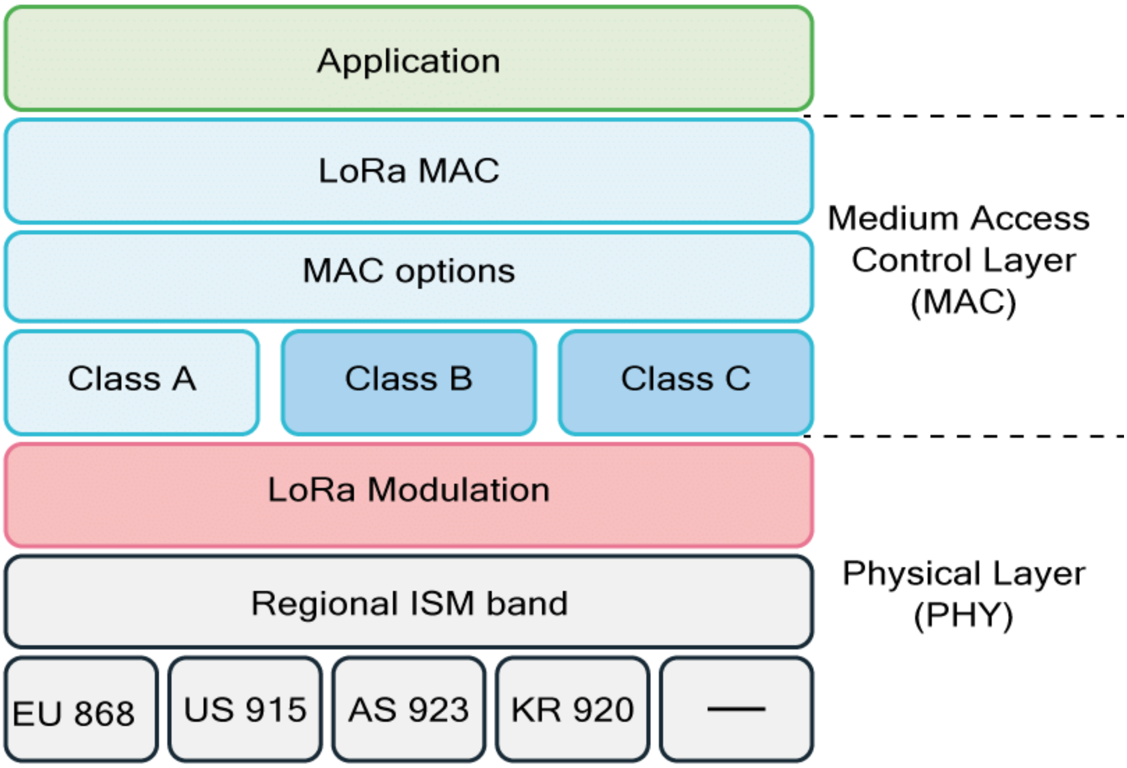


Figure 12: uhuhuh.

LoRa (Long Range) is a proprietary spread spectrum modulation technique by Semtech. It is a derivative of Chirp Spread Spectrum (CSS). The LoRa physical layer may be used with any MAC layer; however, LoRaWAN is the currently proposed MAC which operates a network in a simple star topology.

As LoRa is capable to transmit over very long distances it was decided that LoRaWAN only needs to support a star topology. Nodes transmit directly to a gateway which is powered and connected to a backbone infrastructure. Gateways are powerful devices with powerful radios capable to receive and decode multiple concurrent transmissions (up to 50). Three classes of node devices are defined: (1) Class A enddevices: The node transmits to the gateway when needed. After transmission the node opens a receive window to obtain queued messages from the gateway. (2) Class B enddevices with scheduled receive slots: The node behaves like a Class A node with additional receive windows at scheduled times. Gateway beacons are used for time synchronisation of end-devices. (3) Class C end-devices with maximal receive slots: these nodes are continuous listening which makes them unsuitable for battery powered operations.

2.4.6.1 ALIANCE

1) Class-A

1.1) Uplink LoRa Server supports Class-A devices. In Class-A a device is always in sleep mode, unless it has something to transmit. Only after an uplink transmission by the device, LoRa Server is able to schedule a downlink transmission. Received frames are de-duplicated (in case it has been received by multiple gateways), after which the mac-layer is handled by LoRa Server and the encrypted application-playload is forwarded to the application server.

1.2) Downlink LoRa Server persists a downlink device-queue for to which the application-server can enqueue downlink payloads. Once a receive window occurs, LoRa Server will transmit the first downlink payload to the device.

1.3) Confirmed data LoRa Server sends an acknowledgement to the application-server as soon one is received from the device. When the next uplink transmission does not contain an acknowledgement, a nACK is sent to the application-server.

Note: After a device (re)activation the device-queue is flushed.

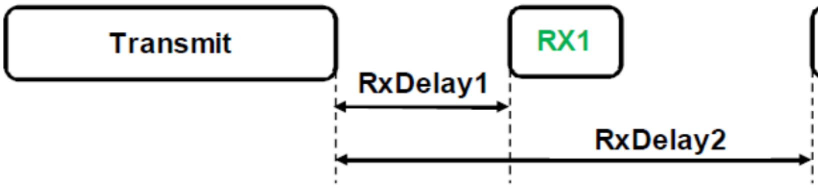


Figure 13: Class A.

2) Class-B LoRa Server supports Class-B devices. A Class-B device synchronizes its internal clock using Class-B beacons emitted by the gateway, this process is also called a beacon lock. Once in the state of a beacon lock, the device negotioates its ping-interval. LoRa Server is then able to schedule downlink transmissions on each occuring ping-interval.

2.1) Downlink LoRa Server persists all downlink payloads in its device-queue. When the device has acquired a beacon lock, it will schedule the payload for the next free ping-slot in the queue. When adding payloads to the queue when a beacon lock has not yet been acquired, LoRa Server will update all device-queue to be scheduled on the next free ping-slot once the device has acquired the beacon lock.

2.2) Confirmed data LoRa Server sends an acknowledgement to the application-server as soon one is received from the device. Until the frame has timed out, LoRa Server will wait with the transmission of the next downlink Class-B payload.

Note: The timeout of a confirmed Class-B downlink can be configured through the device-profile. This should be set to a value less than the interval between two ping-slots.

2.3) Requirements

Device The device must be able to operate in Class-B mode. This feature has been tested against the develop branch of the Semtech LoRaMac-node source.

Gateway The gateway must have a GNSS based time-source and must use at least the Semtech packet-forwarder version 4.0.1 or higher. It also requires LoRa Gateway Bridge 2.2.0 or higher.

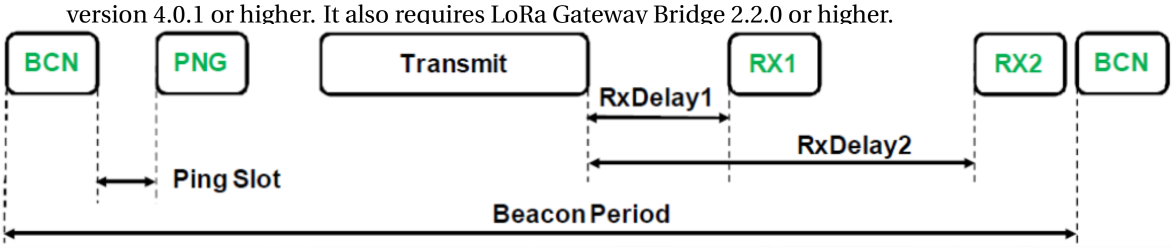


Figure 14: Class B.

3) Class-C

3.1) Downlink LoRa Server supports Class-C devices and uses the same Class-A downlink device-queue for Class-C downlink transmissions. The application-server can enqueue one or multiple downlink payloads and LoRa Server will transmit these (semi) immediately to the device, making sure no overlap exists in case of multiple Class-C transmissions.

3.2) Confirmed data LoRa Server sends an acknowledgement to the application-server as soon one is received from the device. Until the frame has timed out, LoRa Server will wait with the transmission of the next downlink Class-C payload.

Note: The timeout of a confirmed Class-C downlink can be configured through the device-profile.

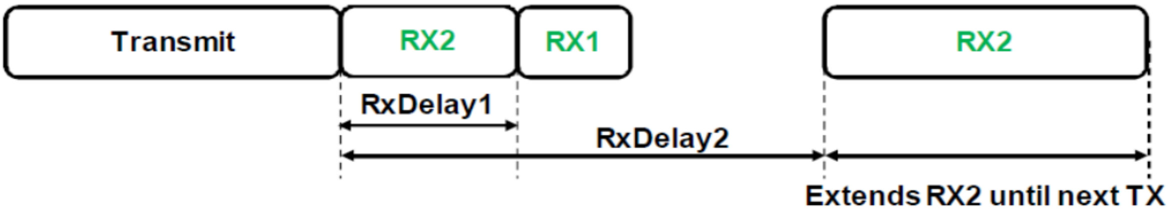


Figure 15: Class C.

2.4.6.2 SEMTECH

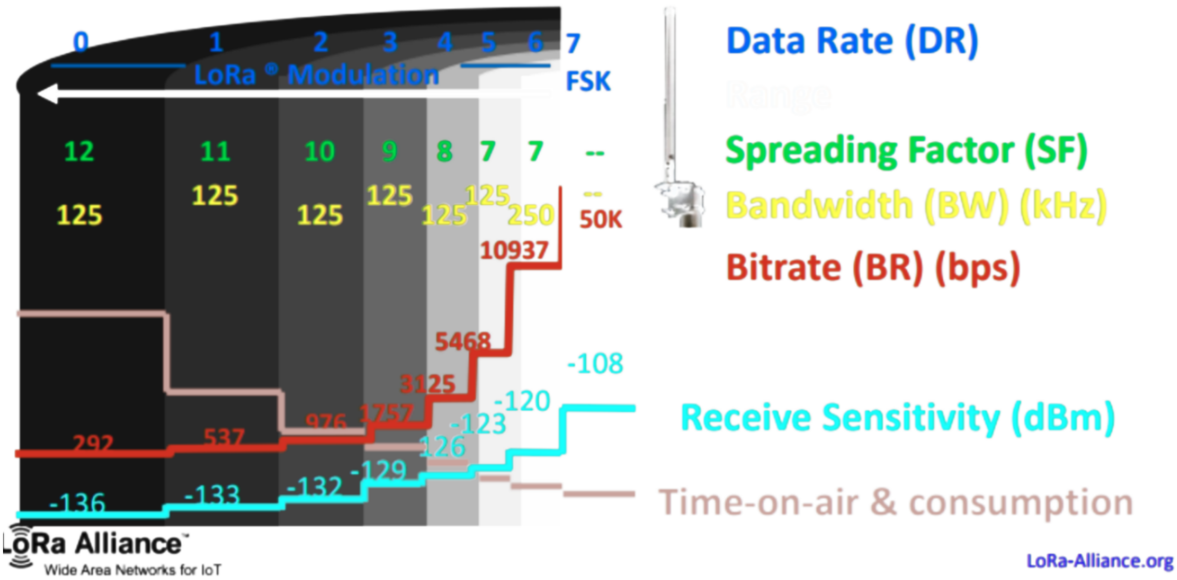


Figure 16: LoraWan Parameters.

LoRa has four configurable parameters:

BW **Bandwidth:** Bandwidth (BW) is the range of frequencies in the transmission band. Higher BW gives a higher data rate (thus shorter time on air), but a lower sensitivity (due to integration of additional noise). A lower BW gives a higher sensitivity, but a lower data rate. Lower BW also requires more accurate crystals (less ppm). Data is send out at a chip rate equal to the bandwidth. So, a bandwidth of 125 kHz corresponds to a chip rate of 125 kcps. The SX1272 has three programmable bandwidth settings: 500 kHz, 250 kHz and 125 kHz. The Semtech SX1272 can be programmed in the range of 7.8 kHz to 500 kHz, though bandwidths lower than 62.5 kHz requires a temperature compensated crystal oscillator (TCXO).

CF **Carrier Frequency:** Carrier Frequency (CF) is the centre frequency used for the transmission band. For the SX1272 it is in the range of 860 MHz to 1020 MHz, programmable in steps of 61 Hz. The alternative radio chip Semtech SX1276 allows adjustment from 137 MHz to 1020 MHz.

CR **Coding Rate:** Coding Rate (CR) is the FEC rate used by the LoRa modem and offers protection against bursts of interference. A higher CR offers more protection, but increases time on air. Radios with

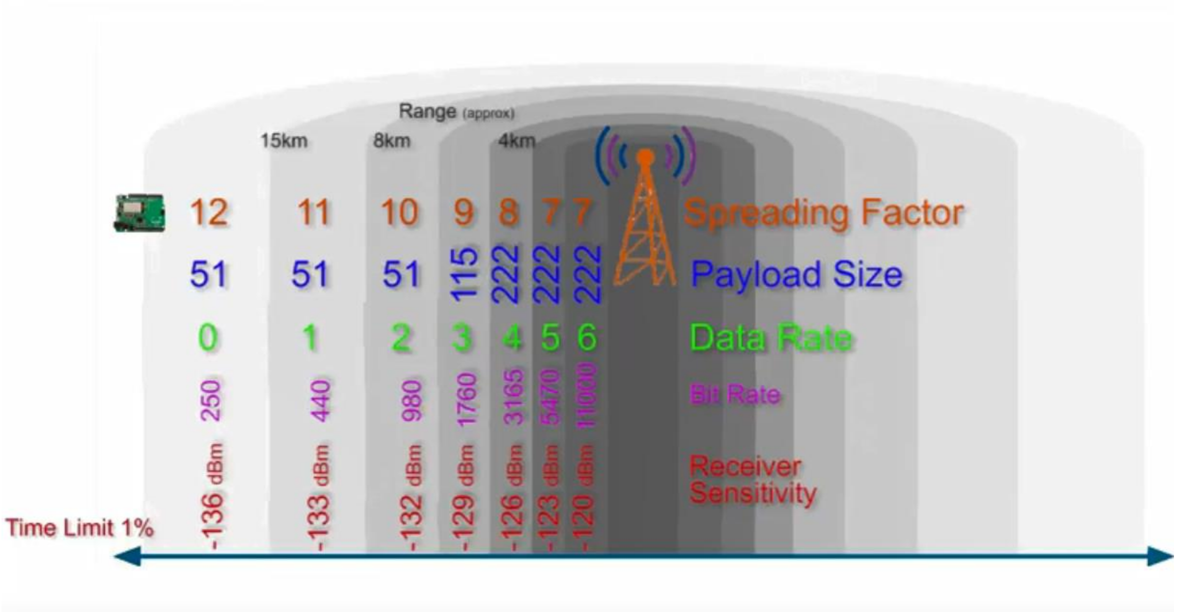


Figure 17: .

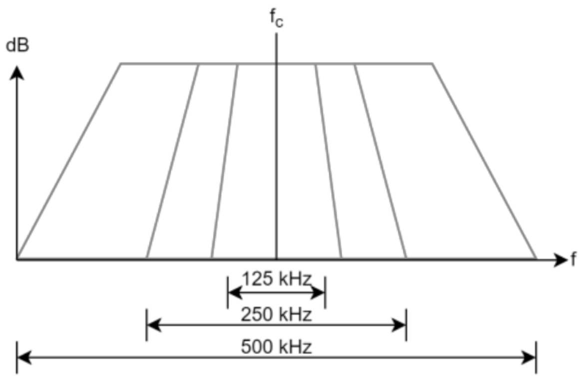


Figure 18: .

different CR (and same CF/SF/BW), can still communicate with each other. CR of the payload is stored in the header of the packet, which is always encoded at 4/8.

SF **Spreading Factor:** SF is the ratio between the symbol rate and chip rate. A higher spreading factor increases the Signal to Noise Ratio (SNR), and thus sensitivity and range, but also increases the air time of the packet. The number of chips per symbol is calculated as 2 sf . For example, with an SF of 12 (SF12) 4096 chips/symbol are used. Each increase in SF halves the transmission rate and, hence, doubles transmission duration and ultimately energy consumption. Spreading factor can be selected from 6 to 12. SF6, with the highest rate transmission, is a special case and requires special operations. For example, implicit headers are required. Radio communications with different SF are orthogonal to each other and network separation using different SF is possible.

Tx **Transmition power:**

Payload **Payload legth:**

LoRa has four metrics performance:

- Radio
  - SINR **signal-to-interference-plus-noise ratio:**
  - SIR **Signal-to-Interference Ratio:**
  - SNR **Signal-to-noise ratio:**
  - ToA **Time on Air:**
  - PL **Path Loss:**
- Network
  - PDR **Packet delivery ratio:**
  - BER **Bit error rate:**

PRR **Packet Reception Ratio:**

$$SF = log_2 \frac{R_c}{SR} \tag{2.1}$$

$$\mathbf{T_s} = \frac{2^{SF}}{BW_{[Hz]}} \tag{2.2}$$

$$SR_{\mathbf{[sps]}} = \frac{BW}{2^{SF}} \tag{2.3}$$

$$DR_{\mathbf{[bps]}} = SF * \frac{BW_{[Hz]}}{2^{SF}} * CR \tag{2.4}$$

$$BR_{\mathbf{[bps]}} = SF * \frac{\frac{4}{4+CR}}{\frac{2SF}{BW}} \tag{2.5}$$

$$Sen_{\mathbf{[dBm]}} = -174 + 10\log_{10} BW + NF + SNR \tag{2.6}$$

$$SNR_{\mathbf{[dB]}} = 20.log(\frac{S}{N}) \tag{2.7}$$

$$SNR_{\mathbf{[dB]}} = 20.log(\frac{S}{N}) \tag{2.8}$$

$$BER_{\mathbf{[bps]}} = \frac{8}{15}.\frac{1}{16}.\sum k = 216-1^k(\frac{16}{k})e^{20.SINR(\frac{1}{k}-1)} \tag{2.9}$$

$$BER_{\mathbf{[bps]}} = 10^{\alpha.e^{\beta SNR}} \tag{2.10}$$

$$PER_{\mathbf{[pps]}} = 1-(1-BER)^{n_{bits}} \tag{2.11}$$

$$PRR = (1-BER)^L \tag{2.12}$$

$$\tag{2.13}$$

$$RSSI = Tx_{power}.\frac{Rayleigh_{power}}{PL} \tag{2.14}$$

$$\mathbf{LoRa} = \frac{2^{SF}}{BW}\left((NP+4.25)+\left(SW+\max\left(\left\lceil\frac{8PL-4SF+28+16CRC-20IH}{4(SF-2DE)}\right\rceil(CR+4),0\right)\right)\right) \tag{2.15}$$

$$\mathbf{Lora} = n_s = 8 + max([\frac{8PL-4SF+8+CRC+H}{4*(SF-DE)}]*\frac{4}{CR}) \tag{2.16}$$

$$\mathbf{Lora} = \frac{1}{R_s}\left(n_{preamble}+\left(SW+max\left(\left\lceil\frac{8PL-4SF+28+16CRC-20IH}{4(SF-2DE)}\right\rceil(CR+4),0\right)\right)\right) \tag{2.17}$$

$$\mathbf{GFSK} = \frac{8}{R_{GFSK}}\left(L_{preamble}+SW+PL+2CRC\right) \tag{2.18}$$

$$\mathbf{GFSK} = \frac{8}{DR}(NP+SW+PL+2CRC) \tag{2.19}$$

$$\tag{2.20}$$

Base Station (BS)

[19] Nous avons vu en effet plus haut quil a été démontré que la méthode CSMA est plus efficace pour le traitement des faibles trafics, tandis que TDMA est nettement plus appropriée pour supporter les trafics intensesj.

2.4.7 Summary and discussion

SF		07	08	09	10	11	12	07	08	09	10	11	12	07	08	09	10	11	12
	BW	125						250						500					
07	125	x								x								x	
08			x								x								x
09				x								x							
10					x								x						
11						x													
12							x												
07	250							x								x			
08									x								x		
09		x								x								x	
10			x								x								x
11				x								x							
12					x								x						
07	500													x					
08															x				
09								x								x			
10									x								x		
11		x								x								x	
12			x								x								x

Table 2.14: uyuyuy

	LoRa[5]	SigFox[6]	NB-IoT [7]	Z-Wave[8]	Wi-Fi[9]
Cost	35e	25e	1020e	812e	< 2e
DR	<50 kbps	<100 bps	<200 kbps	<40 kbps	<300 Mbps
Autonomy	<10 years	<10 years	<10 years	<2 years	<10 days
Range (urban)	<5 km	<10 km	<1 km	<100 m	<40 m
Modulation	CSS	BPSK	QPSK	FSK	BPSK/QAM
BW	125/250 kHz	100 Hz	200 kHz	300 kHz	20/40 MHz
Frequency (EU)	868 MHz	868 MHz	LTE bands	868 MHz	2.4/5.0 GHz
Spectrum Cost	Free	Free	Very High	Free	Free
Max. msg/day	Unlimited	140(), 4()	Unlimited	Unlimited	Unlimited
Max. payload	243 bytes	12(), 8() bytes	1600 bytes	64 bytes	64 KB

Table 2.15: Wireless technologies commonly used in smart buildings [15]

	LTE Cat.1	LTE Cat.0	LTE Cat.M	EC-GSM	NB-LTE*	NB-CIoT
Spectrum	LTE In-Band, Greenfield			GSM In-Band, Greenfield	Greenfield	Greenfield
Release Date/Commercialization	2009	2014	2015/2016	2016/2017	*2016/2017*	*2016/2017*
3GPP Release	Rel-8	Rel-12	Rel-13	Rel-13	*Rel-13/14*	*Rel-13/14*
Peak Data Rate	DL: 10Mbps UL: 5Mbps	DL: 1Mbps UL: 1Mbps	DL: 1Mbps UL: 1Mbps	DL: 74kbps UL: 74kbps	*DL: 128kbps?* *UL: 64kbps?*	DL: 32kbps UL: 48/14.7kbps
System Bandwidth	20MHz	20MHz	1.4MHz	200kHz	200kHz	200kHz
LPWA Network	No	No	No	Yes	Yes	
Link Budget Target	140dB	140dB	155dB	164dB	*164dB*	164dB
Network Upgrade	No Need	SW Upgrade	To be determined	Yes (HW/SW?)	Yes (HW/SW?)	New Network Clean Slate overlaid with GSM network

Figure 19: .

Characteristics	**6LoWPAN**	**LoRaWAN**	**SigFox**	**NB-IoT**	**INGENU**	**TELENSA**
Proprietary			✓			
Standar	IETF	LoRa Alliance		3GPP		
CF$_{[\mathrm{MHz}]}$	902-929 868-868.6	902-928 863-870 and 434	902 868			
Channels	0016 for 2400 0010 for 915 0001 for 868.3	80 for 915 10 for 868 and 780	25			
BW$_{[\mathrm{MHz}]}$	0005 for 2400 0002 for 915 0600 for 868.3	0.125 and 0.50 for 915 0.125 and 0.25 for 868 and 780	0.0001-0.0012			
DR$_{[kbps]}$	0250 for 2400 0040 for 915 0020 for 868.3	0.00098-0.0219 for 915 0.250-0.05 for 868 and 780	0.1-0.6			
Modulation	QPSK for 2400 BPSK for 915 BPSK for 868.3	LoRa for 915 LoRa and GFSK for 868 and 780 CSS unslotted ALOHA	BPSK and GFSK unslotted ALOHA	QSPSK	CDMA-like	
CR$_{[d\mathrm{B}m]}$	-085 for 2400 -092 for 915 -092 for 868.3	-137	-137			
Topology		Star, Stars	Star		Star, Tree	Star
ADR		✓	✗		✓	✗
PL		<250B (depends on SF)	12B(UL),8B(DL)		10KB	
Handover		Multi BS	Multi BS			
Security		AES 128b	✗		16B hash, AES 256b	
LS		✓	✗		✗	✗
FEC		AES 128b	✗		✓	✓
Range	10-100 m	5-15 km	10-50 km	1Km		
Battery lifetime	1-2 years	<10 years	<10 years	<10 years		
Uplink			100bps			
Downlink			8 bytes/msg			
Cost		35e	25e	1020e		
max msg/day		Unlimited	140(UL),4(DL)	Unlimited		
max Payload		243B	12(UL),8(DL)	1600B		

Table 2.16: LPWan Characteristics [16]

	SIGFOX	**LORAWAN**	**INGENU**	**TELENSA**
Modulation	UNB DBPSK(UL), GFSK(DL)	CSS	RPMA-DSSS(UL), CDMA(DL)	UNB 2-FSK
Band	S UB -GH Z ISM:EU (868MHz), US(902MHz)	S UB -GH Z ISM:EU (433MHz 868MHz), US (915MHz), Asia (430MHz)	ISM 2.4GHz	S UB -GH Z bands including ISM:EU (868MHz), US (915MHz), Asia (430MHz)
DR	100 bps(UL), 600 bps(DL)	0.3-37.5 kbps (L O Ra), 50 kbps (FSK)	78kbps (UL), 19.5 kbps(DL)	62.5 bps(UL), 500 bps(DL)
Range	10 km (URBAN), 50 km (RURAL)	5 km(URBAN), 15 km (RURAL)	15 km (URBAN)	1 km (URBAN)
Channels	360 channels	10 in EU, 64+8(UL) and 8(DL) in US plus multiple SFs	40 1MHz channels, up to 1200signals per channel	multiple channels

Table 2.17: [17]

Standard	**802.15.4k**	**802.15.4g**	**Weightless-W**	**Weightless-N**	**Weightless-P**	**DASH 7 Alliance**
Modulation	DSSS, FSK	MR-[FSK, OFDMA, OQPSK]	16-QAM, BPSK, QPSK, DBPSK	UNB DBPSK	GMSK, offset-QPSK	GFSK
BW	ISM S UB -GH Z, 2.4GHz	ISM S UB -GH Z, 2.4GHz	TV white spaces 470-790MHz	ISM S UB -GH Z EU (868MHz), US (915MHz)	S UB -GH Z ISM or licensed	UB -GH Z 433MHz, 868MHz, 915MHz
DR	1.5 bps-128 kbps	4.8 kbps-800 kbps	1 kbps-10 Mbps	30 kbps-100 kbps	200 bps-100kbps	9.6,55.6,166.7 kbps
Range	5 km (URBAN)	up to several kms	5 km (URBAN)	3 km (URBAN)	2 km (URBAN)	0-5 km (URBAN)
MAC	CSMA/CA, CSMA/CA or A LOHA with PCA	CSMA/CA	TDMA/FDMA	slotted A LOHA	TDMA/FDMA	CSMA/CA
Topology	star	tar, mesh, peer-to-peer	star	star	star	tree, star
PL	2047B	2047B	>10B	20B	>10B	256B
Security	AES 128b	AES 128b	AES 128b	AES 128b	AES 128/256b	AES 128b
Forward error correction	✓	✓	✓	✗	✓	✓

Table 2.18: [17]

Phy protocol	**IEEE 802.15.4**	**BLE**	**EPCglobal**	**Z-Wave**	**LTE-M**	**ZigBee**
Standard		IEEE 802.15.1				IEEE 802.15.4, ZigBee Alliance
BW **(MHz)**	868/915/2400	2400	860-960	868/908/2400	700-900	
MAC	TDMA, CSMA/CA	TDMA	ALOHA	CSMA/CA	OFDMA	
DR **(bps)**	20/40/250 K	1024K	varies 5-640K	40K	1G (up), 500M (down)	
Throughput				9.6, 40, 200kbps		
Scalability	65K nodes	5917 slaves	-	232 nodes	-	
Range	10-20m	10-100m				
Addressing	8|16bit	16bit				

Table 2.19: IoT cloud platforms and their characteristics [7]

	802.15.4	**802.15.4e**	**802.15.4g**	**802.15.4f**
CF	2.4Ghz (DSSS + oQPSK) 868Mhz (DSSS + BPSK) 915Mhz (DSSS + BPSK)	2.4Ghz (DSSS + oQPSK, CSS+DQPSK) 868Mhz (DSSS + BPSK) 915Mhz (DSSS + BPSK)	2.4Ghz (DSSS + oQPSK, CSS+DQPSK) 868Mhz (DSSS + BPSK) 915Mhz (DSSS + BPSK)	2.4Ghz (DSSS + oQPSK,CSS+DQPSK) 868Mhz (DSSS + BPSK) 915Mhz (DSSS + BPSK) 3∼10Ghz (BPM+BPSK)
DR	Upto 250kbps	Upto 800kbps	Up to 800kbps	
Differences	-	Time sync and channel hopping	Phy Enhancements	Mac and Phy Enhancements
PL	127 bytes	N/A	Up to 2047 bytes	N/A
Range	1 75+ m	1 75+ m	Upto 1km	N/A
Goals	General Low-power Sensing/Actuating	Industrial segments	Smart utilities	Active RFID
Products	Many	Few	Connode (6LoWPAN)	LeanTegra PowerMote

Table 2.20: IEEE 802.15.4 standards [18]

SF	Sensitivity[dBm]	DR[kb/s]		
6	-118	9.38		
7	-123			
8	-126			
9	-129			
10	-132			
11	-134.5			
12	-137			

Table 2.21: hghg

SF/BW	**125kHz**		**250kHz**		**500kHz**		
-	Sensitivity [dBm]	Bit Rate [kb/s]	Sensitivity	Data Rate	Sensitivity	Data Rate	
6	-118		-115		-111		
7	-123	5.468	-120		-116		
8	-126	3.125	-123		-119		
9	-129	1.757	-125		-122		
10	-132	0.976	-128		-125		
11	-133	0.537	-130		-128		
12	-136	0.293	-133		-130		

Table 2.22: Receiver sensitivity [dBm]

DR	Modulation			Max transmission unit		BR
	SF	BW [kHz]	CR	Total [B]	Payload [B]	x kbit/s
0	12	125	4/6	64	51	0.25
1	11	125	4/6	64	51	0.44
2	10	125	4/5	64	51	0.98
3	9	125	4/5	128	115	1.76
4	8	125	4/5	255	242	3.125
5	7	125	4/5	255	242	5.47
6	7	125	4/5	255	242	11
7		125	4/5	255	242	50

Table 2.23: oioioi

Feature	Wi-Fi	802.11p	UMTS	LTE	LTE-A
Channel MHz	20	10	5	1.4, 3, 5, 10, 15, 20	<100
Frequency band(s) GHz	2.4 , 5.2	5.86-5.92	0.7-2.6	0.7-2.69	0.45-4.99
BR **Mb/s**	6-54	327	2	<300	<1000
Range km	<0.1	<1	<10	<30	<30
Capacity	Medium	Medium	✗	✓	✓
Coverage	Intermittent	Intermittent	Ubiquitous	Ubiquitous	Ubiquitous
Mobility support km/h	✗	Medium	✓	<350	<350
QoS support	EDCA Enhanced Distributed Channel Access	EDCA Enhanced Distributed Channel Access	QoS classes and bearer selection	QCI and bearer selection	QCI and bearer selection
Broadcast/multi-cast support	Native broadcast	Native broadcast	Through MBMS	Through eMBMS	Through eMBMS
V2I support	✓	✓	✓	✓	✓
V2V support	Native (ad hoc)	Native (ad hoc)	✗	✗	Through D2D
Market penetration	✓	✗	✓	✓	✓
DR	<640 kbps	250 kbps	106424 kbps	✓	✓

Table 2.24: An example table.

2.5 IoT SDN

SDN is the equipement vertualisation, NFV is the function vertualisation

2.5.1 Appication

2.5.2 Controle

2.5.3 Data

2.5.4 Summary and discussion

2.6 IoT Security

2.6.1 Application

2.6.1.1 MQTTS

2.6.1.2 Blockchain

Blockchain Layers

- Transaction & contract layer
- Validation layer (forward validation request)
- Block Generation Layer (PoW,PoC, PoA PoS, PBFT)
- Distribution Layer

Consensus algorithms

- Proof of Work (PoW)
- Proof of Capacity (PoC)
- Proof of Authority (PoA)
- Proof of Stake (PoS)
- Proof of Bizantine Fault Tolerant (PBFT)

2.6.2 Network

2.6.3 MAC/Radio

2.6.3.1 LoraWan

2.6.3.2 Zigbee

2.6.3.3 802.15.4

2.6.4 Summary and discussion

3 | Testbed

Testbed

Contents

Abstract

3.1 Introduction [1]

3.1.1 problem statement

3.1.2 Background

Internet of Things (IoT) is a concept aiming at connecting all things to the Internet [1]. The different kinds of devices range from simple sensor devices to complex machines such as industry robots. Home automation has been available for a few years in the forms of timers and remotely controlled devices, such as lights, garage door, and climate control equipment. Also in the industry and workplace, there are current systems that have some of the functionality of IoT, e.g, sensors in robots and machines which keep track of the system status so that maintenance can be scheduled at the right time. However, these systems or sensors rarely communicate with each other or make decisions based on other sensor values; instead they depend on input from a user. In the same way cellphones connected people and made them constantly connected to the Internet, IoT will connect devices and make them constantly connected to the Internet [2]. In theory, this could lead to a future with autonomous technology all around us. The benefits could be huge as it would save time and energy for both the individual at home and for the industry [3]. IoT could be used in industry to automate power-heavy tasks to run when the electricity price is low. This principle can also be applied for the home user with laundry machines and charging of e.g. electric cars. This practice would lead to reduced energy consumption and thus a reduced environmental footprint. i3tex AB wants to investigate potential fields of applicability of this upcoming technology. i3tex AB has customers in the automotive, communication, and pulp industries; those customers have made inquiries on how to integrate IoT and sensor networks into production. As technology evolves, size and energy consumption of the IoT devices will decrease and computation power will increase [4]. This reduction in size and energy consumption, together with the increased computing power, will open up new fields for IoT. Thus, i3tex AB want to have an IoT platform to present to their current and potential customers. The interest in IoT is rapidly increasing, and thus, in the near future, the number of devices connected to the Internet is expected to increase rapidly. To support this huge increase in both number of connected devices and the sheer amount of data that will be sent over both wired and wireless networks, the communication technology must be ready [5].

3.1.3 Purpose (Goal)

The purpose of this project is to find and examine a communication method for devices that are made to be a part of IoT. This will be done by examining the available technologies and then developing a prototype based on the findings, which will be used for examining the communication method. This project will examine the physical, link, and network layers [6, 7] of the Open Systems Interconnection model (OSI model) [8], in order to find suitable technologies on the market. As IoT is still only defined as a concept, there are several technologies to take into consideration and examine in further detail. The prototype will be delivered to i3tex AB together with appropriate documentation, e.g. technical specification, hardware manual, software manual, and API specification.

3.1.4 Limitations

To be able to achieve the project goal within the available time, limitations need to be defined in the three main areas of: Operating System (OS), hardware, and communication method. The OS will not be custom-made, but rather selected amongst those already on the market. Thus, to simplify the hardware selection, only those OSs which already have hardware support that meets the requirements will be taken into consideration. Furthermore, support for either 6LoWPAN, ZigBee, or Bluetooth Low Energy (BLE) as communication method is required, since development to make those standards available is outside the scope of the project. On the hardware side, the limitations will be to only use existing devices and parts as there will be no time for developing hardware or Printed Circuit Boards (PCBs). However, the hardware does not need to have an integrated radio transceiver, but needs to support at least one transceiver supporting IEEE802.15.4 [6]. Thus, communication methods will be primarily selected from specifications building on the IEEE 802.15.4.

3.1.5 Method

To ensure that the right technologies were selected and investigated, the first phase of the project was a literature study. The study served as a foundation when developing and performing the evaluation of the communication methods. At the end of the phase, a requirements specification was formulated to serve as a platform for the next phase. After the literature study, a selection process was performed, where the most promising technologies that met the requirements were examined in further detail and brought into the development phase. This process included the selection of development tools and other decisions bound to the product development. In the development phase, the chosen set-up was configured and assembled to prepare for testing; it was then tested according to throughput, range, latency, and energy consumption. Throughput was measured in kilobyte per second (KB/s) and tested by transferring data of different sizes in both congested and uncongested network set-ups to simulate real world and lab environments. The same set-up was used to measure the latency of a transmission, which was measured in microseconds (ms). Range was calculated instead of measured, with meters (m) as the unit. The power consumption was measured in watts (W). Each week a meeting with the company supervisors was performed, to keep the work on the right track. Here, feedback was be given and other issues and questions handled.

3.2 Related work

3.3 Background

The goal of the implementation phase is to have a working prototype for future assessment. To make the process of implementing the prototype possible, the first part of the implementation process will be to create a set of requirements. When these are set, the process will continue by comparing the data from chapter 2 to find the candidates that fulfil the requirements. After the technologies are selected, the process will continue with setting up the workspace, which includes the platform for development and the required tools to build, debug and test the prototype. Finally, when these three steps have been performed, the next step will be to start with the actual prototype development.

Requirements

As the time dedicated for development is limited, the requirements have to make sure that the development process does not run into any major obstacles. All parts of the total prototype need to fit together seamlessly. However, the hardware platform and the operating system are tied most closely together. Therefore, they need requirements that complement each other, so that they can act as a platform for software development. Naturally, both the hardware and the operating system requirements might have to be altered slightly to enable the best match.

3.3.1 Hardware

Each platform examined in chapter 2 has different strengths and weaknesses. When looking at the MCU, OpenMote has a ARM Cortex-M3 which is more powerful compared to the other two alternatives: it features a 32bit 32MHz core with 32KB RAM and 512KB flash memory compared to the MSP430 16bit 25MHz core with 10KB/100KB memory configuration. In terms of peripherals all three platforms are comparable, with similar amount of DAC, GPIO pins, and external busses. All of the platforms have a temperature sensor and an accelerometer, but OpenMote also features an light/uv-light sensor and a voltage sensor built into the MCUs ADC. The MSB430 platform has a somewhat lower power consumption in active RF mode thanks to the less power-hungry sub-GHz transceiver. On the other hand, the less powerful MSP430 MCU has a better deep sleep power consumption, but as the radio is not integrated in the chip as it is in the CC2538 SoC, that advantage is offset by the external transceiver. Comparing the transceivers, there are two 2.4GHz models and one subGHz model; the sub-GHz CC1100 has a higher transmit power of 10dBm compared to 0dBm for CC2420 and 7dBm for CC2538. Also the sensitivity is similar for all the alternatives but gives a slight advantage to CC2538 with -97dBm compared to -95dBm and -93dBm for CC2420 and CC1100. Using these numbers and Friis range equation (equation 4.1) the range of each transceiver with a Fade margin (FM) of 20dB can be seen in figure 3.1. The benefits of working with lower frequencies can clearly be seen as the theoretical range of CC1100 is almost 3 times longer than the 2.4GHz transceivers.

Adding all this information together, the choice of platform will land on OpenMote with the CC2538SoC. It both has a MCU with more memory and better performance, and a transceiver with really good characteristics both in terms of energy consumption and range. Also OpenMote is the only option that can act as a border router using OpenBase; it lets the SoC interface with USB, UART, JTAG, and Ethernet, which enables the standalone border router mode without the need to be connected to a computer or other hardware. The OpenBattery extension lets the SoC operate as a node in a mesh network and provides a dual AAA battery slot connected to the PCB.

Resume

Transceiver with IEEE 802.15.4 or IEEE 802.15.1 Integrated sensor/sensors MCU with low power mode under 5MicroA Wakeup from low power mode with timer Border router ability Joint Test Action Group (JTAG) support

3.3.2 Operating system

As a modern operating system can be compiled to match almost any hardware, the most important thing to have in mind is the out-of-the-box hardware support. Only RIOT and Contiki have full support for the ARM Cortex-M3 of the considered operating systems and thus both TinyOS and freeRTOS are directly eliminated as developing the support would take too much time. Compared to RIOT, Contiki also has full driver support for the sensors and transceiver, which should decrease the implementation time significantly. When compiled into binary form, RIOT uses less RAM and ROM and thus probably is a bit faster compared to Contiki, which could be important if the application consumes much resources. The lower memory usage might also give RIOT an advantage in being future-proof. Contiki also has support for soft real-time scheduling compared to the hard real-time scheduling of RIOT; this is however not crucial, as the software that will be running on the OS does not have any hard real-time constraints. Both RIOT and Contiki have support for 6LoWPAN but no support for either ZigBee or BLE; this is due to the fact that these are proprietary stacks. Support could be added but would take some time to customize for the given OS. What gives Contiki the largest advantage is that it also have border router software ready for deployment, which in the RIOT case would have to be developed. All in all, as the project has such a limited time frame, Contiki will be selected as the OS; this, mainly because Contiki comes with most advantages time-wise; this choice means that the focus of the software development will be the creation of a test and evaluation system.

Resume

Support for the SoC/MCU and transceiver 6LoWPAN, ZigBee or BLE stack Soft real-time RAM and ROM footprint matching the hardware

3.3.3 Communication protocol

The OpenMote platform has a IEEE 802.15.4 transceiver and thus supports both ZigBee and 6LoWPAN; this means that BLE is not an option. As ZigBee does not have full IPv6 support yet and is not integrated into Contiki, the natural choice is 6LoWPAN. This choice will not only save some development time but also enables evaluation of the header compression. As seen in figure 3.2, the 6LoWPAN stack in Contiki will replace the IP stack while maintaining the same functionality. As the functionality is the same, TCP and HTTP will work with 6LoWPAN, but including them in the source increases the OS build size considerably. On top of the UDP layer, Contiki also has a working implementation of CoAP that can be used for retrieving data from the nodes in a power efficient manner. CoAP is a stateless protocol that uses the HTTP response headers to achieve a very low overhead in transmissions while using application level reliability methods to ensure packet delivery.

Resume

IPv6 addressing support Existing OS support Network type is mesh UDP

3.3.4 Workspace and tools

The ARM Cortex-M3 chip that OpenMote and CC2538 is built upon requires the GCC ARM Embedded compiler. This tool-chain is free and runs on both Linux, OSX, and Windows; however, there is no bundled development application so a secondary application for programming is needed. In Windows, there are several Integrated Development Environments (IDEs) such as IAR Workbench ARM [30], Code Composer Studio [31] and the Eclipse plug-in ARM DS-5 [32, 33]; these IDEs use various proprietary toolchains and have a price tag ranging from free to several thousand SEK. Most of the IDEs also have a code size restriction for the free versions. To minimize the costs, the development machine development machine used in this project will run Ubuntu 14.04 LTS, the used tool-chain is GCC ARM Embedded, and Geany is used as the code development application. To analyse the network traffic in real-time, the open source tool Wireshark is used together with a IEEE 802.15.4 packet sniffer. Together with a laptop, the packet sniffer will grant the ability to traverse the mesh network and analyse the network in real-time as it is seen by the nodes.

3.4 Proposed ...

The goal of the development process is to have a functional border router and at least two nodes to be able to test how response time and throughput differs with each hop in a mesh network. To be able to measure response time and throughput, each node needs to have a CoAP server which can respond to ping and also receive an arbitrary amount of data for throughput measurement. It is desirable for each node to be able to send information about each sensor so the project can be used as a tech-demo. The first part of the development was to set-up of the workspace and tools mentioned in section 3.1.5. Ubuntu OS was installed in a VirtualBox Virtual Machine to make it easier to duplicate and backup; this procedure gave a noticeable decrease in performance and it is recommended to have a dedicated native Ubuntu machine for this type of development. Even though Ubuntu uses an easy-to-use package system, there were some problems in finding a version of GCC ARM Embedded tool-chain that was compatible with Contikis built-in simulator Cooja [34, 35]; eventually, version 4.82 was used to successfully build Contiki. Cooja is a useful tool for testing and debugging network configurations but does not have support for the CC2538 MCU; instead, nodes called Cooja Motes are simulated with generic hardware. As Cooja is written in Java and runs in a JVM, Oracle Java 1.8 was also installed.

3.4.1 Drivers and firmware

Figure 3.3 shows an overview of the full system. The foundation is the SoC with the MCU, transceiver, and sensors. The Contiki operating system implements the soft real-time kernel together with the firmware for the SoC/MCU and the drivers for the peripherals and sensors. The last part is the communication stack, which provides TCP and UDP connectivity over 6LoWPAN. On top of the TCP and UDP protocols, HTTP and/or CoAP can be implemented. The firmware required for the OS to work properly on the hardware platform was already implemented. However, the drivers for the I 2 C bus and the sensors were not implemented. The I 2 C driver is required for the sensor drivers which in turn enables the MCU to communicate with the sensors on the OpenBattery platform.

3.4.2 CoAP server

In order to make each nodes sensor data accessible, a CoAP sever was implemented as an application running on top of Contiki. A CoAP server in general can handle any number of resources; in this implementation, one resource was made for each sensor value i.e. temperature, light, humidity, and core voltage. The temperature, light, and humidity sensors all work in a similar fashion. When their value is requested, the I 2 C bus is initialized and then a request is sent over the bus. When the response with the value arrives, that data is put into either a plain-text or JavaScript Object Notation (JSON) formatted message depending on the request and then sent back to the requester. As the core voltage sensor is part of the MCUs ADC, that value is retrieved by simply getting data from a register (a somewhat faster operation). As a buffer for testing throughput speed was also needed, a resource with a circular buffer was implemented. This resource is configured with CoAPs block-wise transfer functionality for arbitrary data size; however, the buffer in itself is only 1024Kb to allow the program variables to fit into the ultra low leakage SRAM. For testing purposes, the data could have been discarded instead of actually saved into the buffer, but then the transfer can not be verified. Resources are defined by paths as CoAP works in a very similar way as HTTP.

Each resource is registered in the server with its path, media type, and content type. When a package arrives on the CoAP port, the server starts to break down the package to be able to direct it to the right resource. It starts with verifying that the package is actually a CoAP package, and then it checks the path and sends it to the correct resource. The resource then inspects the method field in the package header to direct the incoming data to the right function. CoAP package method can be either GET, PUT, POST or DELETE. This function then inspects the request media type and answer content type so that the function can parse the request and send a correctly formatted answer. If the resource does not implement the received method, the server responds with 405 Method not allowed and if the content/media type is not supported the answer is 415 Unsupported Media Type. The content/media types are text/plain, application/json, application/exi, and application/xml.

3.4.2.1 Testing

Contiki is shipped with a simulation tool called Cooja which is written in Java; it can simulate an arbitrary number of nodes with different roles and configurations. All simulation data, such as radio packages and node serial output may be viewed through different windows and exported to various formats. Unfortunately Cooja did not have support for ARM Cortex-M3, but the general set-up was still tested by using Cooja Motes, which are nodes without specified hardware, and MSP430 nodes such as Wismote or Skymotes. With this simulator the basic understanding of the communication between nodes was gained; also, before the hardware arrived, early testing was performed to test the OS and application software.

3.4.2.2 Final prototype

The final prototype consists of four OpenBattery nodes and one OpenBase border router. Both the nodes and the border router are deployed with Contiki. Each node runs a CoAP server, described in section 3.2.2, on top of the OS in its own thread. The border router runs a router software called 6lbr that acts a as translator between Ethernet and IEEE 802.15.4 [36]. Both types of hardware are configured with a 8Hz Radio Duty Cycle (RDC) driver to keep the power consumption to a minimum. RDC is a OS driver that cycles the listening mode of the transceiver to reduce power consumption. As Contiki puts the MCU into Low Power Mode (LPM) when no function is running and the transceiver is off, the RDC driver indirectly controls when the MCU is in LPM. When using the RDC protocol, the nodes repeatedly send messages until the target node wakes up and sends an Acknowledge packet (ACK); this makes communication seamless, even though most of the time the nodes transceivers are not active. Also, an always-on RDC driver, where the transceiver is constantly listening, will be used to be able to look at the performance impact of the 8Hz RDC.

3.5 Experimentation

In this chapter the results from each type of assessment are presented. The first assessment is range, followed by response time, after that connection speed, and finally the power consumption. The only assessment that is not performed on the prototype is the range assessment.

3.5.1 Range

Range is very hard to measure without advanced equipment and isolated rooms but can be roughly estimated with equation 4.1 called Friis range equation [37]. P t is the sender transmit power, P r the receiver sensitivity, d is the distance between the antennas in meters, f is the signal frequency in hertz, and is the wavelength. G t and G r is the antenna gain for the trans- mitter and the receiver. The last term in equation 4.1, when inverted, is the Free-space path loss (FSPL) and can be expanded as shown in equation 4.3. P r (dB) = P t + G t + G r + 20 log 10 ( ) 4d c f FSPL(dB) = 20 log 10 (d) + 20 log 10 (f) 147.56 = (4.1) (4.2) (4.3) Unlike Friis range equation, the Link budget equation 4.4 also takes external loss like FM into account [38]. This is needed to make a correct estimation of the actual range as there are several things in the environment that obstructs and distorts the signal. P r = P t + G t + G r FM FSPL (4.4) Combining equation 4.1, 4.3 and 4.4 gives us the equation for the estimated distance as seen in equation 4.5. d = 10 x P t + G t + G r P r F M + 147.56 20 log 10 (f)

With this equation an estimation of the transceiver range can be made for different FMs and transmit powers. When deployed, the transceiver is configured to only accept packages with a signal strength of -70dBm and above to minimize packet loss and corruption. The antenna gain for OpenMote is 0dBi and can thus be omitted. Figure 4.1 shows a comparison between three different levels of FM: 0dB, 10dB, and 20dB. A FM of 0dB means that there is no signal loss except the FSPL and this is very hard to achieve outside of a lab environment. When increasing the FM to 10dB, which corresponds to a normal home environment, the maximum range drops to 22m. However, in these kind of environments the desired range is usually around 10m which would let the device reduce the transmit power to around 0dBm. Finally, the FM is increased to 20dB which is roughly what it would be in a office or industrial environment. The maximum range in this environment is now reduced to only 7m when transmitting at maximum power.

Response time Before measuring the response time, some theoretical estimations are needed to be able to evaluate the real values. The theoretical values are based upon the radio duty cycle (RDC) and the average response time to reach a node can thus be derived from equation 4.6, 4.8, 4.9 and 4.10. As each node only

3.5.2 Response time

Before measuring the response time, some theoretical estimations are needed to be able to evaluate the real values. The theoretical values are based upon the radio duty cycle (RDC) and the average response time to reach a node can thus be derived from equation 4.6, 4.8, 4.9 and 4.10. As each node only checks the radio every 125ms, this duration combined with the data packet send time of 4ms (equation 4.7.) and ACK send time corresponds to the worst case delivery, as the node needs to wait a whole cycle before being able to send the package the desired node. When the target node is already listening, the best case delivery time is 5ms. Thus, the average theoretical delivery time to reach any adjacent node is 67.5ms. Radio duty cycle: Transfer time: 1s = 0.125s = 125ms 8 (4.6) 133B + 4B 4ms 31.25KB/s (4.7) Worst case delivery: 125ms + 4ms + 1ms (4.8) Best case delivery: 4ms + 1ms (4.9) 130ms + 5ms = 67.5ms (4.10) 2 The delivery time is only calculating the time to send a packet over a link, but when calculating the response time, the acknowledge (ACK) response has to be included in the calculation. Each ACK also needs to wait for the target node to be awake, adding one more instance of average delivery time, resulting in 125ms in average response time. This time will multiply with each hop, resulting in equation 4.11, 4.12 and 4.13. Avg. delivery: Avg. response time: (2 ů 67.5ms) ů hops (4.11) Best case response time: (2 ů 5ms) ů hops (4.12) Worst case response time: (2 ů 130ms) ů hops (4.13) After doing a test with real nodes set-up with a 8Hz RDC with three hops, as seen in figure 4.2, the values in table 4.1 were obtained. Each node was pinged 200 times at a one minute interval to simulate some traffic on the network. What can clearly be seen in the average field of the table is that the average of 765ms is much higher than the expected average of 135ms; the difference is mainly due to the worst-case pings that in some cases had response times up to 30 seconds. However, when looking at the geometrical mean which is better at smoothing out big spikes seen in figure 4.3, the observed response time is still 265ms which is a bit longer than the expected worst case response time for one hop. Also, for two and three hops the observed average is high, but the geometrical mean shows that this is due to the spikes. The estimated response time for two hops is 270ms which as seen in the geometrical mean table 4.1 is way off by 500ms. The same observation goes for three hops where the observed geometrical mean response time is 1181ms which compared to the estimated response time of 405ms is significantly higher.

With these observations in mind, the estimation could be described much better with equation 4.14. which would result in an average response time of 266, 532 and 1064 ms for one, two and tree hops. However, this would mean that the response time is doubled for one hop and then doubled for each consequent hop making the response time exponential which should not be the case.

With the RDC disabled, i.e. the transceiver is always listening and the MCU does not go into sleep mode, the response time is completely different. As seen in table 4.2 the average response time is around 12ms per hop and the spikes seen in the response time for 8Hz RDC is gone. Furthermore, the estimated best case response time of 10ms is very close to the observed average response time. The response time also scales to the number of hops as expected and is roughly 12ms per hop. It appears there might be a problem in

3.5.3 Connection speed

Connection speeds can be measured in several ways each with their own different pros and cons. One of the most popular ways is throughput, i.e. the amount of data over the link is divided by the time it took to reach the target. However, this gives a false picture of how fast the connection actually is from the developers point of view, as the measured data does not only contain application data but also headers and checksums. IEEE 802.15.4 has a theoretical data rate of 250kb/s as seen in equation 4.15. but this is only a measure of how many bits per second the transceiver is able to output. The application data part, when using no header compression, is only 41% of the total transfer. Thus, resulting in a theoretical application data rate, also called goodput, of only 12.81KB/s. Data rate: 250kb/s = 31.25KB/s 133B 54B = 0.59 133B Theoretical goodput: 31.25KB/s ů (1 0.59) = 12.81KB/s Overhead: (4.15) (4.16) (4.17) When using CoAP as the application level protocol, each package can carry either 32 or 64 bytes of application data. In practice, the 64B mode is only applicable when sending packages between nodes on the same mesh network, as the addressing fields then can be fully compressed. When using applications outside the mesh network, each package can only carry 32B of data, resulting in a packet size of 111B as shown in equation 4.18; this does not affect the theoretical data rate but has a noticeable impact on the goodput due to the large overhead of 71as shown in equation4.19. To be able to use the full data rate, the application needs to use a protocol without handshakes, i.e. UDP, as the transceiver then can send the packets as fast as physically possible. CoAP is implemented on top of UDP and thus has a low transport layer overhead, but uses its own mechanism for handshaking, delivery and ordering. The theoretical CoAP application throughput can be estimated by looking at the average response time of the node and then add that time to the the data delivery time. Each package needs to be acknowledged before the next package is sent, and thus the node response time needs to be taken into consideration. When doing so, the throughput as calculated in equation 4.20. is only 1.64KB/s and thus the theoretical goodput is reduced from 12.81KB/s down to 0.48KB/s as shown in equation 4.21. Packet size: 133B 54B + 32B = 111B Actual overhead: 79B = 0.71 111B (4.18) (4.19)

Using the packet size of 111B together with the theoretical response time from equation 4.11 would give the results shown in figure 4.4. To verify these calculations, the same test set-up as shown in figure 4.2, which also was used in section 4.2, was used to test throughput and goodput at different number of hops. Each node was sent 1KB data each minute for 200 minutes; the time from the first package sent to the final acknowledge packet received was measured for each 1KB transmission. The first test was performed with an RDC of 8Hz and resulted in the values shown in figure 4.5. As the chart shows, the theoretical throughput and goodput is much higher than the observed values, but this is due to the fact that the actual average response time is higher than the theoretical one. With some calculations made the observed throughput and goodput are within range of what is expected, given the observed response times in table 4.1. Equation 4.22 uses the observed values to calculate the average response time, given the values in figure 4.5.

3.5.4 Power consumption

To measure power on devices that use very low power and also changes the power consumption very rapidly and frequently is not an easy task. According to the currency specification from the CC2538, the different power modes have the consumption seen in table 4.3 using the built in voltage regulator TSP6750 that switches the input voltage down from the 3V to 2.2V. The components on the OpenBattery supplied directly by the 3V batteries have the current and power consumption specifications as seen in table 4.4.

Given these power profiles, combined with the time it takes to receive and transmit packages, and retrieve a measurement, the theoretical power for one RDC cycle results in the chart seen in figure 4.7. The node starts in sleep mode using 112.81W and after 109ms wakes up and goes into RX mode where a request for a sensor value is received. The node then switches off the radio and fetches the sensor value. After the value is retrieved from the sensor, the radio is once again put in to RX mode for a Channel Clear Assessment (CCA) before entering TX mode and sending the payload. The transmission is successful and the node goes into RX mode to listen for the ACK, when it is received the node enters sleep mode again. For this cycle the average power consumption is 4.8mW which would drain the 2250mWh batteries in 19 days.

However, as the nodes have a RDC running at 8Mz, most of the time there will be no package for the node to receive and thus no measuring and transmitting, as seen in figure 4.8. This cycling reduces the average power consumption to 0.47mW, which would make the batteries last for ca 200 days. The goal is to have a node that can run for one year without having to change the batteries and to be able to do this on 2xAAA batteries with 750mAh the average consumption has to be under 257W as calculated in

To verify these assumptions, we used a Keithley 2280S power supply [39] to measure the total current draw of the prototype. The node was connected to the power supply, which was set up to make 277 measures each second with a supply voltage of 3V. Several measurements were performed. One of the most interesting ones can be seen in figure 4.9. In this picture, we can clearly see the different operating modes, as the node performs 3 transmissions during the interval. In the first transmission at the 1.6s mark, the strobing feature of the RDC protocol is seen as the package is sent 5 times before the receiving node is awake and can receive the package. In the two following transmissions, the package is delivered on the first try. As our measurement is limited to 277Hz, the current peaks when only waking up to listen for traffic are sometimes missed, and the peak value is hard to extract; but the 8Hz RDC cycle is still visible. The average power consumption for these cycles is 8mW, which would make the batteries only last for 11 days. However, when taking the average of a measuring series without any transmissions, the average goes down to 4mW, which increases the battery time to 23 days. The theoretical sleep power of 0.11mW compared to the measured of 3mW is what makes the average power consumption that high.

Reducing this power consumption by a tenfold would result in an average consumption of 0.39mW, which is closer to the theoretical average power consumption. A discovery made when measuring the power was that the nodes consumed less power when supplied with a lower input voltage. Simply by reducing the voltage from 3V to 2.6V reduced the power consumption in LPM by 15%. However, this reduction could affect the range of the nodes.

Internet of Things can be realised in several ways as there are still many viable options on the market, mainly in terms of hardware, operating systems, and communication standards. Given the recent development in the field, Thingsquare recently released a technology demo using the same practices as used in this thesis; the choices taken are on track with the latest development [40]. Also, both Google and Microsoft have announced that they are developing IoT OSs. When these products are released, it would be very interesting to compare them with Contiki. It would be exciting to see if an open-source project can surpass the commercial offerings in terms of speed, RAM and ROM footprint, and device support. Furthermore, an in-depth comparison between RIOT and Contiki would give much insight into the kind of OS practices that benefit IoT development the most. Google have also started to develop a substitute for 6LoWPAN and UDP that they have named Thread [41]. As 6LoWPAN and ZigBee, it runs on top of IEEE 802.15.4 and thus might be able to out-compete the existing implementations. Google promises lower latencies and power consumption compared to the existing technologies.

The prototype

The prototype development took more time than initially planned; mostly because of the complexity of the OS, but also due to bugs in the untested drivers. The prototype combines the technology from each field, i.e. hardware, OS, and communication protocol, and fulfils the requirements set in section 3.1.1. Even though the OS is relatively simple, compared to Linux, Windows, and OS X, understanding the mechanics of the RDC driver and the LPM driver was difficult, but necessary to be able to interpret the test results. The prototype worked very well during most of the testing, with only a few unforeseen deviations. One occurred during the power measurement, where the power consumption in low power mode tripled in one of the test series; this behaviour could not be reproduced and is therefore not included in the results. Also, in the early stages when working with the 8Hz RDC driver, packet losses over 50% were recorded for packets with more than one hop; this problem was solved, when a new version of radio driver was released by the OS development team. Selecting OpenMote to be the hardware platform together with Contiki as the OS, was a very good choice as companies are starting to build their IoT solutions around Contiki and similar hardware platforms [42, 43]. Already in the beginning of the development, several benefits were noticed; new drivers and bug-fixes were released increasing the stability and functionality of the OS. The active community around the combination of OpenMote and Contiki was really helpful when developing the drivers for the I 2 C and sensor drivers. Example projects for other platforms could be used as references, giving much insight to how the programming for this type of OS worked. It would have been interesting to examine the differences between two operating systems; not only to test which one has the better performance, but also to compare which one that has the more favourable code structure and development procedure.

3.6 Results

Collecting the data went well and were reasonably straight forward; it was easy to transition between the two different test set-ups and thus making several test scenarios. Assessments were made in the areas of range, response time, connection speed, and power consumption. In each area, the theoretical values were first calculated and then compared to the retrieved measurements; except in the range case, as the required equipment for measuring was not economically justifiable to purchase.

3.6.1 Range

The theoretical range for OpenMote when transmitting at full power in an office environment is only 7m. As measuring the range was not a viable option due to the cost of measuring equipment, only distance estimations from the placement of the nodes when maintaining a stable connection can be used as a reference. Using a map of the office and the position of the nodes the range seems to be around 10m, which would mean that the effective FM of the office is around 16dB using the always-on RDC. The FM changed a bit when using the 8Hz RDC as more packages congested the air and the range dropped to somewhere around 5m; resulting in an effective FM of 23dB. To increase the range of the transceiver, a switch to the 860MHz frequency band would be the most effective solution; with a FM of 23dB, the theoretical range would increase to 14m with the same transceiver properties, and with a FM of 16dB the range would be 31m. Usually, transceivers with a lower frequency output also have a lower power consumption while transmitting. Working in sub-GHz also gives the benefit of less interference as fewer other devices uses those frequencies. Changing to a sub-GHz band would thus decrease the power consumption and increase the range, without changing the functionality of the nodes.

3.6.2 Response time

Initially when measuring the response time the always-on RDC was used and the measured response time was very close to the theoretical value. However, when using the 8Hz RDC protocol the values started to drastically differ from the theory. This behaviour is likely to originate from the way the RDC driver predicts the next time when the target node should be awake. The procedure is called phase optimization; when enabled, the node saves the time when the node was last seen, it then uses this value to predict the next time the node should be awake based on the RDC cycle. However, this prediction is based on the nodes internal clock. As the clock can differ from those of the other nodes, misalignments seem to occur, resulting in misses when trying to reach the target node. Each misalignment increases the time it takes to reach the target node as the node then needs to strobe the package until the target nodes wakes up again. In theory, when sending strobes the target node should wake up and receive the package within one cycle (125ms); however, this is not guaranteed as other transmissions might occupy the air, further increasing the response time. If the phase optimization could be improved to guarantee the alignment between the nodes, the response time should get much closer to the theoretical value; as the time to reach the node would be maximum one cycle and the air would not be as congested by nodes sending strobes.

3.6.3 Connection speed

The connection speed, when using CoAP or any other protocol with perpacket ACK, is directly bound to the response time. IEEE 802.15.4 has a relatively low data-rate, only 250kbps, compared to other solutions, e.g. BLE (1Mbps) and WLAN (>54Mbps). As throughput is based on datarate over a longer period of time, both the overhead and the response time is needed to make a good estimation. CoAP has a very low header size compared to many other communication protocols, but due to the very small frame size, the overhead is still relatively high. As of now, the results clearly show that when a reliable transfer is desired the connection speed of IEEE 802.15.4 and CoAP is only sufficient for data exchanges around 32 bytes. When the nodes use the always-on RDC, the goodput is less than 3KB/s for one hop and is halved for every hop; however, when the 8Hz RDC is enabled, the goodput is reduced to under 0.1KB/s. Using messages without per-packet ACK, thus removing the response time from the equation, would let the nodes transfer real-time audio and maybe even highly compressed video. However, using messages without the per-packet ACK disables the reliable transmission guarantee, and thus it can only be used with data streams where packet loss is acceptable.

3.6.4 Power consumption

Making a rough estimation of the power consumption of the platform was straight forward task and so was measuring the actual consumption. When comparing, the two the values differed by a factor of 30, which was not expected. The reason probably originates from the clock interrupt which is triggered every 8ms. Initially, this interrupt was assumed to be disabled when the system entered the lower power modes, but this was not the case. As the interrupt fires at 125Hz and the time to wake up and go back to LPM is only 272s, the power spikes from these interrupts were not seen on the measuring instruments. As seen in figure 4.9, even the peaks from the listening cycles were hard to record and those lasted for at least 4ms; instead, the power consumption from the clock timer looks like an increased LPM power consumption. At the time this was discovered there was no time to fix it, but doing so should decrease the average power consumption to within the limits, granting the nodes the ability to run on battery power for a year. As no delays from calculation could be observed, the clock speed on MCU could, in all probability, have been reduced to save power on the nodes. However, this reduction would only have affected the consumption when the node was in active mode, which is only a few percent of the total cycle time. The OpenMote chip has a step-down DC-DC converter for this purpose which is switched off in LPM mode to reduce quiescent currents; however, as most of the time is spent in LPM, reducing the input voltage to 2.1V by changing battery type and removing the step-down converter would be preferable as it would reduce the power consumption. These changes could affect the range of the device, but this has to be assessed.

3.6.5 Project execution

Looking at the time plan and the milestones, as seen in Appendix A and B, each milestone matches a task or transition in the time plan. The planning report was not submitted to the examiner until the 6/2-15, which is two weeks behind schedule, exceeding the time planned for milestone M1. The first draft was submitted before deadline, but several revisions were necessary. In retrospect, the litterature study should probably have been planned in parallel with the planning report, as the information from the study helped with the report. Milestone M2 marks the switch from the literature study and selection of technology to the development phase. This milestone was met and development could begin in the following week. As seen in Appendix C, the development phase have several risks to consider. The only risk encountered in this phase was R4, as one of the hardware platforms was delivered with a broken sensor. However, this malfunction did not affect the time plan as the development could continue regardless of the malfunction. The end of the development phase was defined by milestone M3, approval of prototype, which was completed ahead of schedule granting an early transition into the assessment phase. In the assessment phase, it could be argued that risk R9 was encountered when measuring the power consumption, as the results from those measurements did not properly show the wake-ups from the clock timer. This phase contained milestone M4 and M5, of which of only M5 was done in time. The Half-time presentation, milestone M4, was performed on the 8/4-15 in the form of a meeting, where the progress, results and continuation plan were discussed. Also, a half-time version of the report was sent the 17/4-15 and approved by the examiner. Milestone M6, deliver the final prototype, was completed a few days before the set deadline which eliminated risk R11 and gave more time to work on the writing and the presentation. Both of the oral presentations were attended on the 1/6-15 to grant some experience in how the presentation and opposition are carried out, thus now following the time plan. However, there were not many presentations to watch during the planned weeks, as the presentation schedule follow the academic semesters. The presentation for this thesis was not performed until the 3/6-15, thus being two weeks behind schedule. However, it was scheduled on the first available date suggested by the institution. The final version of the report will be submitted to the examiner before the 19/6-15, thus successfully completing milestone M7.

3.7 Discussion

The purpose of the project was to find and examine a communication protocol that could be suitable for IoT applications, by investigating the current hardware, OS, and communication protocols and building a prototype from the selected choices. What can be said about the investigation is that it is difficult to examine all candidates in detail; this means that a rough selection has to be made based on initial knowledge potentially discarding good options. The general feeling is, however, that all of the examined candidates in this project were relevant and added valuable insights to the current technology status. The assessment gave relevant and interesting results that improved the understanding in what IoT can be used for, and what further areas of investigation could be. One of the most interesting areas of further investigation would be the RDC driver, as it directly affects the response time and thus also the connection speed. Even though the power consumption was not in line with the expectations, the reason has been found and can be resolved. Another conclusion is that IoT is not ready for real-time applications as the latency is much higher than expected, for the technologies assessed in this thesis, and also has a high spread. As the latency increases for each subsequent network hop and the minimum observed latency per hop is 11ms, when using the always-on RDC, this type of communication will probably only be used for applications where response time can vary greatly, without affecting the functionality. CoAP as a communication protocol shows a lot of promise when combined with 6LoWPAN and IEEE 802.15.4. It performs well given its simplicity but has one disadvantage: the large overhead which comes from the MAC addressing fields in the IEEE 802.15.4 frame. If this overhead could be reduced from the current 71% to only 30the goodput would double. A solution would be to use a similar mechanism as BLE where the packet size varies depending on application. Each node also has computing time left as the MCU is more powerful than needed for the given application; an improvement would be to use a less powerful MCU, like the ARM Cortex-M0+, to reduce the clock speed as suggested in the discussion. When looking at the future-proof aspect the later suggestion is probably the better, as the clock then could be increased if more computing power is needed. In the future, batteries will hopefully be able to store more energy, thus increasing the time between battery changes or reducing the battery size.

4 | Sentilo

Sentilo

Contents

Abstract

4.1 Introduction [20]

4.1.1 Problem Statement

From the start of the computer networks, to the mobile applications nowadays, the amount of information shared has been constantly increasing. We have all kind of devices, from the big servers in datacenters, the TVs at home, mobile phones, car sensors, ... The Internet Of Things (IOT) refers to the idea of connecting all the thingsto the Internet. By things, it refers to any ordinary object that can be useful getting information. These things should be connected by an embedded device, capable to connect to the Internet in one side, and get information from the thing on the other.

4.1.2 Background

4.1.3 Purpose (Goal)

The first objective of this thesis is to document the capabilities of the Z1 motes and the Contiki OS for the IOT, by building applications to gather data from sensors and the network capabilities both from the motes and the OS. Secondly, to build an application using the Z1 motes, and the Contiki OS, using COAP(Constrained Application Protocol) and 6LoWPan(IPv6 over Low power Wireless Personal Area Networks) to retrieve information from the motes, and connect them to Sentilo, an open source sensor and actuator platform.

4.1.4 Limitations

There has been an increased research and development for the Smart Cities. The smart cities objective is to gather information from the city, to enhance quality and performance of urban services, to reduce costs and resource consumption, and to engage more effectively and actively with its citizens. This project is intended to approach two goals, to be used as a starting point for anyone who wants to use the Contiki OS with the Z1 motes, and to build a simple application for the Smart Cities, to collect data from sensors, and sending it to an information center.

4.1.5 Method

This document is divided in two parts. First, the description of the main tools used to create the experimental environments. A description of the Contiki OS, the Z1 motes, and the protocols used to communicate, IEEE 802.14.5, 6LowPAN and CoAP. The a brief description of the sensor data collector Sentilo Secondly, a description of the environment setup, and an explanation of how it works. Finally, conclusions and future work is presented.

4.2 Related work

4.3 Background

4.3.1 Hardware: Zolertia Z1 Motes

The Z1 is a low power wireless module compliant with IEEE 802.15.4 and Zigbee protocols intended to be used for Wireless Sensor Networks. This mote has support for Tiny OS, Contiki OS, OpenWSN and RIOT. The MCU architecture is based upon the MSP430 and the radio transceiver on CC2420 architecture, both from Texas Instruments.

4.3.1.1 Peripherals ports

North Port

East Port

South Port

West Port A Z1 mote has 2 internal sensors, and using the external ports, can be connected to a variety of external sensors. The main issue about collecting data with Contiki is the lack of support for floating point numbers in the stdio library, because of the large amount of code it requires. It has floating point numbers, but those are only usefull for internal operations. If a program needs to send the decimal data to an external source, has to use integers in the stdio functions, to write into the buffers.

4.3.1.2 Internal sensors

1) Temperature Sensor The internal temperature sensor in the Z1 mote is the tmp102 sensor from Texas Instruments. This sensor is integrated with the z1 motes using the I2C interface. It can read the temperature range of -40oC to +125oC. The Contiki OS has his own library of functions that can read the sensor data, located in "platform/z1/dev/tmp102.h". To use it in a program, it has to include the library dev/tmp102.h

2) Accelerometer The internal accelerometer in the Z1 motes is the adxl345 from Analog Devices Inc. This sensor is integrated with the z1 motes using the I2C interface. The Contiki OS has his own library of functions that can read the sensor data, located in "platform/z1/dev/adxl345.h". To use it in a program, it has to include the library dev/adxl345.h The sensor has 8 different interrupts to enable and 2 pins for mapping the interrupts.

4.3.1.3 External Sensors

The Z1 motes have several ways to connect sensors. In the next chapters, there are some examples of sensors, and how to read the data.

1) Analog sensors To read the analog sensors, there is a Contiki library in platform/z1/dev/z1phidgets.h This library reads the values of 4 of the pins of the north ports, and returns a 16 bit register, representing the value. It uses a 12bits A/D converter, so the min value is 0 and the max in 4095.

1.1) Precision Light Sensor The precision light sensor used as an example is the Phidget P/N 1127 sensor. This sensor is an analog sensor that measures light intensities of up to 1000 lux. It is a non-radiometric sensor. The output value does not depend on the input voltage, but the input voltage will limit the maximum measurement value. The sensor can be connected to the north port of the Z1 motes, into the 3V port or the 5V port.

In Fig.60, to read the value of the sensor, the phidgets library from Contiki is used. After a raw read, the value is transformed to lux, knowing the maximum value of the A/D converter is 4095, and the maximum value the sensor can give is 1000 lux. (In this case is connected to 5V)

1.2) Force Sensor The force sensor used used as an example is the Phidget P/N 1106 sensor. This force sensor can be used as a button for human input or to sense the presence of a small object. It is a radiometric sensor. The output value depends on the input voltage. It measures the same force value with 3V or 5V.

In Fig. 64, to read the value of the sensor, the phidgets library from Contiki is used. Once the raw value is read, it is transformed it to Newtons, knowing the maximum value of the A/D converter is 4095, and the maximum value the sensor can give is 39.2 Newtons.

4.3.1.4 Relay actuator

The relay used as an example is the Electronic brick 5V Relay form seeedstudio.

This actuator, works as a switch, when a signal is sent through the signal pin. It has a library for the Z1 motes in platform/z1/dev/relay-phidget.h".

This library conflicts with the phidgets library, because it turns the selected pin from the north port as an output, and the phidgets functions as an input. In this configuration, the switch is powered with 5V supplied by the Z1 in the ON port, and with ground in the OFF port. It toggles the led on and off, each time the signal is triggered.

In Fig.68 example, the main loop waits for a specified time, and then toggles the relay.

4.3.1.5 Distance sensor

The distance sensor used as an example is the SEN-12784 from SparkFun. It has an VL6180 digital sensor integrated, that can read light and distance. It uses the an I2C interface to extract the values from the sensor registers.

Contiki has a I2C interface library adapter for the Z1 motes in platform/z1/dev/i2cmater.h To use it in a program, it has to include the library dev/i2cmaster.h

The function in Fig.71, shows how to read the distance from the device. It calculates the distance by sending a pulse of light, and retrieving it back, the doing an internal calculation with the difference between the power of the signal sent and the received. Between the activation and the collection of the value, there is some time waiting for the light to travel. The functions in Fig.71 and Fig.73, show how to set and get a register from the sensor, using the I2C interface.

4.3.2 Operating systems

4.3.3 Communication protocol

Wireless sensor networks combines 3 concepts together: sensor + CPU + radio. However, combining sensors, radio and CPU's together requires an extensive understanding of the hardware components as well as modern networking technologies to connect the devices. Each node needs to have the necessary tools to send data over the radio channel, while meeting the requirements of size, cost and power consumption. The research and development of this kind of devices, has been increased over the last years. There are a number of operating systems focused on providing communications stacks and at the same time focused on saving power. On the other hand, the devices integrating a CPU and a radio transceiver have become more available and efficient.

4.3.3.1 Composition

There are four main types of nodes in a WSN structure. Sensor nodes: These nodes are in charge of collecting data, and sending it to the network. These nodes have 2 parts, the sensors board and the mote. The sensor board, contains the sensor to acquire data (light, temperature, humidity,...) The mote integrates the CPU and the radio transceiver. Route nodes: Nodes with the only purpose of making possible the link between the sensor nodes and the rest of the network. They work as a repeater of the radio signal, and implement routing tasks. Server station: It is the concentrator of the data sent over the network. It is a node itself, or a node attached to a more powerful machine, able to manage lots of data. Gateway: Connects the WSN to an external network, if needed.

The transmission of sensor's data is done by all the nodes of the network. Each data packet, is sent to the server station hop by hop. Reducing the transmission power in the nodes, may reduce the power consumption on it, but it may require a larger number of hops to arrive to the server station

4.3.3.2 RIME

RIME is a communication stack designed for Contiki. It provides a hierarchical set of wireless network protocols. This protocol stack can send data over the standard IEEE 802.14.5 with very few transmissions and less overhead than an IP based protocol, saving energy in the devices involved in the connection. Implementing a complex protocol (say the multi-hop mesh routing) is split into several parts, where the more complex modules make use of the simpler ones.

These are some of the different modules of Rime:

abc: the anonymous broadcast, it just sends a packet via the radio driver, receives all packets from the radio driver and passes them to the upper layer; **broadcast:** the identified broadcast, it adds the sender address to the outgoing packet and passes it to the abc module;

unicast: this module adds a destination address to the passed packets to the broadcast block. On the receiver side, if the packet's destination address doesn't match the node's address, the packet is discarded;

stunicast: the stubborn unicast, when asked to send a packet to a node, it sends it repeatedly with a given time period until asked to stop. This module is usually not used as is, but is used by the next one.

runicast: the reliable unicast, it sends a packet using the stunicast module waiting for an acknowledgement packet. When it is received it stops the continuous transmission of the packet. A maximum retransmission number must be specified, in order to avoid infinite sending.

polite and ipolite: these two modules are almost identical, when a packet has to be sent in a given time frame, the module waits for half of the time, checking if it has received the same packet it is about to send. If it has, the packet is not sent, otherwise it sends the packet. This is useful for flooding techniques to avoid unnecessary retransmissions.

multihop: this module requires a route table function, and when it is about to send a packet it asks the route table for the next hop and sends the packet to it using unicast. When it receives a packet, if the node is the destination then the packet is passed to the upper layer, otherwise it asks again the route table for the next hop and relays the packet to it.

4.4 Proposed

Sentilo is an open source platform to store sensor and actuators information. This platform is designed for the smart cities environment, to be used as a sensor data server that stores the data from different providers and different components within the providers.

4.4.1 Definitions

Provider: A Sentilo account in the server. It stores the published data, and sends the data to his subscribers.

Publisher: A device that sends data to the server. It publish the data into a provider account.

Subscriber: A device that receives data. It is subscribed to a certain data from a provider

Worker: A threat in the server that executes a programed task

Redis: A in-memory data structure store. It is used as a Publisher/Subscriber implementation to store the data in the memory of the server.

MongoDB: A database that stores the data as 'documents'. A 'document' is a JSON object.

4.4.2 Sentilo Architecture

The platform has 3 distinct parts:

PubSub Server (Core)

Web Catalog Application (A web interface to check the information of the PubSub Server)

Extensions (Also called Agents, they extend the capabilities of the PubSub Server)

The core platform, listens and responds to requests specified in the API. By default, it listens the TCP port 8081

- For a publisher, it registers the data sent, in one of the platform items.
- For subscribers, it responds with a JSON with the requested data of an item.

The web catalog, is a web interface to manage and see the information on the PubSub Server. It listens the TCP port 8080. The platform supports some extensions in order to extend the base functionalities such as alerts or data storage.

4.4.2.1 PubSub Server

The Core of the platform is a running process, that listens to the requests and creates workers (Threads) to do the tasks. There are 2 requesters:

Publishers: Send data from sensors, and alerts.

Subscribers: First, they request a subscription. Then waits for the data they are subscribed is sent. The platform is separated in 2 different layers: Transport and Service. The transport layer manages the incoming requests (as published data, data requests or subscription requests) and generates a queue with tasks containing the information of the request. Then, a limited pool of workers handles the requests, every time each finishes the previous task.

When a client sends an Http request to the platform, the process is: (Fig. 76). 1.The server accepts the request. 2.Queues the request on the list of pending requests. 3.When a Worker is available, a pending task is assigned to it for processing(removing it from the queue) (a) delegates the request to an element of the service layer (b) constructs the HTTP response from the information received. 4.Sends the response to the client's request

The service layer manages the workers information and processes it and registers the data or delivers the data depending on the request. (Fig. 77)

1.The Worker delegates the request to the associated handler depending on the type of request (data, order, alarm, ...)

2.The following validations are performed on each request: (a) Integrity of credential: checks the received token sent in the header using the internal database in memory containing all active credentials in the system. (b) Authorization to carry out the request: validate that the requested action can be done according to the permission database.

3.Stores the data in Redis (in memory), and depending on the type of data (a) Publish the data through publish mechanism (b) Register of the subscription in the ListenerMessageContainer (A list of all subscribers) and into Redis as a subscriber.

4. If any new data is received, Redis publish the data to the subscribers, otherwise this step is skipped.

5. The container notifies the event to each subscriber associated with it by sending an HTTP Request to them.

4.4.2.2 Web Catalog Application

The catalog application platform is a web application that uses MongoDB as data storage database. The Web App has 2 parts:

- A public console for displaying public data of components and sensors and their data

- A secured part for resources administration: providers, client apps, sensors, components, alerts, permissions, ...

It is fully integrated with the Publish/Subscribe platform for data synchronization:

- Permission and authentication data

- Register statistical data and the latest data received for showing it in different graphs of the Web application.

4.4.2.3 Extensions (Agents)

The extensions of Sentilo add functionalities to the Core application. The extensions are subscribed to the Redis module for all the incoming notifications.

When Redis receives a publication of data, sends a message to all subscribers, including all the agents. The agent gets the data, and carries out his task. Currently there are 3 Sentilo agents:

- Relational database agent

- Stores all the incoming data in a external database Alarm agent

- Manages the internal alerts defined into the Web Catalog and published an alert if the condition is met.

- Location updater agent

Is responsible of updating automatically the component location according to the location of the published observations.

4.4.3 Sentilo structure

The platform has 5 main items: - Component - Sensor - Alert - Alarm - Order A component is the item where a set of sensors is attached. A sensor is a representation of a physical sensor, it is attached to a component. The data published is sent for a specific sensor. An alert is a trigger registered in Sentilo when an event happens. There a 2 types of alerts: internal and external. The internal alerts are related to specific sensors and it's logic is defined using basic math rules or configuring an inactivity time. The external alerts are defined by third party entities, which will be the responsible of calculating their logic and throw the related alarms when applies. An alarm is the message sent to the subscribers of an alert when it is triggered. Must be attached to an alert. An order is a message registered for a specific sensor or component. It is received by the subscribers of the sensor or component orders.

4.4.4 Sentilo API

The Application Programming Interface (API) define a set of commands, functions and protocols that must be followed by who wants to interact with the platform from external systems, like sensors/actuators or applications. The requests are HTTP requests with 3 fields in the header:

- The Request Method: GET, POST, PUT

- IDENTITYKEY: The authentication token

- Content-Type: application/json

The platform has 3 operations for publishers:

-Retrieve data: Using the GET method, any kind of data can be consulted, the response is in JSON format

-Register data: Using the POST method, can be registered components, sensors alerts, alarms or orders.

-Update data: Using the PUT method, components, sensors alerts, alarms and orders data can be updated. Also sensor data can be published. It also has 3 kind of subscriptions:

- To sensor data - To orders - To alerts

All the documentation of the Application Programming Interface can be found in:

- http://www.sentilo.io/xwiki/bin/view/APIDocs/WebHome

4.5 Experimentation

The objective of this scenario is to connect a Wireless Sensor Network to a running Sentilo server. There are 2 sides of the network, with the border router in the middle of both. The WSN uses CoAP to extract the sensors information, and the sensor data. The Sentilo server uses HTTP requests, with JSON objects. The JSON (JavaScript Object Notation) is a text format transmit data objects consisting of attributevalue pairs. It is one of most widely used by programming languages to send data over HTTP.

4.5.1 Sensor Network

The wireless sensor network is composed by Z1 motes connected by a border-router.

4.5.1.1 Border Router

The Border Router manages the RPL (Routing Protocol for Low-Power and Lossy Networks), and is connected to a computer using Tunslip, a tool used to bridge IP traffic between 2 devices, over the serial line. Tunslip creates a virtual network interface (tun) on the host side and uses SLIP (serial line internet protocol) to encapsulate and pass IP traffic to and from the other side of the serial line.

4.5.1.2 Nodes

Each of the motes has a CoAP server running, and has a resource for each sensor attached to the mote. In this environment 2 Sentilo items will be used:

Component: The hardware where a sensor is attached.

Sensor: A physical sensor. It must be attached to a component For the Sentilo server, each component, sensor, and alert must have a unique id. In this setup, each mote is a component in the server, the mote id is used for the unique id in sentilo. For this example, the mote 3 will have the id MOTE03. Each sensor has his unique id too, using the component id and the type of sensor. In this setup the temperature sensor of the mote 3 will have the id MOTE03TMP. Every sensor has a CoAP resource defined in the mote. A location resource is defined to set the mote location

4.5.2 Network connector

In the computer connected with the border-router, there's a Java application that pulls the information in the WSN using CoAP, and communicates with the Sentilo server to register the sensor and send the data. A provider must be registered manually in Sentilo in order to get the authentication token. For every request sent, the authorization token is checked.

4.5.2.1 Application workflow

The Java application that connects the 2 networks, follows 5 steps: 1. Searches for all the Motes of the specified network in the border router, by sending an HTTP GET to the border router. It responds with an XML with the information of all the motes.

Discovers all the sensors in each Mote, by sending a CoAP discover to each mote.

Gets the information of each sensor, by sending a CoAP GET to the resources on the mote.

Registers each sensor in Sentilo, by sending a HTTP POST to the server with the information of the sensor.

Starts collecting data from the sensors, and registers it in Sentilo, by sending a CoAP observe to each Mote resource, and for each observation, sends a HTTP PUT with the data to Sentilo.

4.5.2.2 Sensor registration

Once the application has a list of all the motes and the sensors of each one, sends a GET request every mote for each one of the sensors resources, to get the information of the resource. The sensor resource has defined the information needed to register.

Once the information of the sensor is gathered, it creates a JSON Object to register the sensor into the Sentilo server via the API.

4.5.2.3 Sensor data publish

The application starts an OBSERVE on the mote for each sensor resource. At this point, the application starts to listen for messages from the CoAP resource. The sensor periodic resource sends information of the sensor data periodically. The period of observation is defined in the mote. In every observation, the data is sent to Sentilo in a JSON Object via the API.

The parameters sent in the JSON to the server are:

4.6 Results

4.6.1 Range

4.6.2 Response time

4.6.3 Connection speed

4.6.4 Power consumption

4.7 Discussion

The Contiki OS, collects all the technologies needed for the development of centralized data collectors, for the sensors. This platform combined with Sentilo, creates a real application platform, to be able to deploy in several possible real environments. The main advantages of Contiki, are how easy is to create code, and generate concurrent scenarios inside the same mote, being able to have a web server at the same time a root node of a WSN is running, without complexity. At the same time , the application level library as COAP, with the complete examples of this libraries, makes this system a powerful and versatile tool. A disadvantage of this platform, is the lack of documentation and examples, outside the inner code. There's a lot of time and test to make, for a more complex application. Secondly, the Sentilo platform, is an easy to install, use and program applications with. It has a wide set of options and tools, that need to be understand carefully for a rich application that uses all the functionalities properly. The combinations of both, makes a good, simple and potentially improvable scenario, for centralize data collection.

4.7.1 Future lines of work

There are some future lines of work in this experimental environment:

1. Test the CoAP server in the new release of Contiki. Contiki 3.0 A new release of Contiki was released in September 2015, with some changes and improvements overall, specially with CoAP. The new release supports CoAP 18.

2. A Java connector with a dynamic network. The Java connector finds the motes in a stable WSN, if a node is missing or replaced, it needs a manual interaction to find all the motes gain, by erasing all the network, and start to find all the motes again. Besides, the protocol handling the routes, is IP and the protocol handling the links is RPL. The IP routes in the border router expire every certain time, that means that if a mote is missing, a route is still present for a certain time, even if the RPL is aware of the missing mote. As a possible solution, there are repairing route methods in CoAP that are used to repair the broken links between nodes.

5 | Template

Template

Contents

Abstract

5.1 Introduction

5.1.1 Problem Statement

5.1.2 Background

5.1.3 Purpose (Goal)

5.1.4 Limitations

5.1.5 Method

5.2 Related work

5.3 Background

Selection of technology

5.3.1 Hardware

5.3.2 Operating system

5.3.3 Communication protocol

5.3.4 Workspace and tools

5.4 Proposed

5.5 Experimentation

5.6 Results

5.6.1 Range

5.6.2 Response time

5.6.3 Connection speed

5.6.4 Power consumption

5.7 Discussion

6 | Short paper

Short paper

Contents

Abstract

The exponential grow of IoT application in different industries raises many questions in wireless network field. Heterogeneous networks of IoT applications strongly depend on the ability on IoT devices to adapt their data transmission parameters to the application requirement running at each time. One of the most important problem of IoT network is that IoT devices are limited in terms of energy consumption and computation capability. Our work is motivated by the idea of matching each transmission configuration with a reward and cost values to satisfy Iot application requirements. Our goal is to make IoT devices able to select the optimal configuration to transmit their data to the gateway with the QoS required by the application running on the device. Determining the best configuration among 6356 possibilities is difficult due to the lack of tools that could take both applications requirements and external environment in consideration to select the best transmission parameters. To address this problem, we use a genetic algorithm in a fog computing to select the transmission parameters needed by the application. Genetic algorithms are used in selection and ranking problems to model the natural selection of chromosomes in human generation. In our case, each configuration represents a chromosome that need to be selected to match better the QoS criteria. Particularly, we analyze the impact of selecting one configuration in 3 kinds of applications: text, voice and image transmission by modeling a new adaptive configuration selection process. We validate our approach by using both simulation and real environment testbed.

6.1 Introduction

The need of Low power wide area network (LPWAN) networks increased quickly these last years. The main factor is that IoT devices require low power consumption to transmit data in a wide area. Lora, Sigfox and NB-IoT are the most known technologies that satisfy these requirements. Applications like smart building and smart environment are one of hundreds use cases that need to be deployed with such technologies. Unlike Sigfox and NB-IoT, Lora is more open for academic research because the specification that governs how the network is managed is relatively open. LoRa is a wireless modulation technique that uses Chirp Spread Spectrum (CSS) in combination with Pulse-Position Modulation (PPM). The transmission could be configured with 4 parameters: Spreading factor (SF), Transmission power (Tx), Coding rate (CR) and Bandwidth (Bw), to achieve better performance. The main LPWAN research directions are about large scale networks to support massive number of devices, interference issues, link optimization and adaptability. Thus heterogeneous network deployments and Spreading Factor (SF) allocation strategies need to be studied. In this paper, we investigate the performance of homogeneous networks (i.e. when all the nodes select the same LoRa configuration) and heterogeneous networks (i.e. when each node selects its LoRa configuration according to its link budget or their needs) for large scale deployments (up to 10000 nodes per gateway). For that purpose we have developed a LoRa Module, based on improved WSNet simulator, including a spectrum usage abstraction, the co-channel rejection due to the quasi-orthogonality of SFs and the gateway capture effect. Simulation results show the performance comparison in terms of reliability, network capacity and power consumption for homogeneous and heterogeneous deployments as a function of the number of nodes and the traffic intensity. The comparison shows the benefits of the heterogeneous deployment where each node selects its configuration according to its link budget.

The article is organized as follows. Section 6.2 elucidates summary of related works, In section 6.4, we propose our ... to .... Section 6.5 evaluate the performance of our ... in terms of packet delivery ratio, throughput, and power consumption. Section 6.7 concludes the article and gives some ideas for future work.

6.2 Related work

Transmission parameter configuration mechanisms such as ADR scheme need to be executed on both LoRa node and LoRa network server. Taking into account low power consumption, the mechanism running on LoRa node should be as simple as possible and has been detailed in LoRaWAN. However, LoRa network server is responsible for the complex management mechanism, which can be carefully designed to improve network performance. Therefore, the discussed related works focus on server-side mechanism. In addition, the mechanism running on LoRa node is in accordance with the definition of LoRaWAN 1.1 specification [14]. The basic ADR scheme provided by LoRaWAN estimates channel conditions using the maximum value of the received signal-to-noise ratio (SNR) in several recent packets [14]. When the variance of the channel is low, using ADR scheme significantly reduces the interference and increases the system capacity compared with using the static data rate [6],[9],[15]. However, the scheme may also have potential drawbacks. First, SNR measurements are determined by different models of LoRa Gateway. Therefore, the value of SNR is inaccurate as a result of hardware calibration and interfering transmissions. Second, selecting the maximum SNR in the last 20 packets is not an desirable method. Because there may be a long time interval between consecutive packets for some IoT applications. The antiquated SNR information is not able to accurately estimate the channel condition for the next uplink packet. Third, the scheme only considers the link of single node to decide whether to adjust transmission parameters. If massive LoRa nodes are densely distributed near LoRa Gateway, it will cause most of nodes using the fastest data rate. With the number of LoRa nodes using the same data rate increases, the possibility of collisions also increases dramatically. Moreover, a lot of researchers propose various approaches to allocate transmission parameters with different objectives. Most of the approaches utilize SNR or RSSI information to control transmission power and spreading factor. The authors in [9] slightly modify the basic ADR scheme. The maximum operation in the SNR of several recent packets is replaced with the average function.

In [21], EXPLoRa-SF selects spreading factors based on the total number of connected nodes and EXPLoRa-AT equalizes the time-on-air of packets transmitted at the different spreading factors. In addition, the authors in [22] propose a link probing regime to select transmission parameters in order to achieve lower energy consumption. In [23], the authors present a scheme to optimize the packet error rate fairness to avoid near-far problems.

In the last years, the LoRaWAN technology has been the subject of many studies, which analyzed its performance and features with empirical measurements, mathematical analysis and simulative tools. Some seminal papers on LoRaWAN such as [**petajajarvi_coverage_2015b**], [24] test the coverage range and packet loss ratio by means of empirical measurements, but without investigating the impact of the parameters setting on the performance. Other works, such as [22], examine the impact of the modulation parameters on the single communication link between an ED and its GW, without considering more complex network configurations. 2 Note that, from the GW perspective, ACK packets are not distinguishable from any other DL packet and, hence, are subject to the same rules and constraints. To obtain more general results, [25] uses a stochastic geometry model to jointly analyze interference in the time and frequency domains. It is observed that when implementing a packet repetition strategy, i.e., transmitting each message multiple times, the failure probability reduces, but clearly the average throughput decreases because of the introduced redundancy. In [**ferre_collision_2017a**] the author proposes closed-form expressions for collision and packet loss probabilities and, under the assumption of perfect orthogonality between SFs, it is shown that the Poisson distributed process does not accurately model packet collisions in LoRaWAN. Network throughput, latency and collision rate for uplink transmissions are analyzed in [26] that, using queueing theory and considering the Aloha channel access protocol and the regulatory constrains in the use of the different sub-bands, points out the importance of a clever splitting of the traffic in the available sub-bands to improve the network performance. In [**bankov_mathematical_2017a**] the authors present a mathematical model of the network performance, taking into account factors such as the capture effect and a realistic distribution of SFs in the network. However, the model does not include some important network parameters, preventing the study of their effect on the network performance. A step further is made in [27] where the authors develop a model that makes it possible to consider various parameters configurations, such as the number of ACKs sent by the GW, the SF used for the downlink transmissions, and the DC constraints imposed by the regulations. In this work, however, multiple retransmissions have not been considered. The study presented in [28] features a system-level analysis of LoRaWAN, and gives significant insights on bottlenecks and network behavior in presence of downlink traffic. However, besides pointing out some flaws in the design of the LoRaWAN medium access scheme, this work does not propose any way to improve the performance of the technology. In [29], system-level simulations are again employed to assess the performance of confirmed and unconfirmed messages and show the detrimental impact of confirmation traffic on the overall network capacity and throughput. Here, the only proposed solution is the use of multiple gateways, without deeply investigating the specificities of the LoRaWAN standard. In [30] a module for the ns-3 simulator is proposed and used for a similar scope, comparing the single and multi gateway scenarios and the use of unconfirmed and confirmed messages. In this case, the authors correctly implement the GWs multiple reception paths, but do not take into account their association to a specific UL frequency, which usually occurs during network setup: indeed, the number of packets that can be received simultaneously on a given frequency can not be greater than the number of reception paths that are listening on that frequency. Also in this case, the study only focuses on the performance analysis, without proposing any improvement. The authors in [31], [32] target the original ADR algorithm proposed by [33], suggesting possible ameliorations. Generally, the modified algorithms yield an increase of network scalability, fairness among nodes, packet delivery ratio and robustness to variable channel conditions. In [23], the authors compute the optimal SFs distribution to minimize the collision 4 probability and propose a scheme to improve the fairness for nodes far from the station by optimally assigning SFs and transmit power values to the network nodes, in order to reduce the packet error rate. In [34] it is shown how the use of a persistent-Carrier Sense Multiple Access (p-CSMA) MAC protocol when transmitting UL messages can improve the packet reception ratio. However, attention must be payed to the fact that having many EDs that defer their transmission because of a low value of p may lead to channel under-utilization. In [35], the authors investigate, via simulation, the impact of DC restrictions in LPWAN scenarios, showing that rate adaptation capabilities are indeed pivotal to maintain reasonable level of performance when the coverage range and the cell load increase. However, the effect of other parameters setting on the network performance is not considered. In this study we differ from the existing literature in that we target large networks with bidirectional traffic, a scenario that makes it possible to observe some unforeseen effects rising from the interaction of multiple nodes served by one single GW and NS. Furthermore, in our analysis we examine one by one the role played by the configurable network parameters, as detailed in Sec.IV-A, thus highlighting some pitfalls that can affect the network performance. We then propose possible counteractions that require some small changes at the MAC layer, and we evaluate their effectiveness in some representative scenarios. As a side result, we enriched the ns-3 lorawan module with new functionalities.

6.3 Background

6.4 Methodology

A generic scheme to solve the configuration selection problem and any other similar selection problem is given in Figure 1. The genetic selection scheme consists of three main steps, the first step contains a set of small parallel fuzzy logic (FL)-based subsystems, the second step is a multiplecriteria decision making (MCDM) system, and the third step is a genetic algorithm (GA)-based component to assign a suitable weight for the criteria in the second component. The scheme decision phase can be described in more detail as follows.

(i) The heterogeneous wireless environment contains up to n networks (RAT 1 ,RAT 2 ,...,RAT n) and the framework has to select the most promising one or to rank the RATs according to their suitability.

(ii) The selection depends on multiple criteria up to i (c 1 ,c2,...,c i). Different type of criteria can be measured from different sources to cover the different view points of the users, the operators, the applications, and the network conditions. Each criterion is measured then passed to its FL-based control subsystem in the first component.

(iii) Every FL-based subsystem gives an initial score for each RAT that reflects the suitability of that RAT according the FL subsystem criterion. The different sets of scores (d 1 ,d2,...,d i) are sent to the MCDM in the second component.

(iv) The GA component assigns a suitable weight (w 1 ,w2,...,w i) for each initial decision according to the objective function that is specified by the operator according to the importance and sensitivities of ANS criteria to the different characteristics of a wireless heterogeneous environment.

(v) Using the initial scores coming from the first component and the weights that are assigned manually or using the third component, the MCDM will select the most promising AN or will rank the available RATs according to their suitability.

6.5 Experimentation

6.5.1 Range

6.5.2 Response time

6.5.3 Connection speed

6.5.4 Power consumption

6.6 Results

6.6.1 Range

6.6.2 Response time

6.6.3 Connection speed

6.6.4 Power consumption

6.7 Discussion

The main challenge of this work is to investigate the possibility of using genetic algorithm to model the selection of lora configuration that satisfy the applications requirements. Our main contribution was to develope 3 applications that requires different level of QoS. such as text transmition, souns transmission and image transmmission. We used a low cost lora gateway on raspberrypi and builded 2 arduino boards equiped with 2 lora Tranceivers. The gateway capture effect is based on the SX1276 transceiver. To measue the accuracy of applying genetic algorithm in the edge computing to select the best lora configuration we used both simulation and real enviroment tesbeds. Our simulations compare the performance of each configuration selection

homogeneous and heterogeneous deployments as a function of the number of nodes and traffic intensity. First, we have analyzed homogeneous deployments for different SFs from SF6 to SF12. Simulations show better performance for the SF6 deployment but reduced cell coverage. Second, we have compared heterogeneous and homogeneous deployments: the Heterogeneous deployment that selects its LoRa configuration according to its link budget results in the best PDR and throughput, as well as the lowest average power consumption compared to other deployments, for a different number of nodes and different traffic intensity. The results clearly show the benefits of heterogeneity for large scale network deployments and the need for adaptive SF allocation strategies. As a future work, we plan to validate our appproach by using both simulation and real enviroment testbeds. We plane to study the performance of applying our approach in terms of PDR, throughput, and power consumption. The module will optimally select the configuration according to the scenario criteria (e.g., high data rate, energy efficiency, or network congestion) and the radio environment (e.g., link budget, level of interference, device mobility).

7 | Long paper

Long paper

Contents

Abstract

7.1 Introduction

Over the past few years, new approaches called Low Power Wide Area (LPWA) networking technologies have emerged.

These technologies became a new alternative to current generations of cellular networks (2G, 3G, and 4G), while covering large areas (typical range of 10 km).

Several papers have reviewed the different LPWA technologies. For example, Raza et al. [1] surveyed standardization activities (IEEE, IETF, 3GPP, ETSI), as well as industrial ones around LPWA technologies (LoRa alliance, Weightless-SIG, Dash7 alliance). They identified potential research directions to address limitations and challenges of LPWA technologies such as a massive number of devices, link optimization, and adaptability.

After giving an overview of LPWA and cellular technologies for IoT [2], the authors discussed the capabilities and limitations of LoRaWAN. Nevertheless, the potential of an adaptive LoRa solution in terms of spreading factor, bandwidth, transmission power, and topology is still not well studied or exploited. Thus, new protocols and strategies are required to improve LoRa scalability.

The first step is to understand heterogeneous network deployments when devices use different spreading factor. In this paper, we investigate homogeneous (i.e. when all the nodes select the same LoRa configuration) and heterogeneous deployments (i.e. when each node selects its LoRa configuration according to its link budget or their needs) for a large number of devices (up to 10000 nodes per gateway).In order to evaluate performance, we have developed an accurate model of the PHY/MAC LoRa based on the extended WSNet simulator.

The LoRa model takes into account spectrum usage, co-channel rejection due to quasi-orthogonality of the LoRa spreadspectrum modulation, and the gateway capture effect. The simulation results give an insight on reliability, network capacity, and energy consumption for homogeneous and heterogeneous deployments as a function of the number of nodes and traffic intensity.

The article is organized as follows. Section II gives a short overview of the LoRa technology and provides the state of the art on LoRa experimental measurements and simulations, LoRa limitations, and network deployment strategies as well as motivations. Section III presents the developed LoRa WSNet based simulator and the deployment scenarios. Section IV analyzes performance in terms of packet delivery ratio, throughput, and power consumption. Section V concludes the article and gives some ideas for future work.

7.2 Related work

7.2.1 LoRa Overview

In this section, we give a short overview of the LoRa physical layer parameters. LoRa specification [3], [4] and technical documents [5], [6], [7], [8] contain more detailed information.

7.2.1.1 Bandwidth (BW):

is the range of frequencies available for transmission. Larger BW increases the data rate but decreases sensitivity. Typically, BWs are 125 kHz, 250 kHz, and 500 kHz. However, Semtech SX1276 offers BW configurations from 7.8 kHz to 500 kHz.

7.2.1.2 Carrier Frequency (CF):

is the central frequency in a band. For our study, we consider the 868 MHz band.

7.2.1.3 Coding Rate (CR):

characterizes resilience to transmission errors. Higher CRs result in better robustness, but increases the air time. LoRa supports CR of 4/5, 4/6, 4/7, and 4/8.

7.2.1.4 Spreading Factor (SF):

is the ratio between the chip rate and the symbol rate. For a given SF, there are 2 SF chips per symbol. SF can be selected between 6 and 12, where SF12 presents the highest sensitivity and the longest range, but the lowest data rate.

7.2.2 From Experimental Measurements to Simulations

Some authors deployed LoRa networks and experimentally studied its performance [9] [10] [11] [12] [13]. The measurements were done in city centers, tactical troop tracking, and sailing monitoring systems. Nevertheless, experimental results in real life networks are not reproducible and MAC layer optimization is difficult. Blenn et al. [14] performed simulations based on traces from experiments and analyzed results based on real life and large scale measurements from The Things Network but their simulations are limited to the deployed scenario. To and Duda [15] presented LoRa simulations in NS-3 validated in testbed experiments. They considered the capture effect and showed the reduction of the packet drop rate due to collisions with a CSMA approach. In a system level simulator, Haxhibeqiri et al. [16] studied the scalability for LoRaWAN deployments in terms of the number of nodes per gateway. Simulations are performed for a duty cycle of 1% but they are limited to 1000 nodes. We developed a LoRa simulator to compare the performance in different deployment scenarios for large scale networks based on an accurate model of the LoRa PHY/MAC layers. We simulate several deployment scenarios varying traffic intensity and the number of nodes.

7.2.3 LoRa Evaluation and Limits

Several authors evaluated performance and limits of LoRa networks. Reynders et al. [17] evaluated Chirp Spread Spectrum (CSS) and ultra-narrow-band networks. They proposed a heuristic equation that gives Bit Error Rate (BER) for a CSS modulation as a function of SF and Signal to Noise Ratio (SNR). Cattani et al. [18] evaluated the impact of the LoRa physical layer settings on the data rate and energy efficiency. They evaluated the impact of environmental factors such as temperature on the LoRa network performance and showed that high temperatures degrade the Packet Delivery Ratio (PDR) and Received Signal Strength (RSS). Goursaud et al. [19] studied the performance of the CSS modulation. They showed the possibility of interference between different SFs and evaluated co-channel rejection for all combinations of SFs. Feltrin et al. [20] discussed the role of LoRaWAN for IoT and showed its application to many use cases. They considered the effect of non perfect orthogonality of SFs for a link level analysis. Petajajarvi et al. [21] analyzed the scalability of a LoRaWAN wide area network and showed its good coverage (e.g. until 30km on water for SF12 and transmit power of 14 dBm). They also showed the maximum throughput for different duty cycles per node per channel. Mikhaylov et al. [22] discussed LoRa performance under European frequency regulations. They studied the performance metrics of a single end device, then the spatial distribution of several end devices. They showed LoRa strengths (large coverage and good scalability for low uplink traffic) and weaknesses (low reliability, delays, and poor performance of downlink traffic). Bor et al. [23] presented a capability and performance analysis of a LoRa transceiver and proposed LoRaBlink protocol for link-level parameter adaptation. Nunez et al. [24] analytically showed the potential gain of adaptive LoRa solutions that choose suitable radio parameters (i.e., spreading factor, bandwidth, and transmission power) to different deployment topologies (i.e., star and mesh). These studies provide a first view of LoRa performance and its limitations. As a conclusion we need to take into account the capture effect and imperfect orthogonality of SFs. We contribute with an accurate LoRa simulation model considering the co-channel SF interference and the gateway capture effect, allowing accurate performance analysis in large scale simulations for different deployment scenarios. Our study extends the previous evaluations of LoRa limits with the evaluation of reliability, network throughput, and power consumption from sparse to massive access deployment scenarios.

7.2.4 LoRa Network Deployment Strategies

Some authors studied LoRa network deployments and SF allocation strategies. Bor et al. [25] studied LoRa transceiver capabilities and the limit supported by LoRa system. They showed that LoRa networks can scale if they use dynamic selection of transmission parameters. Georgiou et al. [26] investigated the effects of interference in a network with a single gateway. They studied two link-outage conditions, one based on SNR and the other one based on co-SF interference. They showed, as expected, that performance decreases when the number of nodes increases and highlighted the interest of studying spatially heterogeneous deployments. Croce et al. [27] showed the effect of the quasi-orthogonality of SFs and found that overlapped packet transmissions with different SFs may suffer from losses. They validated the findings by experiments and proposed SIR thresholds for all combinations of SFs. They remarked that LoRa networks cannot be studied as a superposition of independent networks because of imperfect SF orthogonality. Abeele et al. [28] studied the capacity and scalability of LoRaWAN for thousands of nodes per gateway. They showed the importance of considering the capture effect and interference models. They proposed an error model from BER simulations to determine communication ranges and interference. They also analyzed three strategies of network deployments (random SF allocation, a fixed one, and according to related PDR), the last one presenting the best performance. Lim et al. [29] analyzed the LoRa technology to increase packet success probability and proposed three SF allocation schemes (equal interval based, equal area based, and random based). They found that the equal area scheme results in better performance compared with other schemes because of the reduced influence of SFs. The state of the art indicates the interest in heterogeneous deployments and SF allocation strategies. Thus, we analyze homogeneous and heterogeneous deployments with different SF allocations as a function of the number of nodes and traffic intensity in order to show network performance and the benefits of heterogeneity for large scale networks.

7.3 Background

7.4 Approach

We use an accurate and realistic WSNet-based simulator [30] written in C/Modern C++ under CeCILL free license. WSNet is a modular event-driven wireless network simulator that implements the required communication protocol layers and simulates the network behavior with a high level of accuracy. We have extended the simulator in several aspects (e.g., spectrum use, interference, capture effect) to take into account flexibility and specificity of the LoRa PHY/MAC layers.

7.4.1 Spectrum

To support PHY layer heterogeneity and flexibility, we have modified the core of the WSNet simulator by including a spectrum model. This new core is inspired by Baldo et al. [31] to provide a support for modeling the frequency-dependent aspects of communications. Moreover, it exploits the spectrum in terms of spectral resources instead of logical channels. It provides more accurate PHY models (several waveforms with different configurations) and a more accurate interference model for heterogeneous simulations. Thus, we can evaluate different configurations of LoRa network (e.g., SF, BW, channel) for different homogeneous and heterogeneous deployments. Furthermore, it permits the evaluation of inter-technology interference (from a non LoRa technology), not presented in this article for space reasons.

7.4.2 LoRa Modulation

The LoRa modulation is an adaptation of the CSS modulation. Its advantages are low power transmissions and channel robustness. LoRa features seven orthogonal SFs from SF6 to SF12. Symbol duration T s and bit rate R b are defined as follows:

We have developed a LoRa modulation module in- spired by the Equation 3 presented in [17]. This equation shows BER as a function of SF and the energy per bit to noise ratio N

where Q(x) is the Q-function.

7.4.3 Spreading Factors Orthogonality

We have developed an interference module taking into account the effect of the quasi-orthogonality of SFs. It uses the values for co-channel rejection for all combinations of the desired signal SF d and the interferer signal SF i [19] presented in Table I.

Table 7.1

7.4.4 LoRa SX1276 and SX1301 Transceivers

The Semtech SX1276 transceiver provides high interference immunity while minimizing energy consumption [7]. Sensitivity dBm is defined according to the Semtech designer guide [5] as:

where 174 accounts for the thermal noise effect,BW is the receiver bandwidth, N F is the receiver noise factor for a given hardware implementation, and SN R is the minimum ratio of the desired signal power to noise that can be demodulated. Table II shows sensitivity and the data rate for the SX1276 transceiver.

Table 7.2

The Semtech SX1301 offers breakthrough gateway capabilities with a multi-channel high performance transmitter/receiver designed to simultaneously receive several LoRa packets with different SFs and up to 8 channels [8]. It enables robust communications for a large number of nodes spread over a wide range.

In our simulator, we consider the gateway capture effect based on SX1301. The gateway may receive, depending on the reception power, several packets with different SFs because they are quasi-orthogonal with respect to each other. If the gateway receives two packets overlapped with the same SF, the gateway will rec

7.4.5 MAC and Application Layers

Our simulator considers a LoRa random access method that basically behaves like ALOHA. The application layer manages the data traffic generation at each node inside a time window Application Period (AP). At every AP occurrence, each node picks randomly a time instant inside the current window, and a data packet is generated and sent to the lower layer (network or MAC). Hence, each node can generate only one data packet per AP. We vary AP between 60 s and 40 min to evaluate performance for different application use cases.

7.4.6 Deployment Scenarios

In this subsection, we present the network deployment scenarios and the SF allocation strategy in each scenario.

- In SF i Homogeneous, nodes are uniformly deployed in a disk of radius equal to D max (SF i), the maximum transmission range of SF i (Figure 1(a)), and select the same LoRa configuration SF i .
- Multi-Homogeneous is the superposition of independent SF i homogeneous deployments (i from 6 to 12 and the corresponding radius D max (SF i)) with a single gateway at the center. The number of nodes are equally distributed between homogeneous networks. As each SF i homogeneous deployment is independent and uniform, we can see in Figure 1(b) higher density close to the gateway.
- In Heterogeneous f(Dmax), nodes are uniformly deployed in a disk of radius equal to D max (SF 12), the maximum transmission range of SF12 and then, each node selects its configuration according to its link budget (i.e., with the maximum data rate or the minimum SF). As shown in Figure 1(d), some rings naturally appear. Nodes in the farthest ring are configured with SF12, in the next ring with SF11, and so on until the last disk with SF6.
- In Heterogeneous Random, nodes are still uniformly deployed in a disk of radius equal to D max (SF 12) but they randomly select their configuration among the ones available according to their link budget. Hence, each node selects its LoRa configuration according to its budget link and its needs. For example, nodes in the third ring can randomly select their SF between SF10 and SF12 depending on their optimization criteria (e.g., energy consumption, data rate, reliability).

7.5 Experimentation

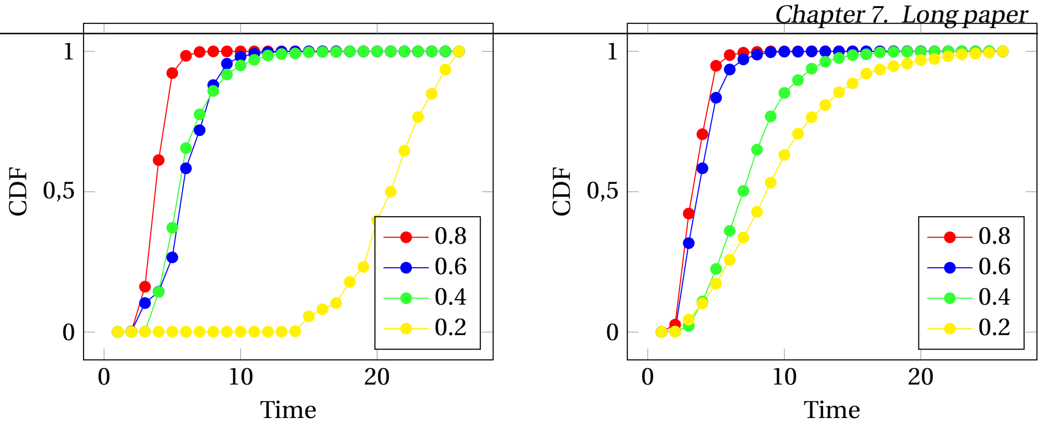
To evaluate our approach, we used Enron dataset in our experimentation. There are many reasons for using Enron dataset to evaluate our vulnerability measure techniques. First of all, it is probably the only actual corporate messaging service dataset available to the public. Second, it contains all kind of emails "personal and official", so email logs can reveal the level of trust between users by studying the information flow in the network. Finally, this dataset is similar to the kind of data collected for fraud detection or counter-terrorism, hence, it is a perfect test bed for testing the effectiveness of new vulnerability measure techniques.

The properties of the Enron dataset used for our experimentation are presented in table 7.3.

Parameter	Enron dataset	Caliopen dataset
Nodes	958	5885
Edges	6966	26547
Diameter	958	2096
Mean degree	2.413361	9.02192
Edge density	0.00252	0.001533
Modularity	0.654600	0.86526
Mean distance	3.042114	3.914097

Table 7.3: Datasets properties.

Due to visibility issues, we extract only important nodes given in [36]. Our substantive interest in this experimentation is in how vulnerability moves through the network. The inputs of our experimentation consist of a set of individual vulnerabilities of users in the network. This set is generated randomly and is represented in the graph of Figure **??**, values of this set are between 0 and 1 and represent the extent to



(a) Enron dataset.

(b) Caliopen dataset.

Figure 1: Cumulative distribution function of infected users.

which users are exposed to different kind of attacks such as (phishing, spam, etc). Unlike social vulnerability values, these values didn't take into account the vulnerability contagion process between users. To evaluate how trust coefficient affects our outputs i.e., social privacy vulnerabilities, we variate the trust coefficient through 4 symmetric values 0.2, 0.4, 0.6 and 0.8.

7.6 Results exploitation

7.6.1 Deployment Scenarios and Assumptions

In this study, we have simulated the deployment scenarios described in Section III.F with a SX1301 gateway, at the center, configured with one single channel of 125 kHz and up to 10000 SX1276 nodes. Hence, the gateway can simultaneously receive up to 7 packets configured with different SF each (SF6 to SF12). Packets are dropped according to the modulation and interference models. Table III presents the simulation parameters. For sake of simplicity, we have considered the 868 MHz band and the Okumura Hata pathloss model.

7.6.2 Homogeneous Deployments Scenarios

In this subsection, we present the results of simulations for homogeneous deployments with different SFs as a function of the number of nodes (up to 10000) with AP set to 60 s (i.e., each node transmits a packet every minute). We have to emphasize that nodes are deployed in a disk of radius equal to the maximum transmission range of SF i for each scenario, i.e. the Homogeneous SF6 in a disk of radius D max (SF 6) and so on. Figure 2 shows PDR for homogeneous deployments with different SFs from 6 to 12. We can see that PDR decreases when the number of nodes increases. The SF6 Homogeneous deployment has better PDR compared with the others due to its short air time but its range is reduced

Figure 3 shows the throughput in packets per second for homogeneous deployments with different SFs. We can see that when the number of nodes increases, the throughput becomes saturated. The SF12 Homogeneous deployment converges faster and presents the lowest throughput due to its low data rate. Nevertheless, it presents the longest range. We have extended the previous analysis by varying AP up to 40 min and we set the number of nodes to 10000. As shown in Figure 4, increasing the Application Period reduces the probability of collision, then PDR increases. Figure 5 shows the throughput as a function of the Application Period. As we can observe, the throughput is higher when nodes use lower SF. Thus, in dense networks with high intensity traffic, we have to prioritize the use of the SF6 configuration. Nevertheless, it reduces the range and therefore, multi-hop communications or denser deployment with several gateways may be necessary to cover the same area. Figure 6 compares the average power consumption per node for the homogeneous deployments as a function of the Application Period for a simulation time offers 100*AP. Increasing traffic intensity (e.g. AP from 40 min to 1 min) results in increased power consumption. SF6 configuration presents low consumption due to its shorter time on air. Higher consumption for the SF12 configuration due to its longer time on air.

7.6.3 Heterogeneous vs. Homogeneous Deployment Scenarios

In this subsection, we compare the simulation results of the heterogeneous strategies with Homogeneous SF12 and Multi-Homogeneous deployments. Each deployment covers the same area (disk of radius D max (SF 12)). For homogeneous deployments, only Homogeneous SF12 is used for comparisons in order to keep the same cell coverage. We simulate the heterogeneous deployments with up to 10000 nodes and AP is fixed to 60 s. Figure 7 compares PDR of homogeneous and heterogeneous deployments. The Heterogeneous f(Dmax) deployment presents the best performance. This is because in heterogeneous deployments we reduce packetloss taking advantage of the quasi-orthogonality of SFs and the deployment strategy. For 100 nodes, the gain in terms of PDR for the Heterogeneous f(Dmax) deployment is 300% compared to the Homogeneous SF12 deployment. For the Multi-Homogeneous and Heterogeneous Random deployments, the gains are respectively 214% and 200%. A reliability of 20% is reached at 170, 360, 400 and 2600 nodes for the Homogeneous SF12, Heterogeneous Random, Multi-Homogeneous and Heterogeneous f(Dmax), respectively. Figure 8 compares the network throughput for the homogeneous and heterogeneous deployments in received packets per second. When the number of nodes increases, the throughput saturates. The Heterogeneous f(Dmax) deployment presents better performance compared with others deployments up to 10000 nodes. Figure 9 compares PDR as a function of traffic intensity (i.e. with AP from 1 min to 40 min). Decreasing traffic intensity reduces the packet loss, then PDR increases. The Heterogeneous f(Dmax) deployment presents better PDR because it takes advantage of the orthogonality of SFs reducing the interference, e.g., all nodes set up to SF12 are far from the gateway, then the interference to nodes close to the gateway set up to SF6 are reduced. The Homogeneous SF12 deployment presents lower PDR due to its long packet duration and the low spectral efficiency. Figure 10 compares the throughput for homogeneous and heterogeneous deployments. When traffic intensity decreases, the throughput decreases. For high traffic intensity (e.g., 1 min Application Period), the Heterogeneous f(Dmax) deployment presents a throughput of 10 packets per second whereas other strategies have a throughput less than 2 packets per second. Figure 11 compares average power consumption for the homogeneous and heterogeneous deployments for a simulation time of 100*AP. When increasing traffic intensity, power consumption increases. Taking the Homogeneous SF12 deployment as reference, the comparison shows that power consumption for the Heterogeneous f(Dmax) deployment increases smoothly compared with the exponential increasing of the Homogeneous SF12 deployment.

7.6.4 Application Period and Packet Duration vs. ERC Regulations

In Europe, ERC regulates access to radio frequency bands. For most of the sub-bands in the 868 MHz band, the duty cycle must be lower than 1%. We analyze this constraint for a 50 Bytes packet configured with different SFs. Table IV shows the packet duration and the Application Period required to respect the regulation for different values of SF. For an AP of 1 min, the packet duration must be lower than 600 ms to respect the duty cycle of 1%. Table IV shows that the constraint can be satisfied with all SFs except SF11 and SF12. APs for SF12 and SF11 must be longer than 2.6 min and 1.4 min, respectively, in order to respect the regulations.

7.7 Conclusion

8 | Template

Template

Contents

Abstract

8.1 Introduction

The difficulty to build such system is

In this work we

The article is organized as follows. Section 8.2 elucidates summary of related works, In section 8.3, we propose our ... to .... Section 8.4 evaluate the performance of our ... in terms of packet delivery ratio, throughput, and power consumption. Section **??** concludes the article and gives some ideas for future work.

8.2 Related work

8.3 Approach

8.4 Experimentation

8.5 Results exploitation

8.6 Conclusion

9 | UTLC

UTLC

Contents

Abstract

Most traffic light's control systems in smart cities are wired and have a semi-static behavior. They are time-based, with pre-configured pattern and expensive cameras. Although traffic lights can communicate wirelessly with incoming vehicles, they are less adapted to an urban environment. If we consider light signs as an Internet of Things (IoT) network, one issue is to model thoroughly the change of signs' states and the Quality of Service (QoS) of this network. In this paper, we propose a new architecture of Urban Traffic Light Control based on an IoT network (IoT-UTLC). The objective is to interconnect both roads' infrastructures and traffic lights through an IoT platform. We designed our IoT-UTLC by selecting motes and protocols of wireless sensor network (WSN). Message Queuing Telemetry Transport (MQTT) protocol has been integrated to manage QoS. It enables lights to adapt remotely to any situation and smoothly interrupt traffic light's classic cycles. Our experimental results show that the MQTT protocol is efficient when the packets rate exceeds 35% of traffic flow, it reduces traffic delay up to 0.05s at 90% of congestion. After verification and validation of our solution using a UPPAAL model checker, our system has been prototyped. Motes' functions have been implemented on Contiki OS and connected through a 6LoWPAN/IEEE 802.15.4 network. Time-stamping messages have been performed throughout the system to evaluate the MQTT protocol with different QoS levels. In our experiments, we measured the Round-trip delay time (RTT) of messages exchanged between the WSN and IoT Cloud. The results show that MQTT decreases the RTT when the Cumulative Distributed Function (CDF) of generated messages exceeds 35%.

9.1 Introduction

According to the French agency of statistics on roads' accidents, 12% of them happen in intersections caused by the non-compliance to traffic rules. 23% of them led to hospitalization in which 14% are fatal [**Routiere2015**]. Urban Traffic Light Control (UTLC) are one possible solution to regulate vehicle flows at intersections. However, a static cycle of traffic lights (or lights signs) has a direct impact on traffic jams, particularly when an emergency vehicle must cross through as quickly as possible. A long period at red or green light might impact the fluidity of the city traffic.

The Internet of Things (IoT) and Everything (IoE) can be a solution to adapt traffic light control to traffic density. It allows heterogeneous connected objects, *e.g.* Zigbee, LoRa, SigFox, ITS-G5, to interact and exchange sensed data on roads, vehicles, pedestrians presence, time of leaving house, etc [**Perera2014**]. Therefore, connecting heterogeneous infrastructures following a Device-to-Device approach is possible through upper layers or an intermediate Cloud platform. Wireless Sensor Networks (WSNs) are the source of these data, and the Cloud is the remote entity that collects them. Fog and Edge computing have been proposed to address the low latency of IoT applications by efficient resource distribution and a local processing. Fog computing leverages Edge devices and remote and private Cloud resources with distributed data processing. It provides the advantage to process data closer to the source and thus mitigate latency issues and reduce network congestion. However, constructing a real IoT Fog Computing is costly for evaluation while the environment has to be controllable for experiments [**Dastjerdi2016**]. However, in our work we would implement our Edge computing with a remote Cloud data gathering and deal with latency through QoS protocol. Remote Cloud offers the possibility to integrate new services. For example, we can deploy sensors to measure noise or air pollution via traffic signs or roads.

Our objective is to model, prototype and evaluate the Quality of Service (QoS) of an IoT solution for a traffic light control system. Modeling of traffic light states control is essential to avoid incoherent situations. Number of models based on Petri Nets (PN) have been proposed in [**difebbraro_trafficresponsive_2006**] and [37]. Their main drawbacks are the limitation to the structural analysis of state transitions and the lack of verification. However, our design is based on UPPAAL (UPPsala and AALborg Universities) [38] timed automata for design and verification of coherent state of cross road's traffic light. It specifies a graph of states with clocks and data variables. To implement our Urban Traffic Light Control based on an IoT network architecture (IoT-UTLC), we setup a real IEEE 802.15.4 WSN with devices that can act as actuators and sensors. Small traffic light signs are driven by a Border Router (BR) device to a sink node which is a gateway to the Internet. This BR is connected to a host computer (or sink) also connected to an IoT Cloud platform. WSN devices forward their data to the IoT Cloud through this sink which defines required levels of QoS based on Message Queuing Telemetry Transport (MQTT) protocol [**Al-Fuqaha2015**]. The collected data can be transmitted to different devices such as sink, BR or wireless sensor/actuator devices. When WSN devices detect the arrival of high priority vehicles, sensed data are routed to the IoT Cloud. Then, the sink node takes a decision to change the light's state and forward generated messages to actuator devices through BR with a high level of QoS. Thanks to IPv6 over Low power Wireless Personal Area Network (6LoWPAN) [**chalappuram_development_2016**], our WSN is energy-efficient and IPv6 compatible.

This paper is organized as follows. In Section 9.2, we review related work. Section **??** reports the design of our prototyping solution. We describe the use case defined with the design model. Section **??** defines our prototype (IoT Testbed), giving our specifications and discussing on our choices of technologies and protocols. Finally, Section 9.6 presents the obtained results that show the usefulness and the best practice for MQTT.

9.2 Related work

Petri nets (PNs) are widely used for traffic light modelling and control [37]. In [**difebbraro_trafficresponsive_2006**], deterministic-timed Petri Nets have been used to describe signalized intersections. Undesirable deadlock states might appear when the nets are tested for some use cases. The authors in [39] have modified PNs models including stochastic-time for one single signalized intersection. Dotoli and Fanti [40] have built a colored timed PN with a deterministic modular framework, in which parts of the system, and even parts of the subsystems, can be specified and analyzed separately. Examples using modularity are given in Soares and Vrancken [41], in which a p-timed PN is used for the control of a traffic signal in both main road and side streets. However, formal characteristics of PNs (*e.g.*, deadlock and liveliness) havent been discussed. Moreover, PNs suffer from a lack of analysis and verification tools. To overcome these limits of PNs, we propose UPPAAL timed automata for design and verification of coherent state of cross road's traffic light. UPPAAL is a timed-based modelling software with a graphical user interface. It is the result of the research works of two universities UPPsala University in Sweden (UPP) and AALborg University in Denmark (AAL) [38].

In [**Web0**], thermal cameras and on-street wired sensors detect vehicles and pedestrians in order to adapt the cycle of traffic light control systems. However, such a solution can be expensive. In addition, the system uses only its local view of the environment to detect the arrival of a vehicle. Other solutions use recent technologies such as wireless sensors devices to limit the cost of thermal cameras and reduce the time needed to deploy sensors. In [**tlig_decentralized_2014**] and [**rose_internet_2015**], the authors propose an adaptive system based on local wireless communication between lights and vehicles. But such a solution requires a global interconnection between all road's users and infrastructure. This problem comes from the rigid definition of technologies' standards. Our work is not only limited to establish WSN, but it is scalable to interconnect heterogeneous wireless technologies through the Internet. The obtained WSN intends to meet multiple QoS requirements of IoT applying the MQTT protocol. In [**Silva2018**], the latency of MQTT has been evaluated by calculating the average round-trip delay between two clients located in two different continents. However, the evaluation has been limited to the impact of messages' size. In our work, we consider the period of generated messages, and we calculate the RTT delays from WSN to Cloud IoT plateform.

In [37] [**difebbraro_trafficresponsive_2006**] [39] [41], the authors focus only on the structural analysis of their models and the transitions between colors of traffic lights. However, the implementation of their models as a service in the Internet of smart cities has not been discussed. Moreover, their methodology is not tested with any real traffic lights' Testbed.

9.3 Related work

9.4 Use Case and Model Design

Our objective is to build a robust Testbed that behaves like a real urban traffic light control system. As presented in section 9.2, a crossroad traffic light design has been proposed based on PNs [37]. Authors demonstrated the necessity to monitor checkpoints like traffic light transitions from red to green and define critical control points to ensure that the transition model is correct. Authors have identified which signal indication sequence optimizes the overall system performance. We were inspired from this work by adding the vulnerability of wireless network *i.e.* packet loss. Indeed, this model is theoretical and static, and would not model entirely traffic light control system. Thus, we propose a UPPAAL timed automata for modelling and verification of the system.

9.4.1 Use case

We consider the use case of traffic lights at an intersection crossed by high priority vehicle like ambulances, fire-fighters or public transportation systems. Crossing delays are important in such use cases, when the goal is to travel in the city from two locations without experiencing traffic jams. We consider the case of a high priority vehicle approaching a traffic light sign on red state. The detection of priority vehicles via (RFID or touch sensor) triggers the transmission of notification to road signs' network asking for a switch of traffic light to green. In that situation, it should be possible to interrupt the usual cycle of a crossroad.

In order to be as close as possible to a real urban traffic light, we prototyped the closest Testbed of a crossroad in Paris. We took an actual crossing point with its dimension and static timers. Fig. 1 shows IoT interconnection of four traffic lights, high priority vehicles and roads. Our system is described by one intersection (or crossing) of two roads A and B, with two traffic lights by road. The signs and roads can use heterogeneous technologies as presented with different colors.

The roads are connected to the Internet through sensors, which are able to detect the arrival of high priority vehicles. The detection is performed by Touch sensors driven by WSN nodes. Note that signals on each road should always have different colors (or states). Of course, when the road A traffic light is on green, the state of the traffic light on the opposite road, must be on red and vice versa. To access the Internet, all messages sent by those objects will go through a Border Router (BR) device and a Middleware or Edge computer. The Middleware is connected to the Internet and forwards all packets to a Cloud. Therefore, the collected data in the Cloud allow a Middleware to decide for the future state of lights. Then, BR disseminates this decision through the network to the actuators.

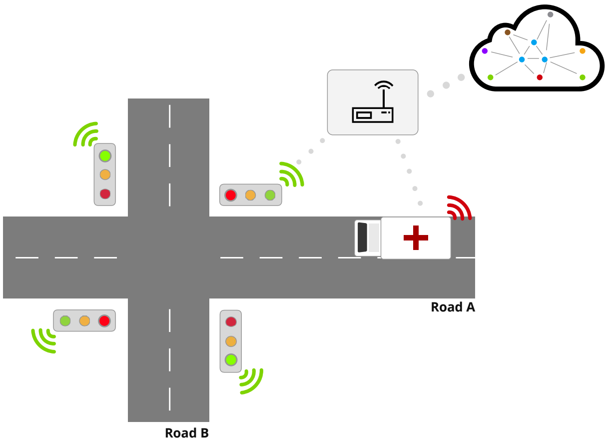


Figure 1: Use case illustration

9.4.2 Design Model

Our design is based on UPPAAL model checker software. It specifies a graph of states with clocks and data variables. It allows us to model how our system works and simulates all possible traffic lights states. The modelling allows us to cover all possible cases of lights change of our IoT-UTLC. It is also a tool for verifying formally the consistancy of traffic light changes: GREEN to YELLOW states, YELLOW to RED states and RED to GREEN states. We simulated our system by three automata shown in Fig. 3, Fig. 2 and Fig. 4 and available at (https://github.com/IoT-UTLC/Resources). We proved that our model worked without deadlock and starvation. It means that in our system, there is always a transition to go to the next state. It proves that the system will not stop functioning over time. Incoherent situations, like four signals on GREEN, must not happen.

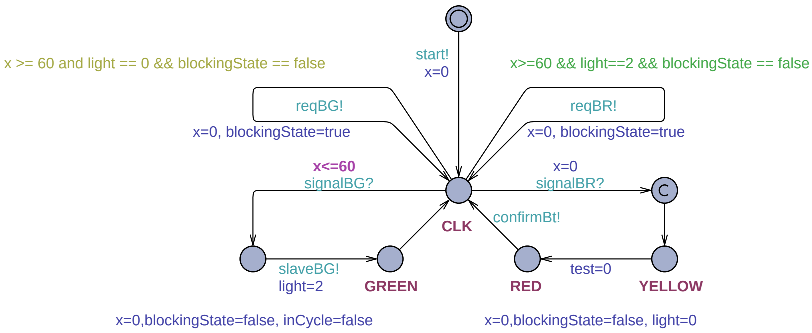


Figure 2: Model of our Traffic Lights in UPPAAL

Fig. 2 highlights the model of traffic lights and describes their behavior. The change of light's state is based on requests and confirmation exchange between lights and the Middleware. We defined two roles for the traffic lights: one is set to master mode and the second one is set to slave mode. Masters send requests every 60 seconds to change their states while considering the current ones. Line 6 of Algorithm 1 shows the condition when a light should change its state. CLK state represents clock or time progress. *reqAG* and *reqAR* are respectively the requests to ask for GREEN and RED states on road A. The same is specified for the road B. When it receives a confirmation from the Middleware, a cycle is started to send the signal with the desired state. Every 30 seconds, the traffic lights can change their states, starting with GREEN light and then to YELLOW light for 3 seconds after that switching to RED. They can also start from RED and then change to GREEN state after additional 3 seconds. We add this extra delay to avoid a dangerous situation when a GREEN state is on the two roads A and B at the same time. Even if we have messages lost due to the wireless nature of the network, our UPPAAL models ensure that this situation doesn't arise. It has been introduced after experiencing a latency between the WSN and the Cloud platform (see section 9.6). When GREEN or RED states are actuated, confirmation messages are sent to the Middleware. The pseudo Algorithm 1 summarizes this mechanism.

The model in Fig. 3 defines the different possibilities in terms of internal cycles depending on the request made by traffic lights (*reqAG*, *reqBG*, *reqAR reqBR*). Mainly, the Middleware sends the message to the

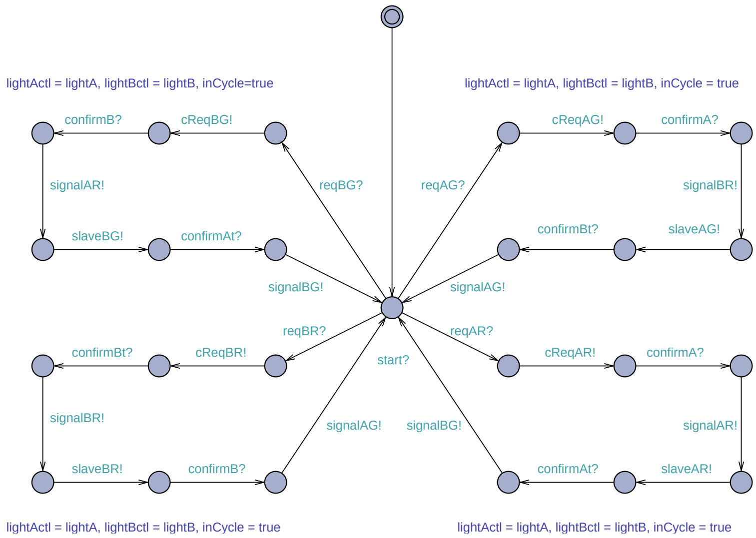


Figure 3: Model of our Middleware in UPPAAL

Algorithm 1: Traffic light

init_60s_timer(); **while** *true* **do**
  if *end_timer()* **then**
    send_request_new_state();
    reset_timer();
  end
  if *msg_received_red() and my_state is green* **then**
    change_state(yellow);
    wait();
    change_state(red);
    send_confirmation_to_middleware();
  end
  if *msg_received_green() and my_state is red* **then**
    change_state(green);
    send_confirmation_to_middleware();
  end
end

Cloud and waits for its response. Then, it sends messages to traffic lights master and slave to change their state following this order: every light go to RED before setting GREEN signals. It also uses acknowledgments from the traffic lights to ensure that the new state has been set. In order to ensure these two features, we used a system to retain messages if the IoT Cloud Platform send GREEN states before RED states (see line 3 of Algorithm. 2). Algorithm 2 shows a simple description showing how the Middleware confirms the order of traffic lights changes.

We introduced the IoT Cloud Platform model shown in Fig. 4. It simulates the subscription mechanism according to messages sent by the Middleware to update collected data. We defined two states RED and GREEN without a transition state like YELLOW state defined for the traffic lights. The condition *touchA* indicates if the road A detects the high priority vehicle. The name *touch* is related to the type of sensor integrated in our prototype (see next section). The Cloud confirms to the Middleware that the state is changed by sending a message *confirmA*.

Exchanged messages within our WSN are based on IEEE 802.15.4 stack. And our Middleware defines QoS levels of exchanged messages via MQTT protocol.

Algorithm 2: Middleware confirmation

```
if message_received() then
    if is_green() then
        retain_msg(); //green then red
    end
    else
        if retained_msg_exist() then
            update_to_red(); //green then red
        end
        else
            update_to_red(); //red then green
        end
    end
end
while true do
    confirmation_red_lights();
    update_to_green();
    confirmation_green_lights();
end
```

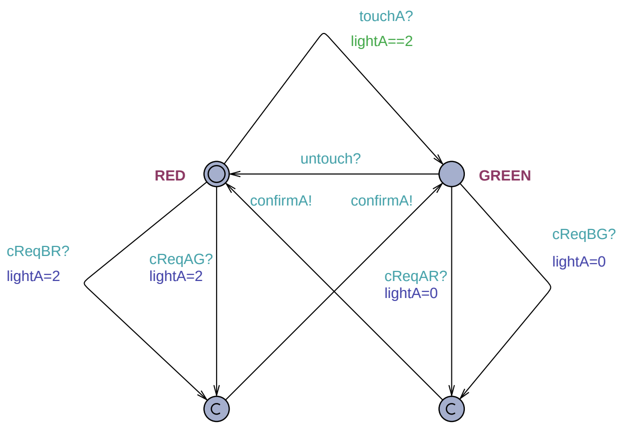


Figure 4: Model of our Cloud variables in UPPAAL

9.5 Prototyping

We have prototyped the wireless sensors and actuator's network of traffic lights and roads on a mockup [1] of real intersection in Paris with a scale of 1:68. Our specifications have been defined considering the low-cost and energy efficiency of the solution. This Testbed is a proof of concept of not limited to our use case as it is scalable for other applications. For example, additional sensors of fine particules could be implanted bringing correlation between traffic jam and pollution.

Fig. 5 shows the architecture of our IoT-UTLC with three layers. From left to right, we have the WSN layer with connected traffic lights actuators, sensors and IEEE 802.15.4 transceivers. The second part is the gateway of the WSN ensured by the BR and the Middleware *i.e.* Python script launched by host computer. The last layer is the Ubidots IoT Cloud Platform. It is an open source solution used to collect and analyze WSN data.

9.5.1 6LoWPAN, Contiki OS, Re-Mote and Border Router

Our WSN is an IPv6 LowPower Wireless Personal Area Network (6LoWPAN) based on IEEE 802.15.4 stack. It is well adapted to embedded wireless devices with energy aware constraint and for its capabilities to define a mesh topology. Contiki Os[2] has been used to implement networks' functions such as send, receive and data processing. It is an embedded operating system with large open source community. It supports Zolertia's

[1] https://github.com/IoT-UTLC/Resources/wiki
[2] http://www.contiki-os.org/

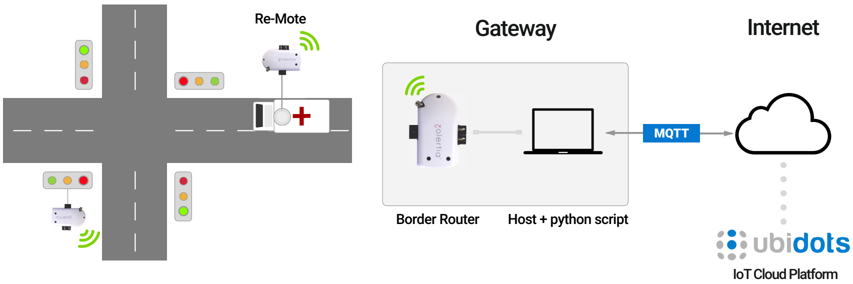


Figure 5: Architecture of Iot-UTLC

Re-Motes [3] and implements recent IEEE 802.15.4 standard specifications. It also includes protocols such as RPL, CoAP and MQTT. Furthermore, developer community is active and makes available source codes examples in order to help developing quickly new applications.

Re-motes are compatible with our WSN specifications and our design model. They are wireless devices with ultra-low power operation mode. This choice has been motivated by long radio range of its IEEE 802.15.4 CC1200 transceiver, which transmits in the frequency band of 868-915 MHz. In addition, each Re-Mote has analog and digital ports with a possibility to connect several analog sensors and actuators. A Re-mote can be driven by a computer and become a sink and/or BR as well as a gateway between the 6LoWPAN network and the computer.

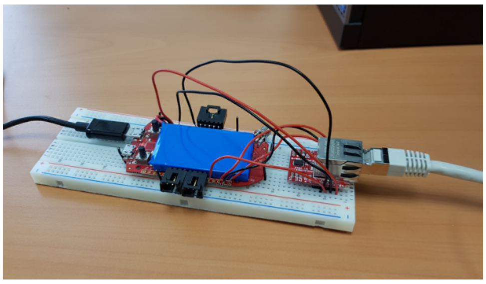


Figure 6: BR and sink combined on one board

To implement the previous model described in Section ??, we used six Re-motes: one for the BR, four to control the traffic lights and one Re-Mote to detect the arrival of a high priority vehicle near a crossing point. For simplicity, we choose a touch sensor as a detecting device of priority vehicle. We have developed four types of programs running on a Re-mote: traffic lights signs, sensors, high priority vehicle detecting device and BR function. Sensors send periodically information to the IoT Cloud Platform with temperature, pressure or any relevant information that can be sensed. As mentioned in the previous section, traffic lights are sub-divided into two modes: slaves and masters. Masters nodes are the only ones to request the Middleware to change its lights state and slaves simply change its state depending on received packets. These roles are defined to reduce the overhead of network, redundancy and collisions, for instance. Masters send periodically packets to request a change of state to the Middleware which forwards them to a Cloud platform.

BR node behaves differently compared to the other Re-motes. The entries of its routing table are the list of Re-motes that pass through it. It reroutes every packet it receives from its neighboring to host computer (or sink), which creates a connection to the IoT Cloud platform. Two options are possible to create our BR: i) separate the BR and sink and ii) combine both on the same device. In our development, we worked on how to implement the sink and the BR nodes on the same Re-Mote board. Fig. 6 shows a prototype of the combined BR and sink, both connected to an ethernet interface. Indeed, if the border router becomes an Ethernet router, there will no longer be any connection between the host/sink machine and the IoT Cloud platform. Every Re-mote is able to connect independently to the IoT Cloud platform. This approach has some advantages, such as the autonomy of the devices, but it generates an overhead requiring extra synchronization packets' exchange. Therefore, we separate the sink and BR, since this solution is more flexible and resilient for our Testbed.

[3]https://github.com/Zolertia/Resources/wiki/RE-Mote

9.5.2 MQTT and UBIDOTS

Fig. 7 presents the layers of our UTLC network. From bottom to up, the WSN network sense and/or detect, process and actuates traffic lights. The second layer manages the 6 LowPan addressing and routing of packets throughout an IEEE 802.15.4 network. The Edge Computing is the Middleware between the WSN and the Cloud platform. For the setup of our UTLC, we start by establishing the access network of WSN. The next step is to connect this network to Core network. MQTT protocol controls three levels of QoS of exchanged packets from the WSN to the chosen Ubidots [4] Cloud platform. It adopts IntServ approach for supporting quality of service in the network, it tags incoming packets in the border routers with different levels of priority. Core routers read incoming packets headers and queue them according to their priority, packets with a high priority are sent faster compared to low priority ones.

MQTT ensures the QoS and publish/subscribe mechanisms through a broker. The broker behaves as a server by filtering messages and organizing them in topics, which are strings used to filter messages and define the hierarchy of our data structure. They allow us to organize how to receive multiple data from sensors such as temperature, up time, battery status and how to display them and obtain a real-time glance of our system. It gets its messages from publishers and sends any modifications to entities, which that subscribed to the updated topics. We used this mechanism with the Middleware in order to publish messages to the broker and get from the main topic the new values of the subscriber.

The QoS feature of MQTT protocol manages network resources by handling retransmissions and guarantees the delivery of messages. It allows more control on messages by defining the level of guarantee. By default, the QoS is defined by three levels. The first one, level 0, is 'At most one'. Level 1 is 'At least one' where there is an acknowledgment to let the sender know that its packet has been received. Finally, level 2 'Exactly once' is the highest verification level with a request/response flows to ensure that only one message will be delivered and processed by the receiver. In our case, we applied levels 1 and 2 using *paho.mqtt.client* Python library. For example, publisher of high priority data such as touch sensor has to indicate the highest level of QoS by the code shown bellow. We shared our implementation and its source codes at https://github.com/IoT-UTLC/contiki.

```
payload = json.dumps({"RoadA": data, "RoadB": 0})
res, mid = conn.publish(MQTT_URL_PUB, payload,
            qos=int(QoS)) # QoS is QoS level to use
```

We experienced significant latency of high priority messages when we tested of IoT-UTLC mockup. Therefore, assessments of the MQTT protocol in our case provided significant information about its efficiency.

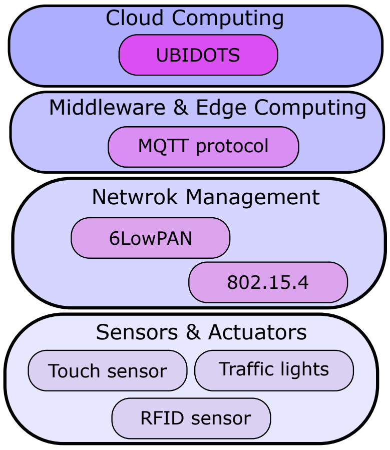


Figure 7: UTLC network layers

9.6 Results

In this section, we report experimental results of urban traffic-light control system based on MQTT protocol. We report the limitations of our solution. To test the efficiency of MQTT protocol, we made 2 scenarios, in the first scenario, the frequency packets sending is 1 packet per 10s, the second scenario, the frequency

[4]https://ubidots.com/

packets sending is 1 packet per 1s. The measured delays have been taken into account between the Middleware (Edge) to the Cloud platform (Ubidots) throughout the Internet. Note that we don't know the routes and the routers that our packets will go through. We measured the Round Trip Time (RRT) for the packets exchanged between the two sides. In each scenario, more than 100 values have been taken. Tests have been reproduced for two different hours and days.

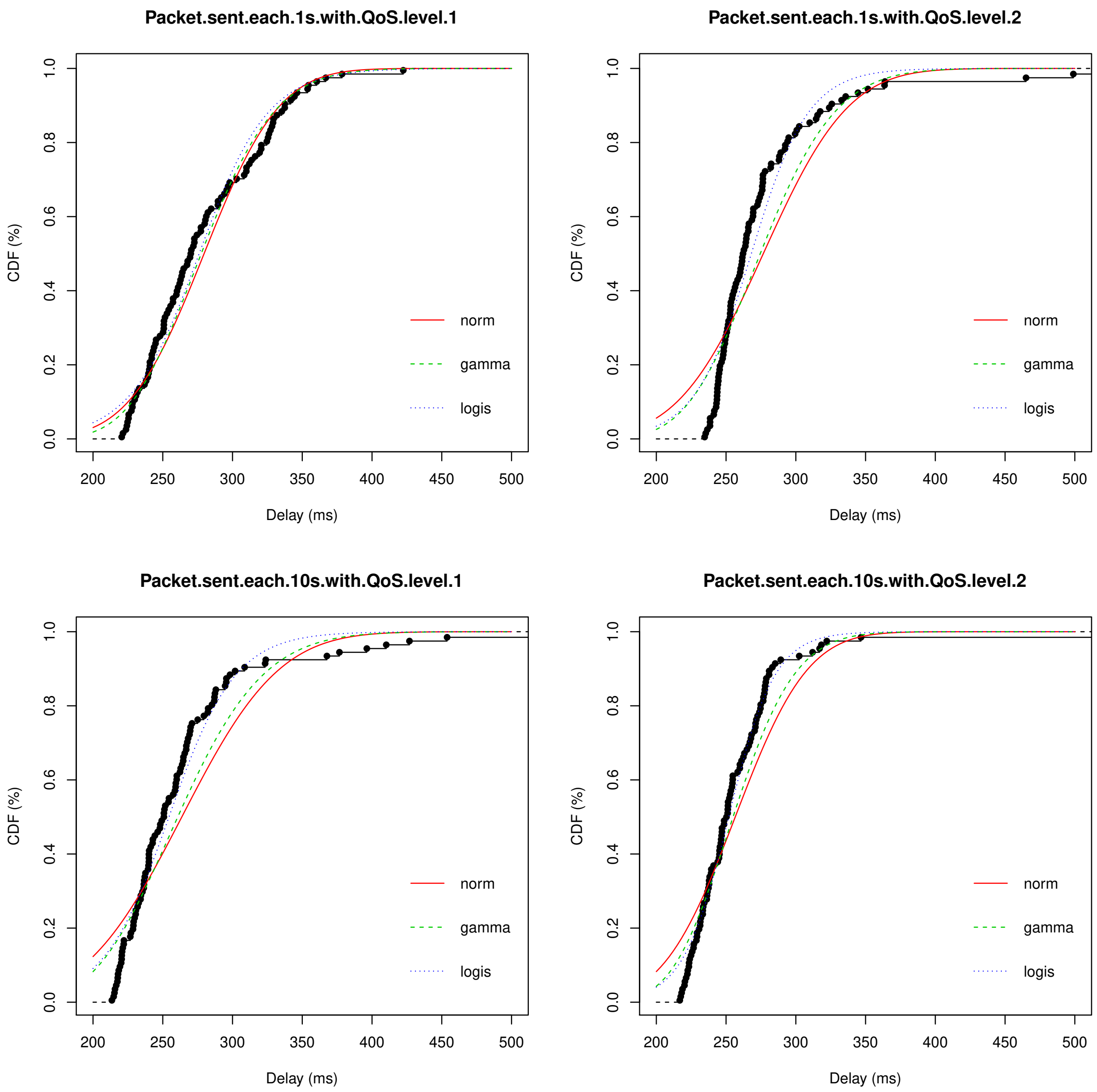


Figure 8: Normal, Gamma and Logistic distribution

Fig. 8 shows the measured Cumulative Distribution Function (CDF) of RTT delays for the two levels of QoS, levels 1 and 2. We consider three probabilistic distribution functions (Normal, Gamma and Logistic) in the experimental results in order to characterize MQTT performance. The obtained correlation allows us to define a representative empirical model. Table 9.1 details the results of a correlation matrix between distributions and experimental results. We can see that logistic distribution [**STEPHENS1979**] fits better with our measured RTT values for the two scenarios. The standard logistic law is of parameters 0 and 1. Its distribution function of a random variable x is the sigmoid following the expression:

$$\mathrm{F}(x) = \frac{1}{1+e^{-x}}, \text{where } x \in [-\infty, +\infty] \tag{9.1}$$

Fig. 9 highlights the empirical model of CDF featuring the RTT of MQTT protocol. As can be seen, RTT protocol is more efficient when the number of packets is greater than 35% of the total number of packets sent. This can be explained by the fact that priority queues are useful when queues of QoS are full. In that case, the packets with a highest priority level will reach their destination with low latency. Furthermore,

Table 9.1: Correlation between distributions and empirical results

	norm	gamma	logis
1s with QoS level 1	172.12074	175.2950	**185.4433**
1s with QoS level 2	159.59630	172.8193	**189.7002**
10s with QoS level 1	146.85668	161.2369	**175.3682**
10s with QoS level 2	176.28502	192.6108	**204.3235**

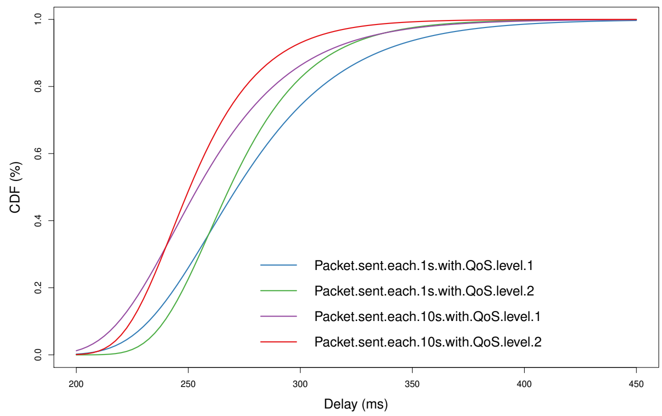


Figure 9: Cumulative distribution function of RTT delay for two QoS levels

when we increase the number of packets sent to Ubidots at one per second, the MQTT still offers the same efficiency with suitable delays. The found latency of up to 400 ms would be problematic for real world safety applications which require at most 100 ms [**Chen2017**].

Although our Mockup is innovative by combining IEEE 802.15.4, 6LowPAN, MQTT protocol and Edge Computing, the performances of our solution depend on external parameters. These parameters are related to Internet Service Provider and all packet's routes throughout Internet network from the Middleware to Ubidots. Thus, real implementation of such a system should be done by introducing a private Cloud near data sources.

9.7 Conclusion

In this paper, we proposed an Urban Traffic Light Control (IoT-UTLC), considering its architectures elements and tools used to build an IoT lockup. We reported three main contributions in this work: i) Modelling through UPPAAL of crossing's traffic lights, ii) Prototyping of IoT Edge Computing, iii) Performance assessment of MQTT protocol. Traffic lights control has been taken as an IoT network. WSN has been deployed on motes running Contiki Os and exchanging IEEE802.15.4/6LowPAN packets. MQTT was the QoS protocol between our WSN and Ubidots Cloud platform. UPPAAL model checker design ensured that lights' colors change is adaptive to the arriving of a priority vehicle.

Our experiments have investigated the relationship between the MQTT protocol and the traffic flow congestion. Our results showed that the MQTT is efficient when the number of packets sent exceeds 35% of the total number. The packets with the highest level of QoS has low latency than other packets. The protocol remains efficient since the delay of priority packets decreases when the network overhead increases. However, found latencies of up to 400ms is higher than the expected one for vehicular safety application. Thus, the proposed IoT architecture and protocols must be improved to consider the safety requirements.

While we are still developing our prototyping, we plane to integrate other use cases such as smart buildings and industrial IoT. As a near future work, we plane to extend our experiments with private Cloud towards a real Fog Computing.

Acknowledgement

We would like to thank our colleague Sebti Mouelhi, associate professor at ECE Paris, who provided us insight and expertise on UPPAAL.

10 | Conclusion

"Everything that has a beginning has an ending. Make your peace with that and all will be well" - Jack Kornfield

Contents

10.1 Conclusion

10.2 Perspectives

A | Appendix A

Appendix A

Contents

Abstract

A.1 Introduction

A.2 Related work

A.3 Related work

A.4 Approach

A.5 Experimentation

A.6 Results exploitation

A.7 Conclusion

B | Appendix B

Appendix B

Contents

B.1 Frame

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Preamble		Sync msg	PHY Header	PHDR-CRC	PHY Payload															CRC	
Modulation	length	Sync msg	PHY Header	PHDR-CRC	MAC Header			MAC Payload											MIC	CRC Type	Polynomial
Modulation	length	Sync msg	PHY Header	PHDR-CRC	MType	RFU	Major	Frame Header									FPort	**Frame Payload**	MIC	CRC Type	Polynomial
Modulation	length	Sync msg	PHY Header	PHDR-CRC	MType	RFU	Major	Dev Address		FCtrl					FCnt	FOpts	FPort	**Frame Payload**	MIC	CRC Type	Polynomial
Modulation	length	Sync msg	PHY Header	PHDR-CRC	MType	RFU	Major	NwkID	NwkAddr	ADR	ADRACKReq	ACK	FPending /RFU	FOptsLen	FCnt	FOpts	FPort	**Frame Payload**	MIC	CRC Type	Polynomial

- Preamble:
  - Modulation :
    - Lora: 8 Symbols, 0x34 (Sync Word)
    - FSK: 5 Bytes, 0xC194C1 (Sync Word)
  - Length :
- Sync msg
- PHY Header: It contains:
  - The Payload length (Bytes)
  - **The Code rate**
  - Optional 16bit CRC for payload
- Phy Header CRC It contains CRC of Physical Layer Header
- PHY Payload: It contains:
  - MAC Hearder
    - MType: is the message type (uplink or a downlink)
      - whether or not it is a confirmed message (reqst ack)
      - 000 Join Request
      - 001 Join Accept
      - 010 Unconfirmed Data Up
      - 011 Unconfirmed Data Down
      - 100 Confirmed Data Up
      - 101 Confirmed Data Down
      - 110 RFU
      - 111 Proprietary
    - RFU Reserved for Future Use
    - Major: is the LoRaWAN version; currently, only a value of zero is valid
      - 00 LoRaWAN R1
      - 01-11 RFU
  - MAC Payload :
    - Frame Header
      - DevAddr: the short address of the device
        - NwkID (Network ID): 31th to 25th
        - NwkAddr (Network Address): 24th to 0th
      - Fcrtl
        - ADR Network server will change the data rate through appropriate MAC commands
          - 1 To change the data rate
          - 0 No change
        - ADRACKReq (Adaptive Data Rate ACK Request): if network doesn't respont in 'ADR-ACK-DELAY' time, end-device switch to next lower data rate.
          - 1 if (ADR-ACK-CNT) >= (ADR-ACK-Limit)
          - 0 otherwise
        - ACK (Message Acknowledgement): If end-device is the sender then gateway will send the ACK in next receive window else if gateway is the sender then end-device will send the ACK in next transmission.
          - 1 if confirmed data message
          - 0 otherwise
        - FPending↓ /RFU ↑ : (Only in downlink), if gateway has more data pending to be send then it asks end-device to open another receive window ASAP
          - 1 to ask for more receive windows
          - 0 otherwise
        - FOptsLen is the length of the FOpts field in bytes ă 0000 to 1111
      - FCnt: 2 type of frame counters
        - FCntUp: counter for uplink data frame, MAX-FCNT-GAP
        - FCntDown: counter for downlink data frame, MAX-FCNY-GAP
      - FOpts is used to piggyback MAC commands on a data message
    - FPort: a multiplexing port field
      - 0 the payload contains only MAC commands
      - 1 to 223 Application Specific
      - 224 & 225 RFU
    - FRMPayload (Frame Payload) Encrypted (AES, 128 key length) Data
  - MIC: is a cryptographic message integrity code
    - computed over the fields MHDR, FHDR, FPort and the encrypted FRMPayload.
  - CRC (only in uplink),
    - CCITT $x^{16} + x^{12} + x^5 + 1$
    - IBM $x^{16} + x^{15} + x^5 + 1$

B.2 Application

B.2.1 CoAP

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Ver	T	TKL	Code	Message ID
Token				
Options				
11111111			Payload	

CoAP Header

Ver: is the version of CoAP
T: is the type of Transaction
TKL: Token length
Code: represents the request method (1-10) or response code (40-255).
- Ex: the code for GET, POST, PUT, and DELETE is 1, 2, 3, and 4, respectively.

Message ID: is a unique identifier for matching the response.
Token: Optional response matching token.

B.2.2 MQTT

0 1 2 3 4 5 6 7

Message Type	UDP	QoS Level	Retain
Remaining length			
Variable length header			
Variable length message payload			

CoAP Header

Message type: CONNECT (1), CONNACK (2), PUBLISH (3), SUBSCRIBE (8) and so on
DUP flag: indicates that the massage is duplicated
QoS Level: identify the three levels of QoS for delivery assurance of Publish messages
Retain field: retain the last received Publish message and submit it to new subscribers as a first message

B.2.3 XMPP

- Extensible Messaging and Presence Protocol
- Developed by the Jabber open source community
- An IETF instant messaging standard used for:
  - multi-party chatting, voice and telepresence
- Connects a client to a server using a XML stanzas
- An XML stanza is divided into 3 components:
  - message: fills the subject and body fields
  - presence: notifies customers of status updates
  - iq (info/query): pairs message senders and receivers
- Message stanzas identify:
  - the source (from) and destination (to) addresses
  - types, and IDs of XMPP entities

B.2.4 AMQP

Size the frame size.
DOFF the position of the body inside the frame.
Type the format and purpose of the frame.
- Ex: 0x00 show that the frame is an AMQP frame
- Ex: 0x01 represents a SASL frame.

B.2.5 DDS

- Data Distribution Service
- Developed by Object Management Group (OMG)
- Supports 23 QoS policies:
  - like security, urgency, priority, durability, reliability, etc
- Relies on a broker-less architecture
  - uses multicasting to bring excellent Quality of Service
  - real-time constraints
- DDS architecture defines two layers:

DLRL Data-Local Reconstruction Layer
  - serves as the interface to the DCPS functionalities

DCPS Data-Centric Publish/Subscribe
  - delivering the information to the subscribers

- 5 entities are involved with the data flow in the DCPS layer:
  - Publisher:disseminates data
  - DataWriter: used by app to interact with the publisher
  - Subscriber: receives published data and delivers them to app
  - DataReader: employed by Subscriber to access received data
  - Topic: relate DataWriters to DataReaders
- No need for manual reconfiguration or extra administration
- It is able to run without infrastructure
- It is able to continue working if failure happens.
- It inquires names by sending an IP multicast message to all the nodes in the local domain
  - Clients asks devices that have the given name to reply back
  - the target machine receives its name and multicasts its IP @
  - Devices update their cache with the given name and IP @

B.2.6 mDNS

- Requires zero configuration aids to connect machine
- It uses mDNS to send DNS packets to specific multicast addresses through UDP
- There are two main steps to process Service Discovery:
  - finding host names of required services such as printers
  - pairing IP addresses with their host names using mDNS
- Advantages
  - IoT needs an architecture without dependency on a configuration mechanism
  - smart devices can join the platform or leave it without affecting the behavior of the whole system
- Drawbacks
  - Need for caching DNS entries

B.3 Divers

Year		**Factors**	**Computation Model**	**Results interpretation**
2018 jhjhjhjhjhjhjhj	[42]	-Closeness Centrality -Degree Centrality	Estimation	**Closeness** have a high degree of kjkjkjkjkjkj correlation with **privacy score**

Table B.1: Social metrics

Naïve modes	Instantaneous Hist. average Clustering
Parametric models	Rarely used Traffic Models Time Series Linear regression ARIMA Kalman filtering ATHENA SETAR Gaussian Maximum Likelihood
Non-Parametric models	k-Nearest Neighbor Locally Weighted Regression Fuzzy Logic Bayes Network Neural Network Include temporal/spatial patterns

Table B.2: Taxonomy of prediction models [**_short_2007**]

- Network selection
  - MADM
    - Ranking methods
    - Ranking & weighted methods
  - Game theory
    - Users vs users
    - Users vs networks
    - Networks vs network
  - Fuzzy logic
    - as a score method
    - another theory
  - Utility function
    - 1
    - 2

Signal-to-interference & noise ratio (SINR) Signal-to-Interference Ratio (SIR) BR DR BW BER PER PRR Packet delivery ratio (PDR) SNR Packet loss rate (PLR) Round time trip (RTT) Transmission Energy (Tx) Payload size (PS) Traffic congestion (TC) Duty cycle (DC) SR Sleep time (SL) Jitter (Jit) Co-channel Interference (CCI) Time on Air (ToA) PL Mobility (Mob) Throughput (Th) Service Cost (SC)

Layer	**Maximize (Reward)**	**Minimize (Cost)**
Application	Sec security Ergonomic	SC Service Cost
Network	BW Bandwidth available PRR Packet Reception Ratio Th Throughput Range Network coverage Availability Availability DR Data rate PDR Packet delivery ratio	PS Payload size Jit Jitter TC Traffic congestion PLR Packet loss rate RTT Round time trip $\mathcal{O}_{time}$ Time Complexity $\mathcal{O}_{space}$ Space Complexity
Radio	SNR Signal-to-noise ratio DC Duty cycle SR Symbol rate Mob Mobility BR Bit rate SINR signal-to-interference & noise ratio SIR Signal-to-Interference Ratio	BER Bit error rate SL Sleep time Tx Transmission Energy CCI Co-channel Interference PL Path loss ToA Time on Air

Table B.3: Network selection inputs and classification of parameters [43] + QoS parameters [44] [45]

[46]

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(p_i - r_i\right)^2 \tag{B.1}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(p_i - r_i\right)^2} \tag{B.2}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|p_i - r_i\right| \tag{B.3}$$

$$Recall = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{B.4}$$

$$Precision = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{B.5}$$

$$F1\_Score = \frac{2 \times \text{ Precision } \times \text{ Recall}}{\text{precision } + \text{recall}} \tag{B.6}$$

$$\text{TPR} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{B.7}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP}+\text{TN}} \tag{B.8}$$

$$\text{ROC} = (\text{TPR}, \text{FPR}) \tag{B.9}$$

$$Novelty = \sum_{i\in \text{L}} \frac{\log_2 \text{P}_i}{n} \quad \text{where} \quad \text{P}_i = \frac{n - rank_i}{n-1} \tag{B.10}$$

$$Serendipity = \frac{1}{n}\sum_{i\in n} \max(\text{P}_{\text{user}} - \text{P}_{\text{U}}, 0) \times rel_i \tag{B.11}$$

$$diversity = \frac{a}{c}\sum_{i=1}^{c}\frac{1}{n}\sum_{j=1}^{n} i_j \tag{B.12}$$

$$Coverage = 100 \times \frac{u}{\text{U}} \tag{B.13}$$

$$Stability = \frac{1}{\text{P}_2}\sum_{i\in \text{P}_2}\left|\text{P}_{2,i} - \text{P}_{1,i}\right| \tag{B.14}$$

$$\text{DCG} = rel_1 + \sum_{i=2}^{\text{pos}} \frac{rel_i}{\log_2 i} \tag{B.15}$$

$$\text{IDCG} = rel_1 + \sum_{i=2}^{|h|-1} \frac{rel_i}{\log_2 i} \tag{B.16}$$

$$\text{NDCG} = \frac{\text{DCG}}{\text{IDCG}} \tag{B.17}$$

(B.18)

Part II

Privacy

C | Social privacy score through vulnerability contagion process

Social privacy score through vulnerability contagion process

Contents

Abstract

The exponential usage of messaging services for communication raises many questions in privacy fields. Privacy issues in such services strongly depend on the graph-theoretical properties of users' interactions representing the real friendships between users. One of the most important issues of privacy is that users may disclose information of other users beyond the scope of the interaction, without realizing that such information could be aggregated to reveal sensitive information. Determining vulnerable interactions from non-vulnerable ones is difficult due to the lack of awareness mechanisms.

To address this problem, we analyze the topological relationships with the level of trust between users to notify each of them about their vulnerable social interactions. Particularly, we analyze the impact of trusting vulnerable friends in infecting other users' privacy concerns by modeling a new vulnerability contagion process. Simulation results show that over-trusting vulnerable users speeds the vulnerability diffusion process through the network. Furthermore, vulnerable users with high reputation level lead to a high convergence level of infection, this means that the vulnerability contagion process infects the biggest number of users when vulnerable users get a high level of trust from their interlocutors. This work contributes to the development of privacy awareness framework that can alert users of the potential private information leakages in their communications.

C.1 Introduction

With increasing frequency, communication between citizens and institutions occurs via some type of e-mechanisms, such as websites, email, and social media. In particular, email platforms are widely being adopted because of their simplicity of use. Due to the social aspect of these mechanisms, users are continuously infected by their friends' privacy vulnerability. Users can take all the required measures to protect themselves from potential information leakage, but if their friends didn't respect the same measures, this indirectly harms their privacy concerns, especially when they grant a high-level of trust to them.

Currently, available solutions address the privacy issues of users by measuring their vulnerability toward active attackers in low layer protocols (e.g. HTTPS, SSL, PGP, IPsec, etc), or by suggesting new privacy policy settings of their applications. These works are efficient to protect users from external vulnerabilities, but they appear very weak to protect users from (legitimate) information leakage between messaging services users. Many other works [**liu_framework_2010**] address this problem by measuring the users' privacy vulnerability individually without caring about the social context of the problem. Few works [47] [48] address this problem from a topological view of users' relationships during their social interactions. Our work is motivated by the potential of privacy awareness frameworks to help users being conscious about the trustworthiness of their social interactions.

Trust networks allow users to rate other users, they can put their level of trust in their interlocutors based on their own beliefs such as Alice trust Bob as 0.8 in [0,1] [49]. Trust statements can then be aggregated in a single trust network representing the relationships between users [49]. Trust metrics in our work are related to the relation strength between users such as the frequency of interactions, common interests, common friends, etc. Trust metrics can also be related to the relationship closeness such as family, friends, colleagues or just unknown. Based on such metrics and the topology of the interaction network, the system can suggest how many users are trustworthy based on different opinions of interlocutors, this suggestion represents their reputation.

Trust and reputation metrics are used in our work in order to study the relationship between them and users' privacy vulnerability. Reputation concept refers to the extent to which a user is trustworthy. This means that he plays a central role in preserving or revealing sensitive information of his interlocutors. Reputation system collects, distributes and aggregates feedback about participants past behavior to allow users decide whom to trust and with whom to exchange sensitive information, users could then decide to not interact with those who are vulnerable to preserve their own privacy.

Messaging services users often exchange messages with a high number of users without caring about the vulnerability of their social environment. In this paper, we deal with privacy issues by studying the impact of trust in preserving privacy.

The remainder of this paper is organized as follows. Section C.3 elucidates summary of related works. In Section C.4, we propose our vulnerability contagion process to reveal the social vulnerability of users. Our experimentation with Enron data set and our findings are presented in Section C.5 and C.6 respectively. Finally, conclusion and future works are drawn in Section C.7.

C.2 Related work

To evaluate the privacy risk of social network users, trust metrics are used to measure the extent to which users can be trusted. Trust metrics can be classified into two main categories: global and local trust metrics.

Local trust metrics, compute trust values that are dependent on the target user, they take into account the very personal and subjective views of the users, they predict different values of trust for every single user based on their own experience. Global trust metrics, on the other hand, predict a global reputation value for each node, based rather on the experience of all other users or on the topology of the social network.

While much work has focused on tools for understanding and adjusting existing privacy settings, Protect_U [50] uses machine learning techniques to recommend privacy settings based on a users personal data and trustworthy friends. Protect_U analyzes user profile contents and ranks them according to four risk levels: Low Risk, Medium Risk, Risky and Critical. The system then suggests personalized recommendations to allow users making their accounts safer. In order to achieve this, it draws upon two protection models: local and community-based. The first model uses the users personal data in order to suggest recommendations, The second model seeks the users trustworthy friends to encourage them to help improve the safety of their counter parts account.

Despite the mole of work on social trust, Social Market [51] is the first system to propose the use of trust relationships to build a decentralized interest-based marketplace. Similarly, TAPE [52] is the first attempt to combine explicit and implicit social networks into a single gossip protocol. Zeng et al. [52] approaches the privacy quantification problem from a different angle. First, they consider how likely a friend reveals others personal information, by computing the privacy trust score, which is a widely studied research problem [53]. Furthermore, the proposed work is related to information diffusion in OSNs such as [54]. TAPE framework differs from other work, in considering information diffusion in the context of privacy protection, which requires different sets of features and considerations.

Zeng and Xing [47] studied how individual users can expand their social networks by making trustful friends who will not leak their private information to unknown parties. This work proposes a security risk estimation framework of social networking privacy to calculate the probability of individual privacy leakage through the social graph. The framework is composed of two parts, the calculation of Individual Privacy Leakage Probability (IPLP) and the Relationship Privacy Leakage Probability (RPLP). Relationship Privacy Leakage Probability considers the factors of relationship strength and interactive behaviors. Two vectors namely privacy protection awareness (PPA) and privacy protection trust (PPT) are proposed in this paper to estimate IPLP.

Ostra [55] utilizes trust relationship to thwart unwanted communication, where the number of a users trust relationships is used to limit the number of unwanted communications he can produce. Ostra utilizes the existing trust relationship among users to charge the senders of unwanted messages and thus block spam. It relies on existing trust networks to connect senders and receivers via chains of a pair-wise trust relationship, they use a pair-wise link-based credit scheme to impose a cost on the originator of the unwanted communication. Unfortunately, the scalability of this system stays uncertain as it employs a per-link credit scheme.

Gundecha et al. [53] propose a feasible approach to the problem of identifying a users vulnerable friends on a social networking site. Vulnerability is somewhat contagious in this context. Their work differs from existing work addressing social networking privacy by introducing a vulnerability-centered approach to a user security on a social networking site. On most social networking sites, privacy-related efforts have been concentrated on protecting individual attributes only. However, users are often vulnerable through community attributes. Unfriending vulnerable friends can help protect users against the security risks.

Hameed [56] proposed LENS, which extends the friend of friend network by adding trusted users from outside of the FoF networks to mitigate spam beyond social circles. Only emails to a recipient that have been vouched by the trusted nodes can be sent into the network. The authors proposed using social networks and trust and reputation systems to combat spam. In contrast, LENS can reject unwanted email traffic during the SMTP time.

SocialEmail [57] considers trust as an integral part of networking rather than working alongside an existing communication system. SocialEmail leverages social network trust paths to rate the messages. The key feature of SocialEmail is that instead of directly connecting the sender and the recipient, messages are routed through existing friendship links. This gives each email recipient control over who can message him/her, In contrast, such social interaction-based methods are not sufficiently effective in dealing with legitimate emails from senders outside of the social network of the receiver.

Social interactions (e.g., the exchange of messages between users) have been suggested as an indicator of interpersonal tie strength [58]. As a consequence, an unsupervised model has been developed to estimate the relationship strength from the interaction activity and the user similarity in the OSN [58]. Although interaction-based methods leverage social relationships for extracting trust, the applications are not designed to be automated in the sense that the user must explicitly score other users, score messages, create whitelists or adjust the credits.

Vidyalakshmi et al. [48] proposed a privacy control framework for information dispersal on social network, they use the quadratic form of bezier curve to arrive at privacy scores for friends, they use the communication information for pre-sorting of friends which is lacking in [59]. Similarly, Akcora et al. [60] develop a graph-based approach and a risk model to learn risk labels of strangers, the intuition of such an approach is that risky strangers are more likely to violate privacy constraints.

Privacy Index (PIDX) proposed in [61] is a measure of a users privacy exposure in a social network. PIDX is a numerical value between 0 and 100 with a high value indicating high privacy risk in social networks. An attributes privacy impact factor is a ratio of its privacy impact to full privacy disclosure. Thus, an attributes privacy impact has a value between 0 and 1. They consider the privacy impact factor for full privacy disclosure is 1.

Fang and Le Fevre [54] proposed a Privacy Wizard to help users grant privileges to their friends. the goal of this tool is to automatically configure a users privacy settings with minimal effort and interaction from the user. The wizard asks users to first assign privacy labels to selected friends, and then uses this as input to construct a classifier which classifies friends based on their profiles and automatically assign privacy labels to the unlabeled friends. In a similar way, some studies [62] propose a methodology for quantifying the risk posed by a users privacy settings. A risk score reveals to the user how far her privacy settings are from those of other users. It provides feedback regarding the state of her existing settings. However, it does not help the user refine her settings in order to achieve a more acceptable configuration.

Abdul-Rahman and Hailes The trust model presented by Abdul-Rahman and Hailes [63] is focused on virtual communities related to e-commerce and artificial autonomous agents. The model defines direct trust and recommender trust. Direct trust is the trust of an entity in another one based on direct experience. Whereas recommender trust is the trust of an entity in the ability to provide good recommendations. Trust can only have discrete labeled values, namely Very Trustworthy, Trustworthy, Untrustworthy, and, Very Untrustworthy for direct trust, and Very good, good, bad and, very bad for recommender trust. The difference between the two ratings from different entities can be computed as semantic distance. This semantic distance can be used to adjust further recommendations. The combination of ratings is done as a weighted sum, where the weights depend on the recommender trust.

All previous work didn't take into consideration the topological aspect of interactions to measure social vulnerabilities of users. The closest study to our approach is that presented in [47]. However, this solution doesn't study the social interaction environment of users and the potential information leakage through a vulnerable social environment. In this paper, we study the impact of trusting vulnerable users in preserving the privacy of all users in the communication network.

C.3 Related work

To evaluate the privacy risk of social network users, trust metrics are used to measure the extent to which users can be trusted. Trust metrics can be classified into two main categories: global and local trust metrics. Local trust metrics compute trust values that are dependent on the target user. They take into account the very personal and subjective views of the users. They predict different values of trust for every single user based on their own experience. Global trust metrics, on the other hand, predict a global reputation value for each node, based on both the experience of all other users and the topology of the social network.

Much work has focused on adapting existing privacy settings to the users profile, for example, the machine learning techniques used in Protect_U [50] allow recommending privacy settings based on trustworthy friends. Protect_U analyzes the existing privacy settings and ranks them under four risk levels: Low Risk, Medium Risk, Risky and Critical. The system then suggests personalized recommendations to allow users to make their accounts safer. In order to achieve this, it draws upon two protection models: local and community-based. The first model uses the visibility of users profile information to suggest recommendations. The second model searches trustworthy friends to encourage them to help improve the safety of their friends account.

Despite a large amount of work on social trust, a decentralized interest-based marketplace is built for the first time in SocialMarket [51], authors of this framework propose the use of trust relationships to build their decentralized interest-based marketplace. In contrast, TAPE framework developed by Zeng et al. [52] tried to solve the privacy quantification problem [53] from a different angle. First, they calculate the privacy trust score of each user to know how likely a friend could reveal or preserve others personal information. Next, they propose an information diffusion process [54]. The most important contribution made by TAPE framework [52] is in considering information diffusion to reveal privacy leakage in communication.

Zeng and Xing [47] studied the maximization of users relationships by making trustful friends without leaking their private information to unwanted parties. The authors propose a security risk estimation framework to deal with such a problem. The framework proposed is composed of two parts, the calculation of Individual Privacy Leakage Probability (IPLP) and the Relationship Privacy Leakage Probability (RPLP). Two vectors namely privacy protection awareness (PPA) and privacy protection trust (PPT) are proposed in this paper to estimate IPLP.

Gundecha et al. [53] propose an advantageous approach to the problem of identifying a users vulnerable friends. The approach proposed differs from existing work by integrating a vulnerability-centered approach. On most online social networks (OSN), a mole of work addressed the problem of protecting user's individual attributes only, however, few works address the problem of protecting users relationships from vulnerable friends.

Another example that consolidates our intention to build our privacy awareness framework is called LENS [56]. Hameed [56] proposed a novel spam protection system that maximizes the number of trusted nodes who send only trusted emails, only emails to a node that have been allowed by these trusted nodes can be sent through the network. The authors proposed using trust and reputation systems to detect whether a user is trustful or not.

SocialEmail [57] is a trust by design communication system, it considers trust as an integral part of the communication system. SocialEmail rank trust paths to rate the messages, these last ones are routed through existing friendship links that are weighted with different trust level. This gives each email recipient an overview of the trustworthiness of path taken by a message to reach him. In contrast, such methods are not sufficiently effective in dealing with legitimate emails from senders outside SocialEmail.

Social interactions allow users exchanging messages with other users easily and quickly. xiang et al. [58] proposed a social interaction as an indicator of interpersonal tie strength. As a consequence, an unsupervised model has been developed to estimate the relationship strength from their interaction activity [58].

Such methods could extract the level of trust between interlocutors based on the relationship strength between them. However, this application is not designed to be automated, users must manually score other users, score messages or create whitelists.

Vidyalakshmi et al. [48] proposed a privacy control framework for information dispersal on social networks, they use the quadratic form of bezier curve to arrive at privacy scores for friends, they use the communication information for pre-sorting friends which is lacking in [59]. Similarly, Akcora et al. [60] develop a graph-based approach and a risk model to learn risk labels of strangers, the intuition of such an approach is that risky strangers are more likely to violate privacy constraints.

Privacy Index (PIDX) proposed in [61] is a measure of a users privacy exposure in a social network. PIDX is a numerical value between 0 and 100 with a high value indicating high privacy risk in social networks. An attributes privacy impact factor is a ratio of its privacy impact to full privacy disclosure. Thus, an attributes privacy impact has a value between 0 and 1. They consider the privacy impact factor for full privacy disclosure is 1.

Fang and Le Fevre [54] proposed a Privacy Wizard to help users grant privileges to their friends. It automatically configures users' privacy settings. The wizard asks users to first assign trust labels to selected friends. The wizard is then training to classify tagged friends based on both profile information and labels assigned by users themselves. Ones this step is finished, the wizard is now able to assign new labels to unlabeled friends based only on their profile information. In a similar way, some studies [62] propose a methodology to quantify the vulnerability of users privacy settings. A risk score reveals to users how far their privacy settings are from those of other users. However, it does not help users refine their settings in order to achieve a more acceptable configuration.

Abdul-Rahman and Hailes [63] proposed a trust model with virtual communities and artificial autonomous agents. The model defines a direct trust and a recommender trust. Trust can only have discrete labeled values, namely Very Trustworthy, Trustworthy, Untrustworthy, and, Very Untrustworthy for direct trust, and Very good, good, bad and, very bad for recommender trust. The difference between the two ratings from different entities can be computed as a semantic distance that can be used to adjust further recommendations. The combination of ratings is done as a weighted sum, where the weights depend on the recommender trust.

All previous work didn't take into consideration the topological aspect of interactions to measure social vulnerabilities of users. The closest study to our approach is that presented in [47]. However, this solution doesn't study the impact of having interactions with vulnerable users. In this paper, we study the impact of trusting vulnerable users in preserving the privacy of all users in the communication network.

C.4 Approach

The aim of this study is to understand how trust coefficient affects social privacy vulnerability. In this section, we present our vulnerability contagion process in messaging services in order to get social vulnerability scores of users. These scores represent the vulnerability within the social environment of each user. In our study, we focus on the impact of trust and reputation in spreading out the privacy vulnerability.

To model this process we measure the impact of vulnerable users in protecting friends privacy. For example, let us say that a user is exchanging messages with five friends as shown in Figure 1, the reputation of this user is given as the probability to be trusted by his friends (Figure 1a). The more trustworthy a user is, the more reputed he becomes. However, if a user with a high reputation level has a high vulnerability, he can spread his vulnerability with a high infection coefficient (Figure 1b). As a consequence, the social vulnerability of users is calculated as the level of the contagion degree of each user in the communication network. To get these values, a weighted matrix value M is used as an adjacency matrix normalized by users' degree. Social vulnerability is calculated in a continuous space, this means that we iterate the vulnerability contagion process until the process converges as shown in Figure 2.

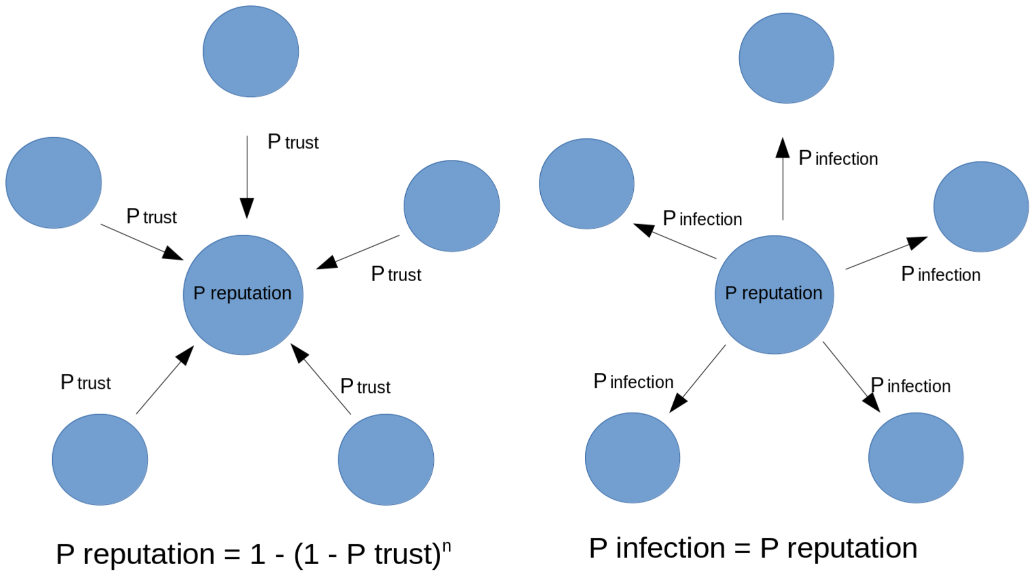
The probability of the infection is based on the trust level between users.

To evaluate the impact of trust in this process we add a reputation parameter $p_{reputation}$, this parameter is used to know how likely a user could be trusted by his friends, it is calculated as the probability to get at least one trust grant from them.

$$p_{reputation} = p(\mathrm{X} \geq 1) \;=\; 1 \;-\; (1 \;-\; p_{trust})^{n} \qquad \text{(C.1)}$$

Where,

- X: Number of trust grant from friends, $\mathrm{X} \sim \mathrm{B}(n, p)$.
- n = number of user's friends.
- $p_{trust} = p(\mathrm{X} = 1)$: Probability to get one trust grant from a friend.



(a) Trust grant. (b) Vulnerability contagion process.

Figure 1: Reputation coefficient.

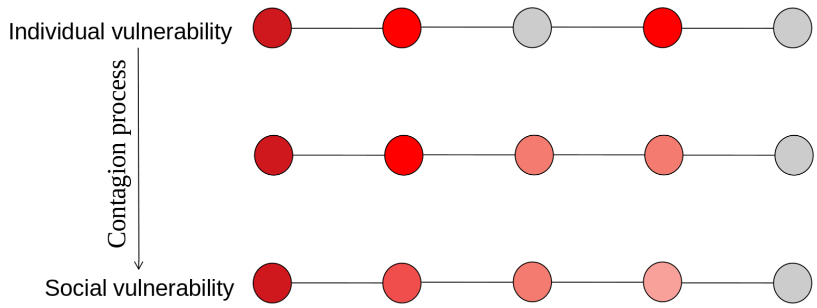


Figure 2: Contagion & peer influence example.

Trust parameter, in this equation, is added as a coefficient parameter to increase or decrease the vulnerability contagion process. Whether a user can infect other users social vulnerability scores depends on the trust level between them, so users should distrust vulnerable users to preserve their own privacy and the privacy of the entire communication network.

The number of friends infected by each user u in each step of the vulnerability contagion process is given as:

$$new\ infected = old\ infected + p_{reputation} * degree(u)$$

Initial privacy vulnerability scores of each user are given as input to our algorithm to estimate social privacy vulnerability scores, a normal distribution is used to generate initial privacy vulnerability scores. The vulnerability contagion equation is given as:

$$\mathrm{P}_{i+1} = p_{reputation} * (\mathrm{M} * \mathrm{P}_i) + (1 - p_{reputation}) * \mathrm{P}_i \tag{C.2}$$

Where,

- P_0 is the initial individual privacy vulnerability scores of each user.
- M is the adjacency matrix normalized by users' degree.
- i is the diffusion process iteration level.

The first part of this equation computes the vulnerability of a user based on his friends' average vulnerabilities weighted by the trust level with them. The second part computes the vulnerability of a user based only on his own vulnerability weighted by his friends' distrust level. Social vulnerability is a function of friends vulnerability and the trust coefficient. Mean distances between privacy vulnerability scores of each iteration is calculated to get the convergence process shown in Figure 4.

C.5 Experimentation

To evaluate our contagion process, we used Enron dataset in our experimentation. There are many reasons for using Enron dataset to evaluate our vulnerability measure techniques. First of all, it is probably the only actual corporate messaging service dataset available to the public. Second, it contains all kind of emails "personal and official", so email logs can reveal the level of trust between users by studying the information flow in the network. Finally, this dataset is similar to the kind of data collected for fraud detection or counter-terrorism, hence, it is a perfect test bed for testing the effectiveness of new vulnerability measure techniques.

The properties of the Enron dataset used for our experimentation are presented in table C.1.

Parameter	Value
Nodes	958
Edges	6966
Diameter	958
Mean degree	2.413361
Edge density	0.00252
Modularity	0.654600
Mean distance	3.042114

Table C.1: Enron dataset properties.

Due to visibility issues, we extract only important nodes given in [36]. Our substantive interest in this experimentation is in how vulnerability moves through the network. The inputs of our experimentation consist of a set of individual vulnerabilities of users in the network. This set is generated randomly and is represented in the graph of Figure 5a, values of this set are between 0 and 1 and represent the extent to which users are exposed to different kind of attacks such as (phishing, spam, etc). Unlike social vulnerability values, these values didn't take into account the vulnerability contagion process between users. To evaluate how trust coefficient affects our outputs i.e., social privacy vulnerabilities, we variate the trust coefficient through 4 symmetric values 0.2, 0.4, 0.6 and 0.8.

C.6 Results exploitation

Below we report the results of applying the contagion process model to the Enron Email dataset.

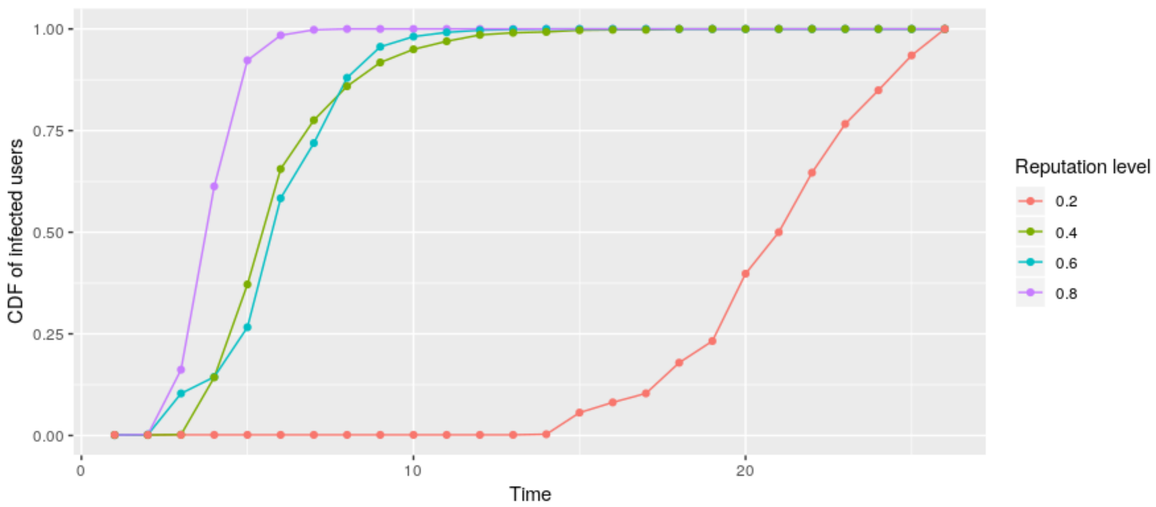


Figure 3: Cumulative distribution function of infected users.

Figure 3 shows the cumulative distribution function of the vulnerability contagion process, it appears clearly that the vulnerability diffusion process increases as the reputation level of vulnerable users increases, because when the vulnerability contagion process is at its 7^{th} iteration, the cumulative distribution function with the highest user's reputation level (0.8) is 1, this means that after the 7^{th} iteration, all users are infected. This is not the case of users with low reputation level which need further contagion steps to diffuse their vulnerability widely in the network. As a consequence, we can say that vulnerability contagion process speed is highly correlated with users reputation, it infects a large number of users quickly when the user's reputation level tends to 1.

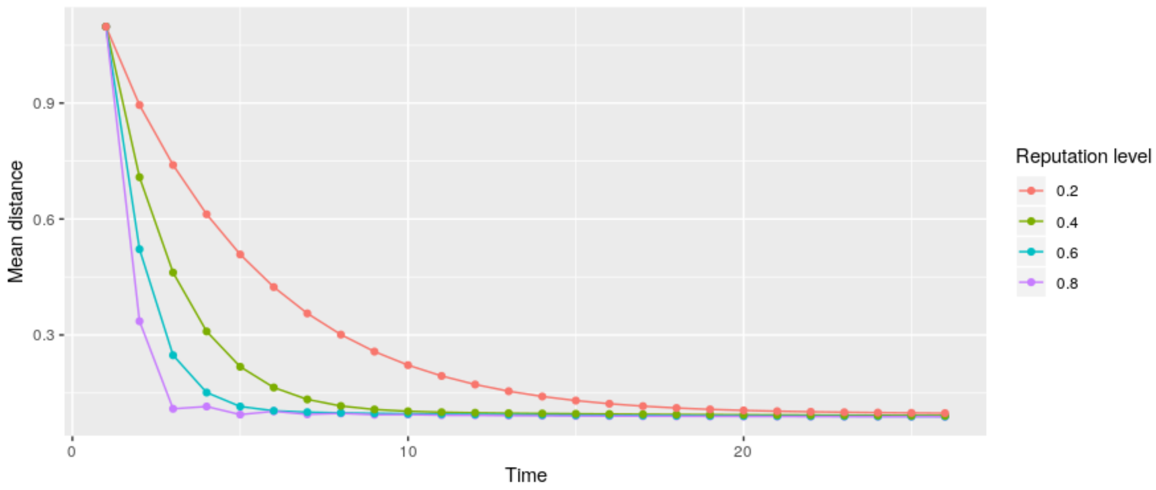


Figure 4: Contagion process convergence.

Figure 4 shows the convergence of the vulnerability contagion process, the mean distance between users' social vulnerability scores in each iteration is calculated to see the convergence of the process. The convergence process shows a high convergence level when the reputation coefficient is 0.8. This happens when the mean distance between the users' social vulnerability scores calculated at each iteration still the same. In other words, there are no more users to affect and the contagion process is at its high level. After viewing these tow graphs, we can conclude that over-trusting vulnerable users allow them to get a high reputation level and consequently infects the entire social vulnerability scores.

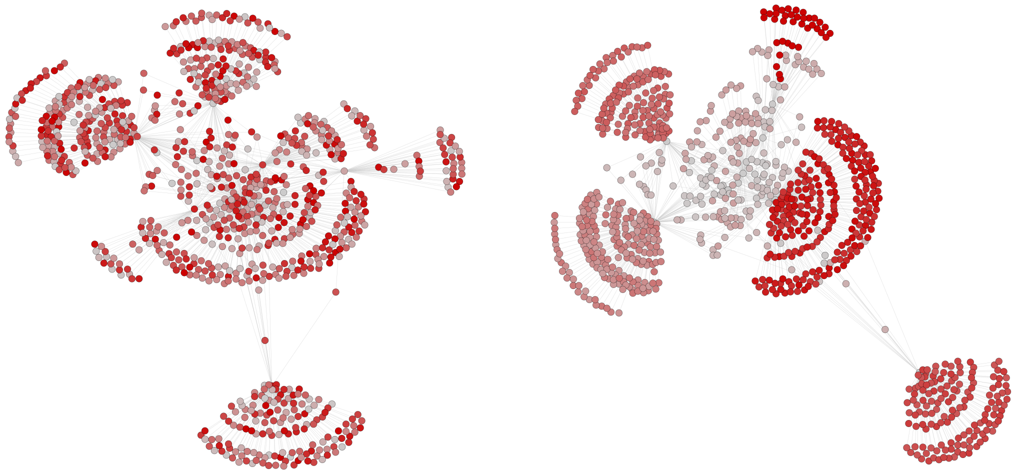
The initial and final measures of the simulation are represented in Figures 5. Figure 5a shows the initial privacy scores, these scores are generated randomly to illustrate the user's individual privacy vulnerability without caring about their friends' vulnerability. Figure 5b shows the final social privacy scores of the contagion process, these scores reveal the social vulnerability of users. Users with dark color are more vulnerable than others with light color in terms of friendship with other users. As a consequence, they have a high level of social vulnerability scores.

User ID	Individual vulnerability	Social vulnerability
34	0.84	0.67
67	0.12	0.87
206	0.76	0.33
588	0.23	0.78

Table C.2: Difference between Individual & Social privacy vulnerabilities.

Table C.2 illustrates the input (Individual vulnerability) and the output (Social vulnerability) values of four arbitrary users of our dataset. Results show that users with low individual vulnerability (e.g. user 67) could have a high social vulnerability due to their interactions with vulnerable users. In contrast, users with high individual vulnerability (e.g. user 206) can, in turn, be less vulnerable from interactions with other users, but harms considerably the social vulnerability of their interlocutors.

In summary, results presented in this section show that if the trust coefficient between users is up to 0.8, the vulnerability diffusion process through trust relationship is at its high level of speed. This what happens when a new information appears in a communication network and users forward it largely in the network. In addition, this work gives a new insight to understand the relationship between trust, reputation, individual vulnerability and social vulnerability in the context of messaging services such as emails.



(a) Individual privacy vulnerability.　　(b) Social privacy vulnerability.

Figure 5: Individual & Social privacy vulnerabilities.

C.7 Conclusions

This paper studies the relationship between trust and reputation metrics in users' interactions and the social privacy vulnerability of users. We observe that such metrics, severely affect users privacy in regard to their social relationships. Trust coefficient in the social network graph plays an essential role in spreading vulnerability. Our experiment investigates the probability of infecting other's privacy by increasing the reputation of vulnerable users. In this case, the number of nodes infected depends on the trust grant assigned to vulnerable users. Our experiment reveals also that the social vulnerability of users could be extracted from their individual vulnerability by applying our vulnerability contagion process. Privacy-preserving in online social network architectures should address this problem by discouraging trusting vulnerable friends.

D | Privacy Framework

Privacy Framework

Contents

Abstract

The exponential usage of messaging services for communication raises many questions in privacy fields. Privacy issues in such services are strongly related to the graph-theoretical properties of users' interactions representing the real friendships between users. One of the most important issues of privacy is that users may disclose information of other users beyond the scope of their interaction, without realizing that such information could be aggregated to reveal sensitive information. Determining vulnerable interactions from non-vulnerable ones is difficult due to the lack of awareness mechanisms.

To address this problem, we analyze the topological trust relationships between users to notify each of them about their vulnerable social interactions. Particularly, we analyze the impact of trusting vulnerable friends in affecting other users' privacy concerns by modeling a new vulnerability diffusion process. Simulation results show that over-trusting vulnerable users speeds the vulnerability diffusion process through the network. Furthermore, vulnerable users with high reputation level spread their vulnerability widely trough the network, this means that the vulnerability diffusion process affects the biggest number of users when vulnerable users get a high level of trust from their interlocutors. This work contributes to the development of privacy awareness framework that can alert users of the potential private information leakages in their communications.

D.1 Introduction

With increasing frequency, communication between citizens and institutions occurs via some type of e-mechanisms, such as websites, email, and social media. In particular, email platforms are widely being adopted because of their simplicity of use. Due to the social aspect of these mechanisms, users are continuously affected by their friends' privacy vulnerability. Users can take all the required measures to protect themselves from potential information leakage, but if their friends didn't respect the same measures, this indirectly harms their privacy concerns, especially when they grant a high-level of trust to them.

Currently, available solutions address the privacy issues of users by measuring their vulnerability toward active attackers in low layer protocols (e.g. HTTPS, SSL, PGP, IPsec, etc), or by suggesting new privacy policy settings of their applications. These works are efficient to protect users from external vulnerabilities, but they appear very weak to protect users from (legitimate) information leakage between messaging services users. Many other works, for example [**liu_framework_2010**], address this problem by measuring the users' privacy vulnerability individually without caring about the social context of the problem. Few works [47] [48] address this problem from a topological view of users' relationships during their social interactions. Our work is motivated by the potential of privacy awareness frameworks to help users being conscious about the trustworthiness of their social interactions.

Trust networks allow users to rate other users, they can put their level of trust in their interlocutors based on their own beliefs such as "Alice trust Bob as 0.8 in [0,1]" [49]. Trust statements can then be aggregated in a single trust network representing the relationships between users [49]. Trust metrics in our work are related to the relation strength between users such as the frequency of interactions, common interests, common friends, etc. Trust metrics can also be related to the relationship closeness such as family, friends, colleagues or just unknown. Based on such metrics and the topology of the interaction network, the system can suggest how many users are trustworthy based on different opinions of interlocutors, this suggestion represents their reputation.

Trust and reputation metrics are used in our work in order to study the relationship between them and users' privacy vulnerability. Reputation concept refers to the extent to which a user is trustworthy. This means that he plays a central role in preserving or revealing sensitive information of his interlocutors. Reputation system collects, distributes and aggregates feedback about participants past behavior to allow users decide whom to trust and with whom to exchange sensitive information, users could then decide to not interact with those who are vulnerable to preserve their own privacy.

Messaging services users often exchange messages with a high number of users without caring about the vulnerability of their social environment. In this paper, we deal with privacy issues by studying the impact of trust in preserving privacy.

The remainder of this paper is organized as follows. Section D.3 elucidates summary of related works. In Section D.4, we propose our vulnerability contagion process to reveal the social vulnerability of users. Our experimentation with Enron dataset and our findings are presented in Section D.5 and D.6 respectively. Finally, conclusion and future works are drawn in Section D.7.

D.2 Related work

To evaluate the privacy risk of social network users, trust metrics are used to measure the extent to which users can be trusted. Trust metrics can be classified into two main categories: global and local trust metrics.

Local trust metrics, compute trust values that are dependent on the target user, they take into account the very personal and subjective views of the users, they predict different values of trust for every single user based on their own experience. Global trust metrics, on the other hand, predict a global reputation value for each node, based rather on the experience of all other users or on the topology of the social network.

While much work has focused on tools for understanding and adjusting existing privacy settings, Protect_U [50] uses machine learning techniques to recommend privacy settings based on a users personal data and trustworthy friends. Protect_U analyzes user profile contents and ranks them according to four risk levels: Low Risk, Medium Risk, Risky and Critical. The system then suggests personalized recommendations to allow users making their accounts safer. In order to achieve this, it draws upon two protection models: local and community-based. The first model uses the users personal data in order to suggest recommendations, The second model seeks the users trustworthy friends to encourage them to help improve the safety of their counter parts account.

Despite the mole of work on social trust, Social Market [51] is the first system to propose the use of trust relationships to build a decentralized interest-based marketplace. Similarly, TAPE [52] is the first attempt to combine explicit and implicit social networks into a single gossip protocol. Zeng et al. [52] approaches the privacy quantification problem from a different angle. First, they consider how likely a friend reveals others personal information, by computing the privacy trust score, which is a widely studied research problem [53]. Furthermore, the proposed work is related to information diffusion in OSNs such as [54]. TAPE framework differs from other work, in considering information diffusion in the context of privacy protection, which requires different sets of features and considerations.

Zeng and Xing [47] studied how individual users can expand their social networks by making trustful friends who will not leak their private information to unknown parties. This work proposes a security risk estimation framework of social networking privacy to calculate the probability of individual privacy leakage through the social graph. The framework is composed of two parts, the calculation of Individual Privacy

Leakage Probability (IPLP) and the Relationship Privacy Leakage Probability (RPLP). Relationship Privacy Leakage Probability considers the factors of relationship strength and interactive behaviors. Two vectors namely privacy protection awareness (PPA) and privacy protection trust (PPT) are proposed in this paper to estimate IPLP.

Ostra [55] utilizes trust relationship to thwart unwanted communication, where the number of a users trust relationships is used to limit the number of unwanted communications he can produce. Ostra utilizes the existing trust relationship among users to charge the senders of unwanted messages and thus block spam. It relies on existing trust networks to connect senders and receivers via chains of a pair-wise trust relationship, they use a pair-wise link-based credit scheme to impose a cost on the originator of the unwanted communication. Unfortunately, the scalability of this system stays uncertain as it employs a per-link credit scheme.

Gundecha et al. [53] propose a feasible approach to the problem of identifying a users vulnerable friends on a social networking site. Vulnerability is somewhat contagious in this context. Their work differs from existing work addressing social networking privacy by introducing a vulnerability-centered approach to a user security on a social networking site. On most social networking sites, privacy-related efforts have been concentrated on protecting individual attributes only. However, users are often vulnerable through community attributes. Unfriending vulnerable friends can help protect users against the security risks.

Hameed [56] proposed LENS, which extends the friend of friend network by adding trusted users from outside of the FoF networks to mitigate spam beyond social circles. Only emails to a recipient that have been vouched by the trusted nodes can be sent into the network. The authors proposed using social networks and trust and reputation systems to combat spam. In contrast, LENS can reject unwanted email traffic during the SMTP time.

SocialEmail [57] considers trust as an integral part of networking rather than working alongside an existing communication system. SocialEmail leverages social network trust paths to rate the messages. The key feature of SocialEmail is that instead of directly connecting the sender and the recipient, messages are routed through existing friendship links. This gives each email recipient control over who can message him/her, In contrast, such social interaction-based methods are not sufficiently effective in dealing with legitimate emails from senders outside of the social network of the receiver.

Social interactions (e.g., the exchange of messages between users) have been suggested as an indicator of interpersonal tie strength [58]. As a consequence, an unsupervised model has been developed to estimate the relationship strength from the interaction activity and the user similarity in the OSN [58]. Although interaction-based methods leverage social relationships for extracting trust, the applications are not designed to be automated in the sense that the user must explicitly score other users, score messages, create whitelists or adjust the credits.

Vidyalakshmi et al. [48] proposed a privacy control framework for information dispersal on social network, they use the quadratic form of bezier curve to arrive at privacy scores for friends, they use the communication information for pre-sorting of friends which is lacking in [59]. Similarly, Akcora et al. [60] develop a graph-based approach and a risk model to learn risk labels of strangers, the intuition of such an approach is that risky strangers are more likely to violate privacy constraints.

Privacy Index (PIDX) proposed in [61] is a measure of a users privacy exposure in a social network. PIDX is a numerical value between 0 and 100 with a high value indicating high privacy risk in social networks. An attributes privacy impact factor is a ratio of its privacy impact to full privacy disclosure. Thus, an attributes privacy impact has a value between 0 and 1. They consider the privacy impact factor for full privacy disclosure is 1.

Fang and Le Fevre [54] proposed a Privacy Wizard to help users grant privileges to their friends. the goal of this tool is to automatically configure a users privacy settings with minimal effort and interaction from the user. The wizard asks users to first assign privacy labels to selected friends, and then uses this as input to construct a classifier which classifies friends based on their profiles and automatically assign privacy labels to the unlabeled friends. In a similar way, some studies [62] propose a methodology for quantifying the risk posed by a users privacy settings. A risk score reveals to the user how far her privacy settings are from those of other users. It provides feedback regarding the state of her existing settings. However, it does not help the user refine her settings in order to achieve a more acceptable configuration.

Abdul-Rahman and Hailes The trust model presented by Abdul-Rahman and Hailes [63] is focused on virtual communities related to e-commerce and artificial autonomous agents. The model defines direct trust and recommender trust. Direct trust is the trust of an entity in another one based on direct experience. Whereas recommender trust is the trust of an entity in the ability to provide good recommendations. Trust can only have discrete labeled values, namely Very Trustworthy, Trustworthy, Untrustworthy, and, Very Untrustworthy for direct trust, and Very good, good, bad and, very bad for recommender trust. The difference between the two ratings from different entities can be computed as semantic distance. This semantic distance can be used to adjust further recommendations. The combination of ratings is done as a weighted sum, where the weights depend on the recommender trust.

All previous work didn't take into consideration the topological aspect of interactions to measure social vulnerabilities of users. The closest study to our approach is that presented in [47]. However, this solution doesn't study the social interaction environment of users and the potential information leakage through a vulnerable social environment. In this paper, we study the impact of trusting vulnerable users in preserving the privacy of all users in the communication network.

D.3 Related work

To evaluate the privacy risk of social network users, trust metrics are used to measure the extent to which users can be trusted. Trust metrics can be classified into two main categories: global and local trust metrics. Local trust metrics compute trust values that are dependent on the target user. They take into account the very personal and subjective views of the users. They predict different values of trust for every single user based on their own experience. Global trust metrics, on the other hand, predict a global reputation value for each node, based on both the experience of all other users and the topology of the social network.

Much work has focused on adapting existing privacy settings to the users profile, for example, the machine learning techniques used in Protect_U [50] allow recommending privacy settings based on trustworthy friends. Protect_U analyzes the existing privacy settings and ranks them under four risk levels: Low Risk, Medium Risk, Risky and Critical. The system then suggests personalized recommendations to allow users to make their accounts safer. In order to achieve this, it draws upon two protection models: local and community-based. The first model uses the visibility of users profile information to suggest recommendations. The second model searches trustworthy friends to encourage them to help improve the safety of their friends account.

Despite a large amount of work on social trust, a decentralized interest-based marketplace is built for the first time in SocialMarket [51], authors of this framework propose the use of trust relationships to build their decentralized interest-based marketplace. In contrast, TAPE framework developed by Zeng et al. [52] tried to solve the privacy quantification problem [53] from a different angle. First, they calculate the privacy trust score of each user to know how likely a friend could reveal or preserve others personal information. Next, they propose an information diffusion process [54]. The most important contribution made by TAPE framework [52] is in considering information diffusion to reveal privacy leakage in communication.

Zeng and Xing [47] studied the maximization of users relationships by making trustful friends without leaking their private information to unwanted parties. The authors propose a security risk estimation framework to deal with such a problem. The framework proposed is composed of two parts, the calculation of Individual Privacy Leakage Probability (IPLP) and the Relationship Privacy Leakage Probability (RPLP). Two vectors namely privacy protection awareness (PPA) and privacy protection trust (PPT) are proposed in this paper to estimate IPLP.

Gundecha et al. [53] propose an advantageous approach to the problem of identifying a users vulnerable friends. The approach proposed differs from existing work by integrating a vulnerability-centered approach. On most online social networks (OSN), a mole of work addressed the problem of protecting user's individual attributes only, however, few works address the problem of protecting users relationships from vulnerable friends.

Another example that consolidates our intention to build our privacy awareness framework is called LENS [56]. Hameed [56] proposed a novel spam protection system that maximizes the number of trusted nodes who send only trusted emails, only emails to a node that have been allowed by these trusted nodes can be sent through the network. The authors proposed using trust and reputation systems to detect whether a user is trustful or not.

SocialEmail [57] is a trust by design communication system, it considers trust as an integral part of the communication system. SocialEmail rank trust paths to rate the messages, these last ones are routed through existing friendship links that are weighted with different trust level. This gives each email recipient an overview of the trustworthiness of path taken by a message to reach him. In contrast, such methods are not sufficiently effective in dealing with legitimate emails from senders outside SocialEmail.

Social interactions allow users exchanging messages with other users easily and quickly. xiang et al. [58] proposed a social interaction as an indicator of interpersonal tie strength. As a consequence, an unsupervised model has been developed to estimate the relationship strength from their interaction activity [58]. Such methods could extract the level of trust between interlocutors based on the relationship strength between them. However, this application is not designed to be automated, users must manually score other users, score messages or create whitelists.

Vidyalakshmi et al. [48] proposed a privacy control framework for information dispersal on social networks, they use the quadratic form of bezier curve to arrive at privacy scores for friends, they use the communication information for pre-sorting friends which is lacking in [59]. Similarly, Akcora et al. [60] develop a graph-based approach and a risk model to learn risk labels of strangers, the intuition of such an approach is that risky strangers are more likely to violate privacy constraints.

Privacy Index (PIDX) proposed in [61] is a measure of a users privacy exposure in a social network. PIDX is a numerical value between 0 and 100 with a high value indicating high privacy risk in social networks. An attributes privacy impact factor is a ratio of its privacy impact to full privacy disclosure. Thus, an attributes privacy impact has a value between 0 and 1. They consider the privacy impact factor for full privacy disclosure is 1.

Fang and Le Fevre [54] proposed a Privacy Wizard to help users grant privileges to their friends. It automatically configures users' privacy settings. The wizard asks users to first assign trust labels to selected friends. The wizard is then training to classify tagged friends based on both profile information and labels assigned by users themselves. Ones this step is finished, the wizard is now able to assign new labels to unlabeled friends based only on their profile information. In a similar way, some studies [62] propose a methodology to quantify the vulnerability of users privacy settings. A risk score reveals to users how far their privacy settings are from those of other users. However, it does not help users refine their settings in order to achieve a more acceptable configuration.

Abdul-Rahman and Hailes [63] proposed a trust model with virtual communities and artificial autonomous agents. The model defines a direct trust and a recommender trust. Trust can only have discrete labeled values, namely Very Trustworthy, Trustworthy, Untrustworthy, and, Very Untrustworthy for direct trust, and Very good, good, bad and, very bad for recommender trust. The difference between the two ratings from different entities can be computed as a semantic distance that can be used to adjust further recommendations. The combination of ratings is done as a weighted sum, where the weights depend on the recommender trust.

All previous work didn't take into consideration the topological aspect of interactions to measure social vulnerabilities of users. The closest study to our approach is that presented in [47]. However, this solution doesn't study the impact of having interactions with vulnerable users. In this paper, we study the impact of trusting vulnerable users in preserving the privacy of all users in the communication network.

D.4 Approach

The aim of this study is to understand how trust coefficient affects social privacy vulnerability. In this section, we present our vulnerability contagion process in messaging services in order to get social vulnerability scores of users. These scores represent the vulnerability within the social environment of each user. In our study, we focus on the impact of trust and reputation in spreading out the privacy vulnerability.

To model this process we measure the impact of vulnerable users in protecting friends privacy. For example, let us say that a user is exchanging messages with five friends as shown in Figure 1, the reputation of this user is given as the probability to be trusted by his friends (Figure 1a). The more trustworthy a user is, the more reputed he becomes. However, if a user with a high reputation level has a high vulnerability, he can spread his vulnerability with a high infection coefficient (Figure 1b). As a consequence, the social vulnerability of users is calculated as the level of the contagion degree of each user in the communication network. To get these values, a weighted matrix value M is used as an adjacency matrix normalized by users' degree. Social vulnerability is calculated in a continuous space, this means that we iterate the vulnerability contagion process until the process converges.

The probability of the infection is based on the trust level between users.

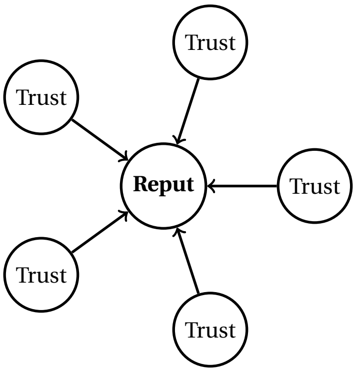
To evaluate the impact of trust in this process we add a reputation parameter $p_{reputation}$, this parameter is used to know how likely a user could be trusted by his friends, it is calculated as the probability to get at least one trust grant from them.

$$p_{reputation} = p(\mathrm{X} \geq 1) \; = \; 1 \; - \; (1 \; - \; p_{trust})^n \tag{D.1}$$

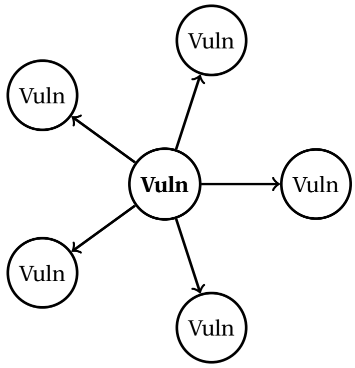
Where,

- X: Number of trust grant from friends, $\mathrm{X} \sim \mathrm{B}(n, p)$.
- n = number of user's friends.
- $p_{trust} = p(\mathrm{X} = 1)$: Probability to get one trust grant from a friend.

Trust parameter, in this equation, is added as a coefficient parameter to increase or decrease the vulnerability contagion process. Whether a user can infect other users social vulnerability scores depends on the



(a) Trust grant.



(b) Vulnerability contagion process.

Figure 1: Reputation coefficient.

trust level between them, so users should distrust vulnerable users to preserve their own privacy and the privacy of the entire communication network.

The number of friends infected by each user u in each step of the vulnerability contagion process is given as:

$$new\ infected = old\ infected + p_{reputation} * degree(u)$$

Initial privacy vulnerability scores of each user are given as input to our algorithm to estimate social privacy vulnerability scores, a normal distribution is used to generate initial privacy vulnerability scores. The vulnerability contagion equation is given as:

$$\mathrm{P}_{i+1} = p_{reputation} * (\mathrm{M} * \mathrm{P}_i) + (1 - p_{reputation}) * \mathrm{P}_i \tag{D.2}$$

Where,

- P_0 is the initial individual privacy vulnerability scores of each user.
- M is the adjacency matrix normalized by users' degree.
- i is the diffusion process iteration level.

The first part of this equation computes the vulnerability of a user based on his friends' average vulnerabilities weighted by the trust level with them. The second part computes the vulnerability of a user based only on his own vulnerability weighted by his friends' distrust level. Social vulnerability is a function of friends vulnerability and the trust coefficient. Mean distances between privacy vulnerability scores of each iteration is calculated to get the convergence process.

D.5 Experimentation

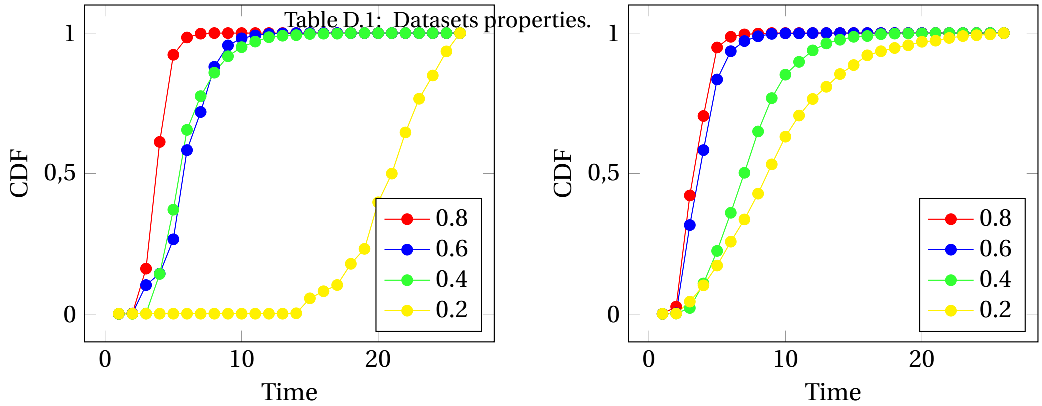
To evaluate our contagion process, we used Enron dataset in our experimentation. There are many reasons for using Enron dataset to evaluate our vulnerability measure techniques. First of all, it is probably the only actual corporate messaging service dataset available to the public. Second, it contains all kind of emails "personal and official", so email logs can reveal the level of trust between users by studying the information flow in the network. Finally, this dataset is similar to the kind of data collected for fraud detection or counter-terrorism, hence, it is a perfect test bed for testing the effectiveness of new vulnerability measure techniques.

The properties of the Enron dataset used for our experimentation are presented in table D.1.

Due to visibility issues, we extract only important nodes given in [36]. Our substantive interest in this experimentation is in how vulnerability moves through the network. The inputs of our experimentation consist of a set of individual vulnerabilities of users in the network. This set is generated randomly and is represented in the graph of Figure 4a, values of this set are between 0 and 1 and represent the extent to which users are exposed to different kind of attacks such as (phishing, spam, etc). Unlike social vulnerability values, these values didn't take into account the vulnerability contagion process between users. To evaluate how trust coefficient affects our outputs i.e., social privacy vulnerabilities, we variate the trust coefficient through 4 symmetric values 0.2, 0.4, 0.6 and 0.8.

Parameter	Enron dataset	Caliopen dataset
Nodes	958	5885
Edges	6966	26547
Diameter	958	2096
Mean degree	2.413361	9.02192
Edge density	0.00252	0.001533
Modularity	0.654600	0.86526
Mean distance	3.042114	3.914097

Table D.1: Datasets properties.



(a) Enron dataset.

(b) Caliopen dataset.

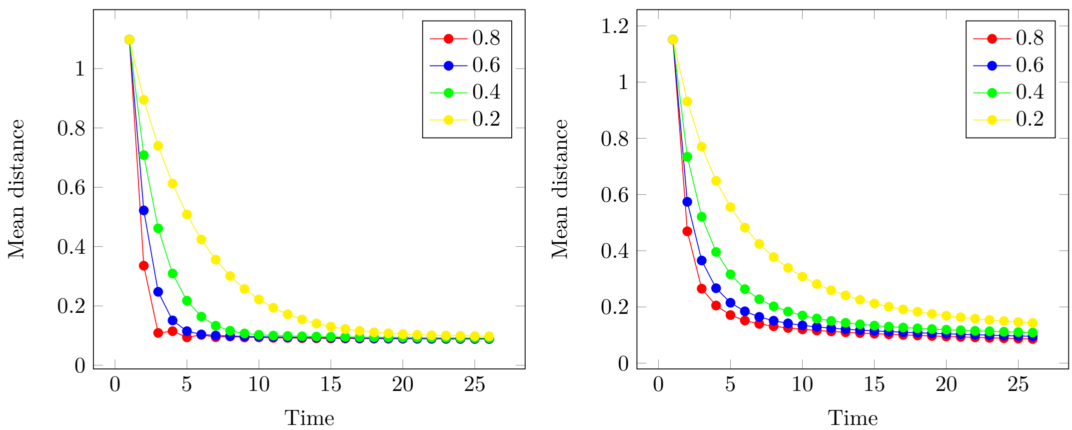
Figure 2: Cumulative distribution function of infected users.

D.6 Results exploitation

Below we report the results of applying the contagion process model to the Enron Email dataset.

Figure 2 shows the cumulative distribution function of the vulnerability contagion process, it appears clearly that the vulnerability diffusion process increases as the reputation level of vulnerable users increases, because when the vulnerability contagion process is at its 7^{th} iteration, the cumulative distribution function with the highest user's reputation level (0.8) is 1, this means that after the 7^{th} iteration, all users are infected. This is not the case of users with low reputation level which need further contagion steps to diffuse their vulnerability widely in the network. As a consequence, we can say that vulnerability contagion process speed is highly correlated with users reputation, it infects a large number of users quickly when the user's reputation level tends to 1.

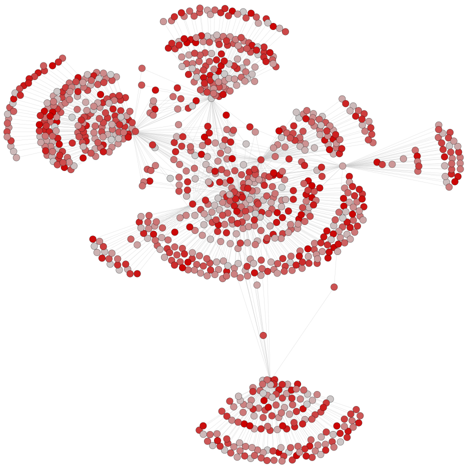
.5



(a) Enron dataset.

(b) Caliopen dataset.

Figure 3: Diffusion process convergence.



(a) Individual privacy vulnerability.

(b) Social privacy vulnerability.

Figure 4: Individual & Social privacy vulnerabilities.

Figure 3 shows the convergence of the vulnerability contagion process, the mean distance between users' social vulnerability scores in each iteration is calculated to see the convergence of the process. The convergence process shows a high convergence level when the reputation coefficient is 0.8. This happens when the mean distance between the users' social vulnerability scores calculated at each iteration still the same. In other words, there are no more users to affect and the contagion process is at its high level. After viewing these tow graphs, we can conclude that over-trusting vulnerable users allow them to get a high reputation level and consequently infects the entire social vulnerability scores.

The initial and final measures of the simulation are represented in Figures 4. Figure 4a shows the initial privacy scores, these scores are generated randomly to illustrate the user's individual privacy vulnerability without caring about their friends' vulnerability. Figure 4b shows the final social privacy scores of the contagion process, these scores reveal the social vulnerability of users. Users with dark color are more vulnerable than others with light color in terms of friendship with other users. As a consequence, they have a high level of social vulnerability scores.

User ID	Individual vulnerability	Social vulnerability
34	0.84	0.67
67	0.12	0.87
206	0.76	0.33
588	0.23	0.78

Table D.2: Difference between Individual & Social privacy vulnerabilities.

Table D.2 illustrates the input (Individual vulnerability) and the output (Social vulnerability) values of four arbitrary users of our dataset. Results show that users with low individual vulnerability (e.g. user 67) could have a high social vulnerability due to their interactions with vulnerable users. In contrast, users with high individual vulnerability (e.g. user 206) can, in turn, be less vulnerable from interactions with other users, but harms considerably the social vulnerability of their interlocutors.

In summary, results presented in this section show that if the trust coefficient between users is up to 0.8, the vulnerability diffusion process through trust relationship is at its high level of speed. This what happens when a new information appears in a communication network and users forward it largely in the network. In addition, this work gives a new insight to understand the relationship between trust, reputation, individual vulnerability and social vulnerability in the context of messaging services such as emails.

D.7 Conclusions

This paper studies the relationship between trust and reputation metrics in users' interactions and the social privacy vulnerability of users. We observe that such metrics, severely affect users privacy in regard to their social relationships. Trust coefficient in the social network graph plays an essential role in spreading vulnerability. Our experiment investigates the probability of infecting other's privacy by increasing the reputation of vulnerable users. In this case, the number of nodes infected depends on the trust grant assigned to vulnerable users. Our experiment reveals also that the social vulnerability of users could be extracted from their individual vulnerability by applying our vulnerability contagion process. Privacy-preserving in online social network architectures should address this problem by discouraging trusting vulnerable friends.

E | Survey

Survey

Contents

Abstract

TODO

E.1 Introduction

TODD aghiles

E.2 Approaches

Quantifying and measuring privacy is very challenging, mainly because the definition of privacy is very subjective, each individual might have a different opinion about this concept. In our work, we present privacy challenges through three points of view: behavioral, social and technical.

E.2.1 Behavioral

We consider two types of behavior models in this work, one based on typical behavior from a user perspective and one on typical behavior of a users' messages. Both of these models can be quantified into a probability score.

E.2.1.1 User behavior

The user's behavior is modeled with respect to their typical service messaging usage, the frequency and type of messages received and sent, and the typical recipients with whom they exchange messages. Known as a behavior profile, this type of model is computed over some training period to learn how the user behaves within the messaging account.

The measurements of the user's message behavior include the frequency of inbound/outbound message traffic, the specific times messages arrive and are sent, the "social cliques" of a user, and the user's response rates when replying to specific senders.

Vidyalakshmi et al. [59] proposed a privacy scoring using bezier curve. They present a framework for calculating a privacy score metric considering **users personal attitude towards privacy and communication information**. They focus on the rating of the users OSN friends based on their attitudes towards privacy, helping him to make an informed decision of sharing information with them. Bezier curve in its cubic form is used as it has to account for both privacy orientation and communication orientation of the user.

[42]

[64]

[**liu_framework_2010**] propose a model to compute a privacy score of a user. The privacy score increases based on how sensitive and visible a profile item is and can be used to adjust the privacy settings of friends. Their solution also focused on the privacy settings of users with respect to their profile items. They use Item Response Theory (IRT) to evaluate **sensitivity and visibility of attributes** when evaluating privacy scores. The authors definition of privacy score satisfies the following intuitive properties: the more sensitive information a user discloses, the higher his or her privacy risk. However, their approach do not support personalized privacy view over profile content for each individual in the social network.

E.2.1.2 Message behavior

The second type of behavior is specific to how spam behaves and how it appears in the message folder among normal messages. In general, spam messages can be easily detected because they appear anomalous with respect to the normal set of messages received and opened by the user. However, ... TODO

E.2.2 Social

In online social networks, users are sometimes either oblivious about their privacy, or concerned but underestimate the privacy risks. OSN service providers allow users to manage who can access which information and communication (e.g. Facebook and Google+). Researcher studied privacy protection from two directions:

Along the first direction, fundamental changes to the current design of OSN were suggested to enhance users' privacy. Within this direction, Privacy by Design (PbD) is an important approach. For example, in [65], Baden et al. proposed a new type of OSNs by using **attribute-based encryption** to hide user data, in which symmetric keys are used to encrypt messages and only the designated friend groups can decrypt the messages. In [66], Erkin et al. proposed to use homomorphic encryption and **multi-party encryption techniques** to hide privacy-sensitive data from the service provider in a recommender system.

The second direction is developing privacy protection tools based on existing OSNs. In our work, we focus on both direction to deal with current and future messaging services.

In [67], the authors propose to use the amount of information that can be inferred from social networks to quantify the privacy risks. PrivAware detect and report unintended information disclosures through quantifying privacy risk associated with friend relationship in OSNs. PrivAware employs inference model which is based on the fact that information about users can be inferred from their social graph. Privacy score is calculated as total **number of attributes visible to the third party applications** divided by total number of attributes per participant. The measured percentage is then mapped to a letter grade, where A score represents very few attributes being revealed and F score indicates that privacy risk to the threat of a malicious third party application is high.

The authors in [68] develop a tool, Privometer, to measure information leakage based on user profiles and their social graph. The leakage is indicated by a probability numerical value. Privometer is based on an augmented inference model where a potentially malicious application installed in the users friend profiles can access substantially more information. It operates in two modes. In online mode, inference is performed based on the friends profile where most frequently value is selected. In offline mode, it uses only immediate friends and "network-only Bayes classifier" to measure the **probability of inference**. The tool can suggest self sanitization actions based on the numerical value.

[48] proposed a privacy control framework for information dispersal on social network, they use the quadratic form of bezier curve to arrive at privacy scores for friends, they use the **communication information** for pre-sorting of friends which is lacking in [59].

Privacy Index (PIDX) proposed in [61] is a measure of a users privacy exposure in a social network. PIDX is a numerical value between 0 and 100 with high value indicating high privacy risk in social networks. An **attributes privacy impact factor** is a ratio of its privacy impact to full privacy disclosure. Thus, an attributes privacy impact has a value between 0 and 1. They consider privacy impact factor for full privacy disclosure is 1.

[60] develop a graph-based approach and a risk model to learn **risk labels of strangers**, the intuition of such an approach is that risky strangers are more likely to violate privacy constraints.

Fang and Le Fevre [54] proposed a Privacy Wizard to help users grant privileges to their friends. the goal of this tool is to automatically configure a users privacy settings with minimal effort and interaction from the user. The wizard asks users to first assign privacy **labels to selected friends**, and then uses this as input to construct a classifier which classifies friends based on their profiles and automatically assign privacy labels to the unlabeled friends.

In a similar vein, some studies [62] propose a methodology for quantifying the risk posed by a users privacy settings. A risk score reveals to the user **how far his/her privacy settings are from those of other users**. It provides feedback regarding the state of his/her existing settings. However, it does not help the user refine his/her settings in order to achieve a more acceptable configuration.

Trust metrics can be classified to two main categories: global and local trust metrics. **Local trust metrics**, compute trust values that are dependent on the target user, Local trust metrics take into account the very personal and subjective views of the users, they predict different values of trust for every single user based on their own experience. **Global trust metrics (reputation)**, on the other hand, predict a global reputation value for each node.

E.2.2.1 Concept of trust

In trust networks users can ask to rate other users, this means that, a user can express her level of trust in another user she has interacted with, i.e. express a trust statement such as "Alice, trust Bob as 0.8 in [0,1]". The system can then aggregate all the trust statements in a single trust networks representing the relationships between users. Trust metrics are algorithms whose goal is to predict, based on the trust network, the trustworthiness of "unknown" users, i.e. users in which a certain user didnt express a trust statement. Their aim is to reduce social complexity by suggesting how much an unknown user is trustworthy. Due to the increased use of OSNs, there is a growing number of studies that focus on using social network data for scoring messages in order to filter unwanted messages in messaging systems. The difference between each study has to do with the way the concept of trust is represented, computed and used.

The concept of trust is used to indicate the relationship between two entities. Trust in an entity is a commitment to an action based on a belief that the future actions of that entity will lead to a good outcome. There are three main properties of trust that are relevant to the development of algorithms for computing it [**wang_trustinvolved_2010**], namely, transitivity, asymmetry, and personalization. The primary property of trust that is used in our work is transitivity. if Alice highly trusts Bob, and Bob highly trusts Chuck, it does not always and exactly follow that Alice will highly trust Chuck. It is also important to note the asymmetry of trust, for two people involved in a relationship, trust is not necessarily identical in both directions. The third property of trust that is important in social networks is the personalization of trust, trust is inherently a personal opinion, two people often have very different opinions about the trustworthiness of the same person.

While much work has focused on tools for understanding and adjusting existing privacy settings, **Protect_U** [50] uses machine learning techniques to recommend privacy settings based on a users personal data and trustworthy friends. Protect_U analyzes user profile contents and ranks them according to four risk levels: Low Risk, Medium Risk, Risky and Critical. The system then suggests personalized recommendations to allow users to make their accounts safer. In order to achieve this, it draws upon two protection models: local and community-based. The first model uses the **users personal data** in order to suggest recommendations, The second model seeks the **users trustworthy friends** to encourage them to help improve the safety of their counter parts account.

Despite the mole of work on social trust, Social Market is the first system to propose the use of **trust relationships** to build a decentralized interest-based marketplace.

Similarly, TAPE [52] is the first attempt to combine explicit and implicit social networks into a single gossip protocol. Zeng et al. [52] approaches the privacy quantification problem from a different angle. First, they consider **how likely a friend reveals others personal information**, by computing the privacy trust score, which is a widely studied research problem [53]. Furthermore, the proposed work is related to **information diffusion in OSNs** such as [54]. Finally, TAPE framework differs from other work, in considering information diffusion in the context of privacy protection, which requires different sets of features and considerations.

Ostra [55] utilizes trust relationship to thwart unwanted communication, where the number of a users trust relationships is used to limit the amount of unwanted communications he can produce. Ostra utilizes the existing trust relationship among users to charge the senders of unwanted messages and thus block spam. It relies on existing trust networks to connect senders and receivers via **chains of pair-wise trust relationship**, they use a pair-wise link-based credit scheme to impose a cost on originator of unwanted communication. Unfortunatly, the scalability of this system stays uncertain as it employs a per-link credit scheme.

Gundecha et al. [53] propose a feasible approach to the problem of identifying a users vulnerable friends on a social networking site. Vulnerability is somewhat contagious in this context. Their work differs from existing work addressing social networking privacy by introducing a **vulnerability-centered approach to a user** security on a social networking site. On most social networking sites, privacy related efforts have been concentrated on protecting individual attributes only. However, users are often vulnerable through community attributes. Unfriending vulnerable friends can help protect users against the security risks.

In [47], Sun et al proposed a **probability trust model** that uses Beta function to address concatenation propagation and multi-path propagation of trust.

SOAP [69] presents a social network based personalized spam filter that integrates **social closeness**, **user (dis)interest** and adaptive **trust** management into a Bayesian filter. SOAP proposed an email scoring mechanism based on an email network augmented with reputation ratings. An email is considered spam if the **reputation score of the email sender** is very low. Different from these social network based methods, SOAP focuses on personal interests in conjunction with social relationship closeness for spam detection. However, several issues with SOAP, including the intrinsic cost of initialization and continuous adaptation of social closeness (between sender and recipient), and social interests (of an individual) in the Bayesian filter, limit its usage.

Relationship between **users trustworthiness** and privacy risk is presented in [70].

Hameed [56] proposed LENS, which extends the FoF network by adding trusted users from outside of the FoF networks to mitigate spam beyond social circles. Only emails to a recipient that have been **vouched by the trusted nodes** can be sent into the network. The authors proposed using social networks and trust and reputation systems to combat spam. In contrast, LENS can reject unwanted email traffic during the SMTP time.

SocialEmail [57] considers the trust as an integral part of networking rather than working alongside of an existing communication system. SocialEmail leverages **social network trust paths** to rate the messages. The key feature of SocialEmail is that instead of directly connecting the sender and the recipient, messages are routed through existing friendship links. This gives each email recipient control over who can message him/her, In contrast, such social interaction-based methods are not sufficiently effective in dealing with legitimate emails from senders outside of the social network of the receiver.

Social interactions (e.g., **the exchange of messages between users**) have been suggested as an indicator of interpersonal tie strength [58]. As a consequence, an unsupervised model has been developed to estimate the **relationship strength** from the interaction activity and the user similarity in the OSN [58]. Although interaction-based methods leverage **social relationships** for extracting trust, the applications are not designed to be automated in the sense that the user must explicitly score other users, score messages, create whitelists or adjust the credits.

Fong [71] formulated this paradigm called a Relationship-Based Access Control (ReBAC) model, it bases authorization decisions on the **relationships between the resource owner and the resource accessor** in an OSN. However, most of these existing work could not model and analyze access control requirements with respect to collaborative authorization management of shared data in OSNs.

E.2.2.2 Concept of reputation

Trust and reputation concepts are used in order to preserve users privacy while increasing their social capital in OSNs. Reputation concept is used to refer to a more general sense of trust towards a particular entity based on opinions of multiple entities.

A reputation system collects, distributes and aggregates feedback about participants past behavior. Such systems help people decide whom to trust, encourage trustworthy behavior and deter participation by those who are unskilled or dishonest. Various applications use real-time reputation-based systems, including online markets and anti-spam solutions. Anti-spam reputation systems generate a score, or rating, for each incoming message or IP, based on analysis of various parameters: message volume, type of traffic (e.g. sporadic vs continuous), rate of user complaint reports, feedback from spam traps, compliance with regulations, etc. This aggregated information, collected over time, forms the reputation of the sender.

SNARE [72] infers the reputation of a message sender based on **network-level features**, (e.g. **geodesic distance between sender and recipient, number of recipients**). The most influential feature in the system was the AS number of the sender. Using an automated reputation engine, SNARE classifies message senders as spammers or legitimate with about a 70% detection rate for less than a 0.3% false positive rate, without looking at the contents of a message. However, lacking authentication and non-repudiation in standard trust and reputation solution make these solutions be subject to identity spoofing, false accusation and collusion attacks. Further, these solutions consume extra valuable resources of messaging servers on message reception and filtering.

In TrustMail, which is a prototype E-mail client, an approach is proposed that makes use of OSN reputation ratings to attribute different scores to E-mails [73]. The actual benefit of this system is that, by using **social network data**, it identifies potentially important and relevant messages even if the recipient does not know the sender [73].

Qian et al. [74] addressed this issue by presenting a clustering technique that refines **AS-based and BGP prefix-based clusters**. The authors combined **BGP and DNS information** to identify a cluster of IP addresses within the same administrative boundary, and thus constructed the reputation for an entire cluster. This cluster-based reputation system allowed more accurate identification of the reputation of previously unknown IP addresses, and reduced the false negative rate by 50 percent compared to blacklists, without increasing the false-positive rate [74].

Paradesi et al. [75] adopted a multi-agent based reputation model to define **trustworthiness of services**. Moreover, they developed a trust framework to derive trust for a composite service from trust model of component services.

E.2.2.3 Collaborative management

The trust value assigned to a person in previous work is estimated on the basis of his/her reputation, which can be assessed taking into account the person behaviour. Indeed, it is a matter of fact that people assign to a person with unfair behaviour a bad reputation and, as a consequence, a low level of trust. A possible solution is to estimate the trust level to be assigned to a user in a collaborative community on the basis of his/her reputation, given by his/her behavior with regards to all the other users in the community.

Hu et al. [76] propose an approach to enable **collaborative privacy management** of shared data in OSNs. In particular, they provide a systematic mechanism to identify and resolve privacy conflicts for collaborative data sharing. their conflict resolution indicates a trade-off between privacy protection and data sharing by quantifying privacy risk and sharing loss

Dealing with **collaborative information sharing**, Hu et al. [77] proposed a method to detect and resolve privacy conflicts.

The collaborative systems, called **COAT** [78], do not rely upon semantic analysis but on the community to identify spam messages. Once a message is tagged as spam by one SMTP server, the signature of that message is transmitted to all other SMTP servers. This class requires the collaboration of multiple SMTP servers to implement the system.

SocialFilter [79] proposes a collaborative spam mitigation system that uses social trust embedded in OSN to asses the trustworthiness of Spam reporter. The spammer reports from the SocialFilter nodes are stored at a centralized repository that computes the trust values of the reports and identifies spammers based on IP addresses. However, the SocialFilters effectiveness is doubtful as spammers may use dynamic IPs.

Squicciarini et al. [76] proposed a solution for collective privacy management for photo sharing in OSNs. This work considered the privacy control of a content that is co-owned by multiple users in an OSN, such that each co-owner may separately specify her/his own privacy preference for the shared content. The Clarke-Tax mechanism was adopted to enable the collective enforcement for shared content. Game theory was applied to evaluate the scheme. However, a general drawback of this solution is the usability issue, as it could be very hard for ordinary OSN users to comprehend the Clarke-Tax mechanism and specify appropriate bid values for auctions. In addition, the auction process adopted in their approach indicates only the winning bids could determine who was able to access the data, instead of accommodating all stakeholders privacy preferences.

E.2.3 Technical

Nowadays, the Web converged over two main protocols, namely HTTP/HTTPS and SMTP/SMTPS Beside these, DNS still has a central role for reaching almost any Web server. In this section we focus on SMTP privacy, the next section relies on HTTP privacy.

Many anti-spam techniques have been proposed and deployed to counter email Spam from different perspectives. Based on the placement of anti-Spam mechanisms, these techniques can be divided into two main categories: recipient-oriented and sender-oriented.

E.2.3.1 Recipient-oriented Techniques

This class of techniques either (1) block/delay email Spam from reaching the recipients mailbox or (2) remove/mark Spam in the recipients mailbox. Due to the flourish of techniques in this category, we further divide them into content-based and non-content-based sub-categories.

1) Content-based Techniques The techniques in this sub-category detect and filter spam by analyzing the content of received messages, including both message header and message body.

Email address filters: Email address filters are simply whitelists or blacklists. Whitelists consist of all acceptable email addresses and blacklists are the opposite. Blacklists can be easily broken when spammers forge new email addresses, but using whitelists alone makes the world enclosed.

Heuristic filters: The features that are rare in normal messages but appear frequently in spam, such as nonexisting domain names and spam-related keywords, can be used to distinguish spam from normal email.

Machine learning based filters: Since spam detection can be converted into the problem of text classification, many content-based filters utilize machine-learning algorithms for filtering spam. As these filters can adapt their classification engines with the change of message content, they outperform heuristic filters.

2) Non-content-based Techniques The techniques in this sub-category use non-content spam characteristics, such as source IP address, message sending rate, and violation of SMTP standards, to detect email spam.

DNSBLs: DNSBLs are distributed blacklists, which record IP addresses of spam sources and are accessed via DNS queries. When an SMTP connection is being established, the receiving MTA (Mail Transfer Agent) can verify the sending machines IP address by querying the subscribed DNSBL. Mail server records the number and frequency of the same email sent to multiple destinations from specific IP addresses. If the number and frequency exceed thresholds, the node with the specific IP address is blocked. Even DNSBLs have been widely used, their effectiveness and responsiveness [**jung_empirical_2004**, **ramachandran_can_2006**] are still under study.

MARID: MARID (MTA Authorization Records In DNS) is a class of techniques to counter forged email addresses by enforcing sender authentication. MARID is also based on DNS and can be seen as a distributed whitelist of authorized MTAs. Multiple MARID drafts have been proposed, some of them (SPF and DKIM) are deployed in real world [**spf:_2018**, **BibEntry2014Dec**]. PGP and S/MIME are also

Challenge-Response (CR): CR is used to keep the merit of whitelist without losing important messages. To add a sender email address in the whitelist, senders are requested a challenge that needs to be solved by a human being. After a proper response is received, the senders address can be added into the whitelist.

Cryptographic: Pretty Good Privacy (PGP) [**pgpaghiles2007**] and S/MIME are both cryptographic approaches that sign the message body using public-key cryptography and append the signature in the body. In PGP, Keys are stored in end-user key rings or in public key-servers. Key management uses a peer-to-peer web of trust architecture. Whereas in S/MIME, management follows a hierarchical model similar to SSL and keys are signed by a certificate authority.

Delaying: As a variation of rate limiting, delaying is triggered by an unusually high sending rate. Most delaying mechanisms are applied at receiving MTAs.

E.2.3.2 Sender-oriented techniques

To effectively deny spam at the source, ISPs and ESPs (Email Service Providers) have taken various measures to manage the usage of email services. For example, message submission protocol [**BibEntry1998Dec**] has been proposed to replace SMTP, when a message is submitted from an MUA (Mail User Agent) to its MTA.

The proposed work in [78] differs from the other techniques in a way that all of them categorize mail messages at receiver side, whereas COAT works at the sender side and reduces outgoing spam rather than inbox spam.

Cost-based approaches: Borrowing the idea of postage from regular mail systems, many cost-based techniques attempt to shift the cost of thwarting spam from receiver side to sender side.

All these techniques assume that the average email cost for a normal user is trivial and negligible, but the accumulative charge for a spammer will be high enough to drive them out of business.

Cost concept may have different forms in different proposals. Bonded Sender [**BibEntry2018May**] advocates associating email with real money.

E.2.4 Technical HTTP

In this section we enumerate existing privacy protection measures available to users and one new protection proposal [**mayer-do-not-track-00**].

Blocking requests to targeted third parties: This block measure includes using an advertisement blocking tool (AdBlock Plus [**adblockplus**]) to syntactically block selected third parties via server/domain name. Another measure blockhidden [**krishnamurthy_privacy_2009**] determines the true source of hidden third-parties by examining their authoritative DNS servers.

Refusing cookies to prevent tracking: Browsers can be set to refuse all cookies (nocook) or just third-party cookies (no3rdcook).

Disabling script execution: JavaScript execution can be disabled (nojs) either permanently via the browser or selectively via a tool such as NoScript [**NoScript**].

Filtering protocol headers: This is done via extensions or at an intermediary and includes the referrer measure available in some browsers to modify or remove the Referer header in an HTTP request.

Anonymizing the user and user actions: One such anon measure is anonymizing users IP address via an anonymizing proxy or by using Tor.

Do-Not-Track HTTP header proposal: Researchers proposed, in early 2010, that browsers add a HTTP DoNot-Track-Header (DNT-Header) [**mayer-do-not-track-00**] to allow users to express their interest in not being tracked by any aggregator or ad network. However, the extent to which third parties would honor such a header is unknown.

E.3 Background

The study of privacy measurement applied to information and communication technology (ICT) is very wide and embraces many fields of knowledge from Sociology and Statistics to Cryptography and Artificial Intelligence. To better understand the privacy issues in ICT, we describe different types of privacy threats used by attackers to access users' private information through messaging services. Next, we present four options available to the end users to deal with ICT privacy issues.

E.3.1 Privacy threats

Four types of privacy threats have been discovered in the literature:

E.3.1.1 Private information disclosure

When users share information with trusted social network community. they implicitly assume that their information shared through messages would stay within the community destination. However, this assumption is not always valid, an individual messages may be accessed by adversaries. For instance, messages sent to an email-based social network may be stored at a repository and consequently visible to the public, malicious users and applications may follow people through social networks, add-ons and third parties may access users private information, etc.

[**krishnamurthy_leakage_2010**] shows that online social networks and applications leak users personally identifiable information to third parties. The results of their study clearly show that the indirect leakage of PII via OSN identifiers to third-party aggregation servers is happening. OSNs in our study consistently demonstrate leakage of user identifier information to one or more third-parties via Request-URIs, Referer headers and cookies. In addition, two of the OSNs directly leak pieces of PII to third parties with one of the OSNs leaking zip code and email information about users that may not be even publicly available within the OSN itself.

E.3.1.2 Information aggregation

When users authenticate to their favorite service messaging; generally, online social networks, they voluntarily release different types of personal information: name, screen name, telephone numbers, email addresses, locations, etc. Moreover, when users post messages in forums, blogs and webmails, they also disclose small pieces of private information. However, with the development of information retrieval techniques, private information of the same user may be collected from different sources and aggregated to reveal user privacy [80].

E.3.1.3 Inference attacks

Aside from voluntary disclosure of explicit personal information, users information could be inferred from public information items. The privacy literature recognizes two types of private information leakage: identity leakage and attribute leakage, and identity leakage often leads to attribute leakage. Identity disclosure occurs when the adversary is able to determine the mapping from a record to a specific real-world entity (e.g. an individual). Attribute disclosure occurs when an adversary is able to determine the value of a user attribute that the user intended to stay private.

E.3.1.4 Re-identification and De-anonymization attacks

When social network data sets are published for various legitimate reasons, user identity and some profile information are often removed to protect the user privacy. The privacy literature recognizes two types of privacy mechanisms: interactive and non-interactive. In the interactive mechanism, an adversary poses queries to a data-base and the database provider gives noisy answers. In the non-interactive setting, a data provider releases an anonymized version of the database to meet privacy concerns. Some of the well-known techniques for this purpose includes k-anonymity, l-diversity and t-closeness [**li_tcloseness_2017**]. For instance, in a k-anonymized data set, an individual cannot be distinguished by attributes from other k-1 records. However, possibilities of **attribute re-identification attacks** on publicly available data sets have been studied in [81]. They show that user identities could be recovered from anonymized data sets.

On the other hand, due to the nature of social network data, just anonymizing node attributes is not enough. Graph structure contains significant amount of information which could be utilized to hurt user privacy, i.e. **structural reidentification attacks**. A good survey on structural anonymization and re-identification attacks could be found at [**zhou_brief_2008**]. most works show that node identities could be inferred through graph structure.

E.3.2 Privacy solutions

When an ICT user accesses a service, in particular, a messaging service, she has to share some information with the service provider, namely identity, type of service required, location, etc. Clearly, the shared information depends on the service but regardless of the exchanged information, to deal with the existing privacy issues, users have to choose between four main options:

E.3.2.1 Privacy based on trust

This is probably the most common situation. Users tend to trust service providers because, they do not really have alternatives in many cases. Due to the fact that privacy is considered a right, most countries have regulations that oblige companies to guarantee the privacy of their users. Among this regulation, when users data are released to third parties they should be sanitized so as to guarantee users privacy [**chen_privacypreserving_2009**].

E.3.2.2 Privacy based on Individual User Actions

Despite the legislation, users might prefer to keep some of their private information away from the service provider. In this case, we assume that the user cannot collaborate with the service provider. For example, in the case of sending a query to an Internet search engine such as Google or Yahoo. The user cannot initiate a **collaborative protocol** with the search engine, because the search engine is only able to receive and answer queries.

E.3.2.3 Privacy based on Collaboration with the Provider

There are situations in which the service provider might collaborate with the user to protect her privacy by running **privacy-aware protocols.** TODO

E.3.2.4 Privacy based on Collaboration with other Users

This is an evolution of the proposals described in E.3.2.2 in which users collaborate to protect their privacy. In this case, users do not want to trust the provider nor other third parties. TODO

E.4 Conclusions

TODO

F | Appendix A

Appendix A

Year		Factors	Computation Model	Results interpretation
2018	[42]	-Closeness Centrality -Degree Centrality	Estimation	**Closeness** have a high degree of correlation with **privacy score**
2017	[64]	-Users Credibility	Privacy-preserving	Correlation between identifying **malicious users** and **user's credibility** is significantly higher
2016	[82]	-Influence probability -Degree Centrality -Number of seeds neighbors -Number of experts neighbors	Utility degree and Utility privacy cost ratio discount algorithms	Methods evaluations: -Number of seeds activated (False positive) -Number of expert activated (True Positive)
2015	[70]	-Attitude Information -Popularity (Page rank) -Privacy Settings: visibility	Users trustworthiness	Relationship between **users trustworthiness** and **privacy risk**
2015	[52]	-Node information diffusion -Link information diffusion -Undesirable Destination	Birnbaum's measure (BM)	**Users trustworthiness**
2015	[48]	-Tie-strength -Communication information -Number of mutual friends	Bezier curve	Sorting user's **Friends Privacy**
2015	[59]	-Communication frequency -Privacy setup	Cubic bezier curve	Estimating friends privacy scores with various **ego users dispositions** to privacy and communication
2014	[83]		3-class supervised learning	Correlation between **Users Privacy** and Users **Friends Privacy** Score
2014	[47]	-Undesirable Destination -Closeness centrality -Diffusion centrality -User's behavior	Probability trust model	-The **information diffusion** through nodes mainly depends on the **users behavior** -The **information diffusion** through links mainly depends on the **closeness** between friends
2013	[61]	-Sensitivity, Visibility	Linear model	**Users privacy** exposure
2013	[77]	-User valuation on app -Network valuation on app	Collaborative Interdependent Privacy Game, Nash equilibrium Sub-optimal Equilibrium	Show how network and personal effects can shape the behavior of social network users with regard to app usage
2012	[60]	-Attitude similarity	Baseline estimation Learning Friend Impacts	Risks of friendships can be mined by analyzing users attitude towards friends of friends
2012	[78]	-Sender address -Sender password -Message body	Collaborative outgoing anti-spam technique	Once a message is tagged as spam by one SMTP ser the signature of that message is transmitted to all other SMTP server
2011	[53]	-Individual index -Community index	Unfriending vulnerable friends	Correlation between unfriending vulnerable friend security improvement is significantly higher
2011	[62]	-User behavior -Sensitivity and visibility	Item response theory	Behavioral, Quantitative and Qualitative privacy evaluation
2011	[76]	-Number of privacy conflicts -Trust of an accessor -Sensitivity and Visibility	Collaborative privacy	Quantify **multiparty privacy risk** and sharing loss. Evaluate multiparty privacy conflicts by generating a resolving score
2010	[58]	-Interaction level -User similarity	Unsupervised linear model Link-based latent variable model	The estimated link weights result in higher correlat and lead to improved classification accuracy
2010	[57]	-Trust path	Trust path probability	Message paths trustworthiness
2010	[68]			
2010	[54]	-Community membership -Online activity	Privacy-Preference Model as a classifier	**Community structure** of a users is a valuable resource when modeling the users privacy preferen
2010	[74]		Cluster-based reputation	
2009	[79]	-Nodes IP -Reporter Trust -Identity uniqueness	Distributed SocialFilter-repository	Node reputation
2009	[72]	-Geodesic distance -Number of recipients	Classifier: Supervised Learning	Sender reputation system that can accurately classi email senders based on several factors
2009	[65]		Attribute-based encryption	
2009	[75]		Multi-agent reputation	
2008	[55]			
2004	[84]	-Community settings -Number of malicious peers	Trust computation	

Table F.1: Social metrics

Bibliography

Gateway selection

[3] V. Prasanna Venkatesan, C. Punitha Devi, and M. Sivaranjani. “ Design of a Smart Gateway Solution Based on the Exploration of Specific Challenges in IoT ”. In: *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). 00004. Palladam, Tamilnadu, India: Feb. 2017, pp. 22–31 (p. 7, 8).

Network selection

[43] Fayssal Bendaoud, Marwen Abdennebi, and Fedoua Didi. “ Network Selection in Wireless Heterogeneous Networks: A Survey ”. In: *Journal of Telecommunications and Information Technology* 4 (Jan. 2019). 00000, pp. 64–74 (p. 93).

Configuration selection

[21] Francesca Cuomo et al. “ EXPLoRa: Extending the Performance of LoRa by Suitable Spreading Factor Allocations ”. In: *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). 00000. Rome: Oct. 2017, pp. 1–8 (p. 60).

[22] Martin Bor and Utz Roedig. “ LoRa Transmission Parameter Selection ”. In: *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. 2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS). 00042. Ottawa, ON: June 2017, pp. 27–34 (p. 60, 61).

[23] Brecht Reynders, Wannes Meert, and Sofie Pollin. “ Power and Spreading Factor Control in Low Power Wide Area Networks ”. In: *2017 IEEE International Conference on Communications (ICC)* (2017), pp. 1–6 (p. 60, 61).

[24] Andrew James Wixted et al. “ Evaluation of LoRa and LoRaWAN for Wireless Sensor Networks ”. In: *2016 IEEE SENSORS* (2016). 00112, pp. 1–3 (p. 60).

[25] Zhuocheng Li et al. “ 2D Time-Frequency Interference Modelling Using Stochastic Geometry for Performance Evaluation in Low-Power Wide-Area Networks ”. In: *2017 IEEE International Conference on Communications (ICC)* (2016). 00000, pp. 1–7 (p. 61).

[26] René Brandborg Sørensen et al. “ Analysis of Latency and MAC-Layer Performance for Class A LoRaWAN ”. In: *IEEE Wireless Communications Letters* 6 (2017). 00000, pp. 566–569 (p. 61).

[27] Martina Capuzzo, Davide Magrin, and Andrea Zanella. “ Mathematical Modeling of LoRa WAN Performance with Bi-Directional Traffic ”. In: *2018 IEEE Global Communications Conference (GLOBECOM)* (2018). 00000, pp. 206–212 (p. 61).

[28] Alexandru-Ioan Pop et al. “ Does Bidirectional Traffic Do More Harm Than Good in LoRaWAN Based LPWA Networks? ” In: *GLOBECOM 2017 - 2017 IEEE Global Communications Conference* (2017). 00000, pp. 1–6 (p. 61).

[29] Floris Van den Abeele et al. “ Scalability Analysis of Large-Scale LoRaWAN Networks in Ns-3 ”. In: *IEEE Internet of Things Journal* 4.6 (Dec. 2017). 00000, pp. 2186–2198 (p. 61).

[30] Brecht Reynders, Qing Wang, and Sofie Pollin. “ A LoRaWAN Module for Ns-3: Implementation and Evaluation ”. In: 00009. 2018 (p. 61).

[31] Vojtech Hauser and Tomas Hegr. “ Proposal of Adaptive Data Rate Algorithm for LoRaWAN-Based Infrastructure ”. In: *2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*. 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud). 00000. Prague: Aug. 2017, pp. 85–90 (p. 61).

[32] Mariusz Slabicki, Gopika Premsankar, and Mario Di Francesco. “ Adaptive Configuration of Lora Networks for Dense IoT Deployments ”. In: *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium* (2018). 00025, pp. 1–9 (p. 61).

[34] Nikos Kouvelas, Vijay Rao, and R. Venkatesha Prasad. “Employing P-CSMA on a LoRa Network Simulator”. In: abs/1805.12263 (2018). 00000. arXiv: 1805.12263 (p. 61).

[35] Daniel Zucchetto and Andrea Zanella. “Uncoordinated Access Schemes for the IoT: Approaches, Regulations, and Performance”. In: *IEEE Communications Magazine* 55 (2017). 00007, pp. 48–54 (p. 61).

Measurement

[4] Mattia Rizzi et al. “Evaluation of the IoT LoRaWAN Solution for Distributed Measurement Applications”. In: *IEEE Transactions on Instrumentation and Measurement* 66.12 (Dec. 2017). 00000, pp. 3340–3349 (p. 7).

[5] Luca Feltrin et al. “LoRaWAN: Evaluation of Link- and System-Level Performance”. In: *IEEE Internet of Things Journal* 5.3 (June 2018). 00000, pp. 2249–2258 (p. 8).

[29] Floris Van den Abeele et al. “Scalability Analysis of Large-Scale LoRaWAN Networks in Ns-3”. In: *IEEE Internet of Things Journal* 4.6 (Dec. 2017). 00000, pp. 2186–2198 (p. 61).

Plateform

[1] Johan Bregell. “Hardware and Software Platform for Internet of Things”. In: *Master of Science Thesis in Embedded Electronic System Design* (2015). 00002 (p. 7, 9, 37).

[19] *Évaluation et Amélioration Des Plates-Formes Logicielles Pour Réseaux de Capteurs sans-Fil, Pour Optimiser La Qualité de Service et l’énergie*. 00000. URL: http://docnum.univ-lorraine.fr/public/DDOC_T_2016_0051_ROUSSEL.pdf (visited on 04/17/2019) (p. 31).

Weather

[15] Sérgio I. Lopes et al. “Design of Compact LoRa Devices for Smart Building Applications”. In: *Green Energy and Networking*. Ed. by João L. Afonso, Vítor Monteiro, and José Gabriel Pinto. Vol. 269. 00000. Cham: Springer International Publishing, 2019, pp. 142–153 (p. 32).

[46] Rim Fakhfakh, Anis Ben, and Chokri Ben. “Deep Learning-Based Recommendation: Current Issues and Challenges”. In: *International Journal of Advanced Computer Science and Applications* 8.12 (2017). 00002 (p. 94).