UNIVERSITÀ DI PISA

Scuola di Dottorato in Ingegneria "Leonardo da Vinci"

Corso di Dottorato di Ricerca in
Ingegneria dell'Informazione

Tesi di Dottorato di Ricerca

Threats and perspective for the Internet of Things

SSD: ING-INF/03

Autore:

Loris Gazzarrini

Relatori:
Ing. Rosario G. Garroppo
Prof. Giuliano Manara
Prof. Stefano Giordano

Anno 2013

Manuscript received February 28, 2013

Accepted April 9, 2013

SOMMARIO

Il paradigma di *ubiquitous computing* sta lentamente entrando nella vita di tutti i giorni, gli utenti sono sempre connessi e una nuova esigenza di monitoraggio e controllo sta nascendo. Dispositivi di comunicazione intelligenti, le reti domestiche multimediali e l'automazione industriale sono alcune coniugazioni possibili del paradigma di ubiquitous computing che sono ora disponibili per l'utente finale. La diffusione di questi sistemi é infatti destinata a crescere, spinta da entrambi i mondi accademico e industriale - la quantitá di lavoro di ricerca in questo campo é in aumento, e diverse aziende hanno messo le proprie soluzioni sul mercato. Nella visione di molti, l'utente sarebbe poi in grado di godere dei vantaggi di un sistema intelligente e impercettibile che si adatta l'ambiente ed ai suoi bisogni e unisce tutte le applicazioni e i servizi in un unico sistema integrato e facile da controllare.

In questo lavoro vengono valutate alcune delle sfide architettoniche di questo nuovo modo di interagire tra l'utente e il suo ambiente circostante. Vengono mostrate la progettazione e la realizzazione di un SIP-based Home Gateway per il controllo remoto di *Smart Objects* in un ambiente domotico. É presentata anche un'architettura basata sul protocollo SIP per realizzare un sistema di domotica capace di interagire con dispositivi eterogenei e con varie interfacce utente, l'architettura si basa sull'uso del protocollo SIP come piano di controllo comune ed é centrata sul SIP Gateway Home. Per valutare le capacitá del sistema descritto abbiamo effettuato anche una valutazione delle prestazioni, considerando i due problemi principali per questo tipo di dispositivi: scalabilitá ad un elevato numero di richieste di servizio per secondo e l'interferenza/coesistenza di dispositivi appartenenti a diverse tecnologie/standard (ZigBee, Bluetooth, e Wi-Fi) presenti sullo stesso dispositivo. Sono stati valutati anche i problemi di sicurezza attraverso lo studio sperimentale di un Intrusion Detection System per attenuare tali problemi.

ABSTRACT

The ubiquitous computing paradigm is slowly entering into everyday life. Users are always connected and a new need for monitoring and remote control is arising. Smart communication devices, ambient intelligence deployments, multimedia home networks, and building automation are some possible conjugations of ubiquitous computing that are now available to the end user. The pervasiveness of these systems is in fact expected to grow, pushed by both the academic and industrial worlds – the amount of research work in this field is increasing, and several companies have put their own solutions on the market. In the vision of many, the user would eventually be able to enjoy the benefits of an intelligent and unperceivable system that adapts the environment to his/her needs and merges all applications and services into a unified and easy-to-control system. In this work we present the evaluation of some of the architectural challenges of this new way of interacting between the user and his surroundings, the design and implementation of an SIP-based Home Gateway for remote control of *Smart Objects* in a domotic environment is shown. An architecture for realizing a domotics system with heterogeneous devices and user terminals is shown, the architecture is based on the use of SIP as the common control plane and is centered on the SIP-based Home Gateway. To asses the capabilities of the described system we performed also a performance evaluation, considering two main problems for such a device: scalability to a large number of request per seconds and interference/coexistence of devices belonging to different technologies/standard (ZigBee, BLuetooth, and Wi-Fi). Even security issues are evaluated and an experimental study of an Intrusion detection system is performed to address these issues.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF PUBLICATIONS

International Journals

1. Garroppo R.,Gazzarrini L.,Giordano S.,Tavanti L., *"Experimental evaluation of a SIP-based Home Gateway with multiple wireless interfaces for domotics systems"*, in Journal of Computer Networks and Communications, Vol 2012, DOI: 10.1155/2012/190639.

2. Callegari C., Gazzarrini L., Giordano S., Pagano M., Pepe T., *"Improving PCA-based anomaly detection by using multiple time scale analysis and KullbackLeibler divergence"*, in International Journal of Communication Systems, September 2012, DOI: 10.1002/dac.2432.

International Conferences

3. Garroppo R., Gazzarrini L., Giordano S., Tavanti L., *"Experimental assessment of the coexistence of Wi-Fi, ZigBee, and Bluetooth devices"*, International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2011), Jun 20-23, Lucca, Italy.

4. Michel A., Caso R., Tavanti L., Gazzarrini L., Garroppo R., Nepa P., *"Design and Performance Analysis of a Slot Antenna Integrated in a Photovoltaic Panel"*, International Symposium on Antennas and Propagation and USNC-URSI National Radio Science Meeting (AP-S/URSI 2012), Jul 8-14, Chicago, Illinois, USA.

5. Bonelli N., Gazzarrini L., Giordano S., Procissi G., Trammell B., *"On Memory Allocation for HighSpeed Packet Analysis Applications"*, International Conference on Communication (ICC 2013), Jun 9-13, Budapest, Hungary.

6. Callegari C., Gazzarrini L., Giordano S., Pagano M., Pepe T.,*"When Randomness Improves the Anomaly Detection Performance"*, International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL 2010), 7-10 Nov, Rome, Italy.

7. Callegari C., Gazzarrini L., Giordano S., Pagano M., Pepe T.,*"A Novel Multi Time-Scales PCA-based Anomaly Detection System"*, International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2010), July 11-14, Ottawa, Canada.

8. Callegari C., Gazzarrini L., Giordano S., Pagano M., Pepe T.,*"A Novel PCA-based network Anomaly Detection"*, International Conference on Communication (ICC 2011), Jun 5-9, Kyoto, Japan.

9. Callegari C., Gazzarrini L., Giordano S., Pagano M., Pepe T.,*"Detecting Network Anomalies in Backbone Networks"*, [poster] Recent Advances in Intrusion Detection Symposium (RAID 2010), Sep 15-17, Ottawa, Ontario, Canada.

CHAPTER 1

INTRODUCTION

In the golden age of Information and Communication Technology (ICT) users are always connected and so a new need for remote control and sensing is arising. To match user requests a new way of interacting with everyday devices is becoming hot topic in research and development, this new area spans from new electronical device requirements to intuitive user interfaces leading to new models of interaction between the user and his surroundings.

We everyday assist to the explosion of this new interaction paradigms between the user and his surroundings, leaving the classical *active* monitoring and control in favour of a new *reactive* paradigm. In this new scenario are the object themselves (*Smart Objects*) that can notify events and react on their own to those events through the interaction with other *Smart Objects*. An example can be used to clarify this concept, a fire has broken inside the user's house:

- 40 years ago scenario: the house goes on burning until the neighbours watching out from the windows can see flames and call the firefighters (minutes-hours), the house is burnt
- 15 years ago scenario: the alarm system detects the smoke and starts sounding until the neighbours can ear it and call the firefighters (minutes), the house is almost burnt
- nowadays-near future: the alarm system detects the smoke and notifies to the firefighters and to the user that a fire has broken (few seconds), firefighters can save user's house from burning (hopefully)

The meaning is: a *Smart system* can detect events and react in a fast and reliable way, enhancing the user experience. In this scenario we are committed to replace everyday object/systems with *Smart Objects* to enable those new interaction paradigms, those *Smart Objects* are the basic building blocks for the Internet of Things (IoT).

1.1 Interconnecting Smart Objects: The rise of the Internet of Things

Two approaches are currently used/studied for Interconnection of *Smart Object* :

- *Direct connection* of *Smart Object* to the Internet.
- The various technologies/networks composed of *Smart Object* are connected to the Internet through a Gateway, that must be specific for every technology.

The nowadays scenario is a mixed approach. Some notable examples of spreading *Smart Object* directly connected to the Internet is going on (see 6LowPan), but the vast majority of *Smart Object* and *Smart systems* using proprietary standards is connected through custom gateways. *Direct connection* is a complex task, and is not backward compatible with already deployed systems, we will face all connected issues in section 1.1.1 with particular interest in domotic systems.

Our approach is oriented to the IoT as a network of Gateways that can support and interact directly with *Smart Object* that can be connected to the Internet, and with other Gateways through a common control plane. In our vision this control plane could be SIP-based, exploiting the flexibility of the already deployed SIP architecture. This idea will be detailed in the following chapter.

1.1.1 Domotic systems Control Plane

In this Section we just draw a sketch on the current state of the art in domotics systems, with specific focus on the works whose topic is most similar to ours.

Starting from the domotics area, some authors approached the integration between WSNs and control plane protocols by bringing customized or reduced versions of SIP or REST on the sensor nodes.

For example, Luckenbach *et al.* [40] employed REST to provide clients connected to the Internet with the ability to directly interact with MICAz sensors. However, an interface between the clients and the sensors is still necessary and is provided by a custom gateway that maps every HTTP request to a TinyREST message and forwards the message to the sensors.

Similarly, Krishnamurthy and Lange [31] [32] proposed TinySIP, an architecture to offer to multiple clients the access to sensor-based information via SIP. The advantage is that the

user interacts with a client he/she is already familiar with. TinySIP supports the concept of session, as well as the publish/subscribe and instant messaging approaches. Similarly to [40], this solution needs a gateway for mapping SIP messages to and from TinySIP. Moreover, a "zone manager" is necessary for sensor node registration.

These kind of approaches suffer from a series of drawbacks. Since the device are resource constrained, the protocols must be stripped of many functionalities. Due to the particular operative system running on the sensor nodes, the development times might be non-negligible. Also, given the high heterogeneity of the devices, it might be necessary to repeat and modify the customization and development steps for every technology that is going to be integrated into the system. Finally, compatibility with deployed hardware and software is not retained. Conversely, our framework moves the development effort to a single high-end device (the SHG), allowing faster implementation times and full compatibility with both existing sensor and actuator devices and also with the SIP protocol.

An approach that follows the philosophy of making an open and flexible service platform can be found in [10], whose authors started their work from an architecture similar to ours. The proposed Device Integration Framework is based on the creation of a middleware with a two-levels architecture. In the lower level, the middleware provides a set of functions that allow integrating the different technologies used in the devices that handle the multimedia services. In the higher level, a framework that interacts using SIP has been realized to allow connecting the various SIP and non-SIP components to create advanced services.

The work closest to ours is perhaps the one by Bertran *et al.* [12] [11], who tested SIP as a universal communication bus for home automation environments. A SIP gateway and a series of SIP adapters and interpreters have been implemented and deployed to make all devices SIP compliant. However, there are some aspects that may put our framework one step ahead.

Bertran *et al.* did not consider the issues with addressing and reachability of the single DSAN devices. Conversely, we designed a functional addressing scheme that greatly simplifies the user interaction and does not require the DSAN nodes to register to any SIP server or other additional entities. Then, we devised a way of keeping the compatibility not only with the DSAN elements, but also with the user terminals. This allowed us to provide the user also with functionalities that are not natively supported by his/her device. This paradigm can even be extended to ensure forward compatibility with new domotics services. Conversely, Bertran *et al.* did not pay much attention to this aspect. A third distinguishing point is in the adaptation between the SIP and the DSAN worlds. While Bertran *et al.* design a single software module to be put in the gateway, we perform this operation in two steps,

via the DFA layer. This allows to decouple the implementation of the two domains, making the system more flexible. Finally, we studied in much more detail the integration with two possible DSANs, namely ZigBee and Bluetooth, and showed how it is possible to exploit their features to simplify the integration into the system. In [11], the main focus of the experimental platform was on the performance figures of the gateway (which, if we consider the current hardware technology, might not be the most relevant hurdle to the domotics development, as proved by our tests in Section 3.5.2).

A major disadvantage that is common to proposals like [10], [9], and [12], is the need for every home device to register with its own URI. When the number of devices increases (heavily monitored and automated buildings may have hundreds of nodes), the user capability of handling them through their URIs is clearly hampered. The same shortcoming applies to the zone manager solution proposed by [31], in which the majority of the communications are possible only by knowing the address of each gateway to which the sensors of interest refer. On the other hand, in our system the sole SHG must register to an external SIP server (unless the SHG itself implements a registrar) and we can mask the multitude of DSAN nodes by means of the "functional addressing" method.

1.2 Security for The Internet of Things

Security is important for *Smart Objects* because they are often deployed in scenarios where reliability is a key feature and a failure may lead to disastrous results: think about on-body systems or critical infrastructure like electrical power grids.

Security for the IoT is a challenging task, due to the heterogeneity of its components ranging from small battery powered devices to servers/gateways that have to face all the threats of the Internet. *Smart Objects* often employ microcontrollers that cannot execute complex decryption algorithms in reasonable time, due to their low-memory, low-power, and low-energy nature. The environment in which *Smart Objects* operate is even worst than common general purpose computing systems due to their wireless nature combined with low power algorithms employed to reduce power consumption. New threats arise everyday for this kind of systems. These threats can be faced only using Anomaly Based Intrusion Detection techniques. To reach this target we studied and tested an Anomaly Based Intrusion Detection System based on Principal Component Analysis. Our tests were focused on backbone networks, that in our vision will be the network composed by gateways and *Smart Objects* directly connected to the Internet. In any case, the algorithms are designed and implemented mainly to be used

against denial of service (DoS and DDoS) attacks, which are similar to many attacks suffered by Wireless Sensor Networks such as the Sleeping Deprivation Attacks or attacks carried out by Jammers.

1.3 Real World Applications

To clarify the impact on the real world of the Internet of Things in this section we will detail a set of real application scenario, many of them already present in standards and in use in real world systems:

- Home Automation: this field targets smart homes that can control appliances, lighting, environment, energy management and security, as well as the expandability to connect with other networks and enable remote control and monitoring from the user.
- Smart Energy/Smart Grid: Smart grid is one of the major application for *smart object* networks. It relies on the development of systems that monitor, control, and automate the production, delivery, and use of energy and water, in order to improve grid efficiency, performance, and reliability.
- Health Care: a secure and reliable monitoring and management of non-critical, low-acuity healthcare services targeted at chronic disease, aging independence and general health, wellness and fitness.
- Building Automation: the target is a secure and reliable monitoring and control of commercial building systems, there are currently miles of existing data cable installed within commercial buildings. Wireless building solutions embody the prevailing goal of sustainable buildings: Reduce, Reuse, Recycle.
- Smart Cities and Urban Networks: ubiquitous networks can modify the citizens interaction with their surroundings, areas like Intelligent Transport System(traffic flow management, speed control, vehicle tracking), Public safety (access control systems, alarm monitoring, emergency warning), Environmental data collection, Services (public lighting control).
- Telecom Services: targets a wide variety of value-added services, including information delivery, mobile gaming, location-based services, secure mobile payments, mobile

advertising, zone billing, mobile office access control, payments, and peer-to-peer data-sharing services.

1.4 Outline

The work is organized as follows: Chapter 2 addresses the design and implementation of an SIP-based Home Gateway for remote control of *Smart Object* in a domotic environment. In particular the architectural challenges and the choice of SIP protocol are justified. Chapter 3 asses the performances of such a device. Performance evaluation is performed considering two main problems for such a device: scalability to a large number of request per seconds and interference/coexistence of devices belonging to different technologies/standard (ZigBee, BLuetooth, and Wi-Fi). The latter chapter 4 face the previously introduced problem security for the IoT. In particular we have developed an Anomaly Based Intrusion Detection System, to asses the performances of the proposed method we used freely available datasets of traffic captured over the Internet.

CHAPTER 2

A SIP-BASED HOME GATEWAY FOR DOMOTICS SYSTEMS: FROM THE ARCHITECTURE TO ZIGBEE INTEGRATION

Domotics refers to systems that control a number of home services, such as lighting, HVAC, communications, and entertainment, in a integrated and automatic way, allowing the user to manage them by means of heterogeneous terminals, either at home or remotely. Sensors and actuators are one of the major components of a domotics system, and several standards exist to connect and control them. However, dealing with devices belonging to different technologies in a uniform way is still an open issue.

To achieve this kind of interoperability we propose a domotics framework based on the Session Initiation Protocol (SIP). While SIP is used to build a common control plane, a SIP-based Home Gateway (SHG) is in charge of translating the user commands from and to the specific domotics languages. Since the SHG retains the compatibility with the existing SIP infrastructure, our architecture allows the user to control all domotics devices through his favourite SIP client. Particular attention has been paid to the usability and scalability of the system, which brought us to define a functional addressing and control paradigm. A working prototype of the SHG and a customized SIP event package have been used to provide a proof-of-concept of our architecture, in which the SHG has been interfaced with a ZigBee network.

2.1 Introduction

The ubiquitous computing paradigm is slowly entering into everyday life. Smart communication devices, ambient intelligence deployments, multimedia home networks, and building automation are some possible conjugations of ubiquitous computing that are now available to the end user. The pervasiveness of these systems is in fact expected to grow, pushed by both the academic and industrial worlds – the amount of research work in this field is increasing, and several companies have put their own solutions on the market. In the vision of many, the user would eventually be able to enjoy the benefits of an intelligent and unperceivable system that adapts the environment to his/her needs and merges all applications and services into a unified and easy-to-control system.

When focusing on the home environment, *domotics* is the word that condenses such concepts. Domotics refers to a system that control several (or all) home "services", such as lighting, HVAC (heating, ventilation, and air conditioning), communications, security, healthcare, and entertainment, in a integrated and automatic or semi-automatic way, allowing the user to manage them from a series of heterogeneous devices (e.g. touch panels, remotes, mobile handsets, smartphones), either at home or from anywhere in the world. In the domotics archetype, all subsystems are able to talk to each other and interact in a seamless manner, realizing an intelligent structure that improves the quality of life, reduces the costs, and achieves energy savings. To put this paradigm into practice, the communication among the single devices and between the various subsystems is the fundamental operation. Hence, wired and wireless networks will be one of the building blocks of the present and future domotics solutions. On top of this somewhat "physical" element, a common control plane is also necessary, in order to unify the management operations into a single and portable user interface.

One of the major components of a domotic system is the set of sensors and actuators. These usually come in the form of one or more networks, backed either by a single technology or by differerent ones. In the remainder of the chapter we will use the acronym DSAN (domotics sensor and actuator network) to refer to one such sub-system.

Sensor networks[1]are a maturing and widely studied technology employed for various applications such as remote monitoring, remote control, home automation, and smart metering. Among them, wireless sensor networks (WSNs) are the version that is growing faster, due to shorter deploying times and simplified configuration. Several technologies and standards are nowadays available for the implementation of a WSNs [50], but ZigBee is undeniably

regarded as the most interesting one. This is beacause it ensures interoperability among devices built by different manufacturers and allows the deployment of large and almost entirely self-configuring networks. Furthermore, the ZigBee specification [55] defines a series of application profiles that targets the most common scenarios in which a sensor network might be useful and profitable. Each profile consists in a set of functions and a set of devices that must implement those functions in order to satisfy the expected application needs. As a result, application profiles allow engineers and enterprises to design and deploy a ZigBee WSN in relatively short times.

Controlling the network can also be easily achieved by means of the existing profiles and devices. However, neither the generic ZigBee specification nor any application profile defines a common control plane that is suitable to contemporarily manage devices belonging to different profiles. Even the recently standardized ZigBee Gateway [56] is meant to provide solely an interface towards external protocols, such as REST and SOAP, without caring for the interaction and coordination among the various profiles that may coexist behind it. As a result, the burden of coordinating and making devices running different profiles interoperate is left entirely to the system implementer. Indeed, a scenario with mixed profiles is not so uncommon, especially in those environments where multiple services might be requested. One such example is exactly the "smart home" or domotics concept, in which the Home Automation, Smart Energy, and Telecom Services profiles (just to cite the most appealing ones) might all be present.

From the user perspective, the devices belonging to the diverse profiles of the domotics network can be contolled exploiting the features defined by each profile, typically through dedicated appliances located in the house (e.g. a touch panel, a smart telephone, a TV remote). However, this paradigm no longer holds for remote control operations that occur when the user is far from home. In this case the user would normally have a single device at hand, such as a notebook or a smartphone, by means of which he/she would like to control any device in the home, not just those belonging to a specific profile, and possibly in a profile-transparent way, i.e. without complex profile configuration or selection procedures.

The situation grows worse when thinking to cooperation and interaction with other popular, non-ZigBee devices, such as Bluetooth, Z-Wave, Wi-Fi, KNX, LonWorks, or Homeplug appliances. Currently, with the existing technologies, this operation is not even possible.

To overcome these drawbacks and advance the domotics state-of-the-art, we propose an

[1]The term "sensor" is often used for both sensors in the strict sense, and also for actuators.

architecture to gain interoperability among devices belonging not only to different ZigBee profiles but even to different technologies – hence we address the general DSAN paradigm. In our vision, a common control plane is realized through the Session Initiation Protocol (SIP) [46]. Thus the architecture encompasses a *SIP-based Home Gateway* (SHG) and, on the user side, a generic SIP client. In this scenario the SHG translates the user commands from and to the specific DSAN language, thus allowing the user to control all domotics devices either at home or away from it, using his mobile terminal or his favourite SIP client, in a transparent, uniform, and simple way. The SHG is devised to retain the compatibility with the existing SIP infrastructure, which can therefore be exploited in full.

We also provide a proof-of-concept of our SIP-based domotics architecture by means of a working SHG prototype. A customized SIP event package and a notification server have also been developed to validate a possible extension to new services. On the user side, the SHG was interfaced with both a common SIP client and a client we developed with some "extended" features. On the DSAN side, the SHG interfaces with an actual ZigBee network. The synergies between SIP and ZigBee have also been highlighted and exploited for a more efficient implementation.

The chapter begins with presenting the major requirements that a domotics system should meet. We then explain the reasons for employing SIP as the common control plane and describe the general reference architecture for a domotics system controlled through SIP. The implemented SIP-based Home Gateway and its integration with a ZigBee DSAN are then discussed in detail. We also report the outcome of some tests performed on our prototype system. Finally, we perform a comparison with the related work, throwing light on the most significant differences.

2.2 Domotics requirements

Implementing the domotics concept is indeed a non-trivial task. The complexity of the system demands for a series of requirements that allows an easy integration among the sub-systems and the development of a "friendly" and always available user interface. A set of major requirements is represented by the following list (see also [47] and [10] for similar surveys).

- The user would want the DSAN devices to perform disparate actions, from e.g. reading the temperature value of the dining room to closing the windows in case of rain. In other terms, the user would want to send commands, and possibly have a feedback. But

commands can also be exchanged among the various domotics entities, transparently to the user (e.g. the HVAC could be automatically turned on at a given time). Hence the domotics system must implement and provide a *request/response* paradigm.

- Both the user and the system should be promptly notified when events of some importance occur in the environment. This means that the domotics network is an event driven system, which is expected to react to changes in the environment where the sensors are placed. Hence, asynchronous and/or periodic *event notification* is a key element for remote control and monitoring.

- While commands and events perfectly suit the need of exchanging small amounts of data in very short times, for larger and continuous data transferrals a different paradigm must be followed. This typically involves the use of *sessions*, which allow the streaming of various types of data over a period of time (audio and video are the most common ones, but also fast varying sensor readings might be of interest).

- The extensive adoption of mobile devices such as smartphones and tablets changed the way of interacting with the global network, bringing connection available everywhere. As a consequence, the user should be regarded as a *mobile user*, who would want to control his/her home network not only when he/she is at home, but also from different places and via diverse access technologies (e.g wireless LAN, cellular, ADSL). Thus the domotics system must allow for both a technological and a geographical mobility.

- Various sub-systems can contribute to build a "smart home". Each sub-system may use its own communication technology and application semantic. Indeed, numerous manufacturers are already offering their own solutions to address a vast range of domotics applications. These solutions are based on various *technologies* and employ a disparate number of devices. Integrating the whole of them might be a tremendous, but nevertheless essential task.

- Despite the heterogeneity of the available (and future) domotics devices, the user would hardly be keen on using several and different human interface devices (HIDs), remembering the network addresses of every device of the DSAN, or understanding and learning other technology-specific aspects of its domotics system. Conversely, it would be beneficial if he/she could interact with a unique interface layer, associating mnemonic names to the devices and their functions (e.g. the room where they are

placed and/or the action they perform). Therefore, a truly appealing domotics system should integrate these sub-systems at both the technological level (see above) and also at the user (interface) level. In this last respect, realizing a *functional addressing and control scheme* should be one of the objectives of the designer.

- While the domotics idea is slowly gaining field, *communication services and entertainement systems* had long since settled into everyday life. Therefore, a sensible design choice would be to seamlessly include these services into the domotics platform (see [12] for some interesting examples).

- The success of a technology depends not only on the user side aspects, but also on the possibilities that are given to the designers and developers. The widespread adoption of domotics is therefore subject to the availability of a simple and efficient *development frameworks*, to enable short implementation and deployment times, and to ease the installation and configuation.

- Even though several domotics and building automation scenarios have already been thought of, the attractiveness and survivability of the domotics solution depends also on its *openness to future* applications and ideas. In this respect, the possibility to extend the system to new technologies (either standard or proprietary) and paradigms without rethinking it from scratch is undoubtedly a central topic.

- Due to their ubiquitous, distributed, and open nature, domotics services are inevitably subject to many *privacy and security* implications. Authentication mechanisms and other means and models to make the domotics system secure should be a further point to be put on the designer agenda.

2.3 Choosing SIP

Among the many options for realizing the common control plane (see e.g. [47], [13], [27], [16], [22]), we selected the Session Initiation Protocol for its numerous advantages. These are illustrated in the following, with specific reference to the requirements presented in Section 2.2.

From the conceptual point of view, which relates to the operations that are to be carried out by the control plane, SIP provides a set of *methods* that fit well the requirements of DSAN control and management:

- Real time command delivery and sensor data collection are efficiently performed by means of simple messages with no connection-oriented communications. The low overhead of the `MESSAGE` method (no setup phase is needed) perfecty matches the requirements of these request/response operations.

- A Publish/Subscribe-Notify semantic is available in SIP specifications and allows the user to be promptly notified of events that occur in the network. The desired time between two notifications and the events of interest can also be specified by the user. This allows an almost direct mapping of asynchronous and/or periodic event notifications to SIP methods.

- The core SIP infrastructure transparently manages the movement of the user between different points of attachment to the network. SIP User Agents take advantage of the `REGISTER` method to inform the network elements of the host at which they can be reached. Proxy servers can thus locate the user on behalf of any contacting client. As a result, SIP natively enables user mobility.

- SIP has been natively designed to offer session management capabilities (i.e. session creation, modification, and tear down). The usage of SIP as the control plane protocol for Multimedia over IP and IP telephony services is constantly increasing.

From a more practical and implementation perspective, we can identify the following key points:

- SIP is a text based protocol, which means that message building and parsing is a relatively uncomplicated task. Also, SIP parsers and interpreters are widely available. Hence the development effort is greatly reduced.

- The body of SIP messages is flexibly structured, and can contain a wide variety of information. This allows an easy extension of the protocol to support customized DSAN-related data and commands.

- A huge SIP infrastructure is already deployed and working, hence there is no need to deploy new infrastructural elements (either servers or core-network software).

- SIP works at the application layer, which means that it is transparent to the underlying physical and networking technologies. Thus it can work as a gluing layer for heterogeneous systems.

A further valuable asset of SIP is the use of mnemonic names. Every SIP resource is associated to a URI (uniform resource identifier), a mnemonic text pattern based on the same syntax established for web services. This allows the user to remember names rather than complex numeric addresses. A way of exploiting this feature to further simplify the user experience is presented later on in the chapter.

A last, though security and privacy are out of the focus of the chapter, a note on these aspects is due. SIP can take advantage of a set of options that allow to secure both the control plane and the data plane of a SIP-based communication. For an in depth analysis of the available security options of the SIP architecture we address the reader to [18]. In that paper, the authors also propose a rigorous definition of five profiles to build different levels of security for a SIP-based system. Such profiles can be applied to a domotics system as well.

2.4 System Architecture

The general architecture of the conceived domotics system is illustrated in Figure 2.1. We can identify four major elements: the Clients, the SIP Servers, the SHG, and the DSANs. These are described in the next subsections.

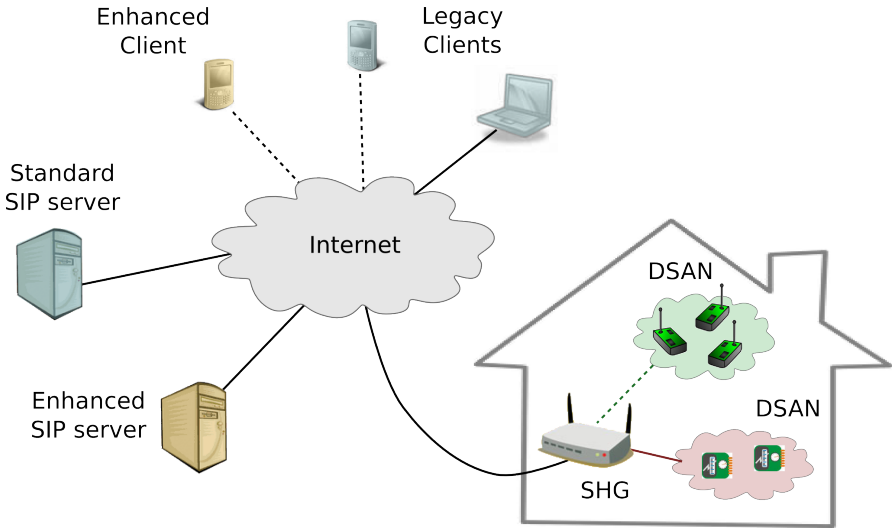

Figure 2.1: *Architecture of the SIP-based domotics system.*

The expert reader may have noticed that this kind of architecture is not completely novel.

A similar picture has been presented for example by [10] and [12]. This means that the general framework can be considered quite settled. What makes the various works different is how the SIP is integrated into the architecture, what semantics are taken into account, how they interact with the other components of the system, and how they can be beneficial to the user. Our work, in particular, targets a more scalable, transparent and painless integration, especially from the user perspective. A more in depth discussion about the literature and the differences with the cited works can be found in Section 3.2.

2.4.1 Client

The Client is any device at the user's disposal supporting a minimum set of SIP methods. All widely known SIP clients (e.g. Linphone [4], Jitsi [2], Bria [24]) fit this description, without the need for any customization or added feature. This is a key factor for the early adoption of our proposal and for backward compatibility with already deployed hardware and software: any user in possession of a Smartphone or a PC with a SIP client application can control the DSANs. However, for the full exploitation of the domotics features envisioned in our system, it might be necessary to enhance the Client with further capabilities and SIP methods. Therefore we can distinguish two types of Client:

- Legacy Client: a Client supporting only the basic SIP methods;
- Enhanced Client: a modified or upgraded Client that supports domotics-specific operations.

2.4.2 SIP-based Home Gateway (SHG)

The SIP-based Home Gateway (SHG) is the key element of the system. It enables the remote control of the various DSANs by translating the messages and procedures from the SIP world to the specific DSAN technology.This translation actually occurs in two phases. At first, the SIP message is translated into a domotics abstraction of the intended operation. Then, the proper DSAN procedure is executed to implement the requested semantic. The DSAN→SIP translation works in a similar way.

We have therefore introduced an intermediate entity, which we call Domotics Facility Abstraction (DFA), whose goal is twofold: disjoin the implementation of the two domains of the system, and create a single and user-friendly interface. The former goal is meant to ease the development of the SIP and the DSAN interfaces, which may be carried on separately. In

addition, the abstract definition of the domotics services matches the functional addressing and control paradigm employed for the user interface, leaving SIP on the pure transport layer, which is thus hidden to the user.

A pictorial description of the framework can be seen in Figure 2.2. The user is immersed into the functional service abstraction, which is implemented through the user interface on the Client, and is understood by the DFA at the SHG. Examples of such framework can be swiftly provided. Imagine a user at a remote place (e.g. returning from a travel abroad) whishing to find the home at a comfortable temperature. He/she can then issue a simple command, such as "set home temperature 20". The Client then wraps the command into the proper SIP procedure and conveys it to the SHG, where it is mapped to a DFA service. The SHG then cares for translating it into a proper set of DSAN operations, such as starting the HVAC system and setting an alarm threshold on the temperature sensors deployed in the house. When the desired temperature is reached, the SHG will automatically stop the HVAC system. We can see from this example that the user demands a specific operation to be performed, but is clearly ignoring all the technical processes of the domotics system, which are transparently handled by the SHG by means of the DFA layer.

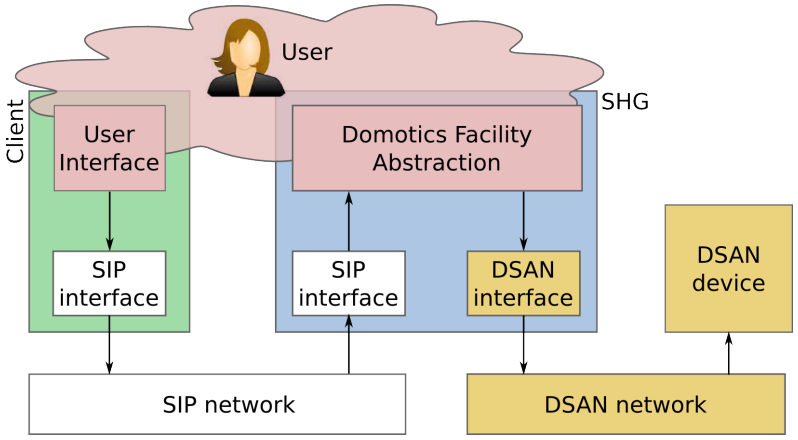

Figure 2.2: *The functional paradigm implemented via the Domotics Facility Abstraction.*

The SHG is also in charge of using the proper set of SIP methods in order to ensure backward compatibility with Legacy Clients and full operativity to Enhanced Clients. Furthermore, it can perform "intelligent" operations, such as piloting devices of a DSAN in response to events from another DSAN (e.g. turn on a KNX-enabled heater after a ZigBee

sensor has reported a temperature/humidity change).

2.4.3 Server

A SIP server is necessary for device registration and to support the Publish/Subscribe-Notify semantic. This server can be any server on the SIP infrastructure providing such functions. Similarly to Clients, the distinction between “legacy” and “enhanced” involves the servers too. Typically, existing servers are unaware of the domotics functions, hence only standard SIP features are available. However, as the domotics paradigm gains ground, new domotics-aware servers can be expected to appear. An example of such server is given later on in the chapter. As an always valid alternative, the SIP server functions can be implemented directly in the SHG to increase the system self-reliance.

2.4.4 Sensor and actuator networks

The sensors and actuators deployed around the house are usually connected through one or more domotics networks (DSANs). Each DSAN has its own transport technology and can use either wired or wireless connections (or both). All DSANs are then attached to the SHG.

2.4.5 Selected SIP methods

In this section we provide a brief description of the SIP mechanisms we reckoned should be integrated into the domotics system. Particular attention is paid to the reasons that drove our choice and how these methods interact with the other elements of the system. Note that we exploit the existing SIP features as they are. Backward compatibility with existing SIP clients, which is a key element of the system, is thus preserved.

Instant messaging

The `MESSAGE` SIP method [19] is used to supply the instant messaging service, i.e. the real time dispatch of short text messages where each text message is independent from the others. Correlation between consecutive `MESSAGE` transactions may be present on the user interface, but is not a intrinsic feature of SIP. Every UA receiving a `MESSAGE` must send an immediate answer to the sender to inform it about the successful or failed reception of the message (in case of success, the answer is `200 OK`). To keep the overhead low, no session set-up is necessary.

This kind of interaction reduces the signaling overhead to the bare minimum and provides the real time behaviour needed for the domotics application. As outlined in Section 2.2, many operations on the DSANs are performed in real time. For example, the `MESSAGE` method is suitable for actions such as light switching on / off, HVAC commanding, temperature reading.

Though the SIP acknowledgement message (such as `200 OK`) can include application data, we preferred to separate the request transaction from the response transaction. This is because the processing time of the request might be highly variable. Think for example to actions such as opening or closing the window blinds, or collecting and merging data from different and sparse sensors. The default 200 ms delay allowed by SIP for sending the final response is clearly inadequate. Therefore, the acknowledgement message is meant to refer just to the correct reception of the preceding `MESSAGE`, whereas a new `MESSAGE` must be sent from the SHG to the user in an asynchronous way to give him/her the response (such as the confirmation that the command has been executed). The same applies when the requested operation fails (e.g. due to the unreachability of a sensor). Thus, four SIP messages are necessary to realize the request/response paradigm.

An alternative method could be using a provisional response, e.g. `100 TRYING`, to suspend the timer associated with the `MESSAGE`, and then conveying the outcome of the action via the `200 OK` message (thus mapping the request/response paradigm to a single `MESSAGE` transaction). However, this method does not shift the problem, because the allowed suspension time would in fact depend on the specific action, thus making the timer management definitely too complex. In addition, provisional responses are not sent reliably.

Figure 2.3 illustrates a possible usage scenario of the `MESSAGE` method. The user asks for some data to be retrieved from a sensor. Accordingly, a `MESSAGE` is sent from the user's SIP client (the so-called UAC) to the SHG, which acknowledges the reception of the message with a `200 OK`. Then the gateway maps the message into a DFA service, and then into the form required by the specific sensor technology, which is dispatched over the proper DSAN interface. When the sensor node is reached by the request, it performs the necessary ations and then sends a reply to the gateway. The SHG can thus build a new `MESSAGE` that is sent to the user.

A very important aspect of the `MESSAGE` method is its compatibility with all existing SIP clients. In fact, every SIP client must support this method, and therefore this ensures that the basic managing functions of our system are also supported.

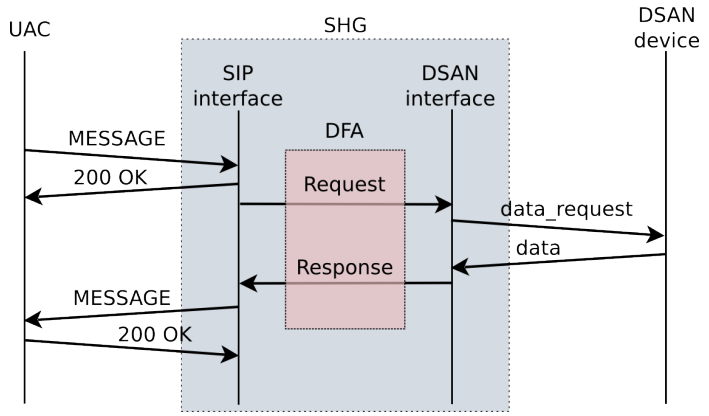

Figure 2.3: *The Request-Response paradigm implemented via the* `MESSAGE` *method.*

Publish/Subscribe-Notify paradigm

Asynchronous notification of events is crucial for a sensor network that, for its own nature, must react to many kinds of changes: signal an alarm after a fire has broken, send measured data about energy produced/consumed in a smart grid scenario, shut the windows when the wind is too strong, and so on. The SIP Event Notification Framework (ENF), defined in [44], provides a way for SIP elements to learn when "something interesting" has happened somewhere in the network. The procedures to allow for the prompt distribution of such events are known as the Publish/Subscribe-Notify paradigm. This paradigm suits well the event driven nature of the DSAN networks, providing a simple and effective way to help the SHG in distributing the information to the interested users.

Going into the details, an initial `SUBSCRIBE` message is sent by the subscriber (the user that is interested in the event) to the notifier (the node that is first aware of the event). If the subscription is accepted, a `200 OK` answer is sent to the subscriber. Then, the events are reported from the notifier to the subscriber by means of the `NOTIFY` SIP method. Notifications can be sent either periodically or when the specific event occurs (or both).

SIP also provides a framework for the publication of event states on a notification server, called Event State Compositor (ESC). This task is accomplished using the `PUBLISH` method [42]. The ESC is then responsible for managing and distributing this information to the interested parties through the ENF.

An example of the complete Publish/Subscribe-Notify paradigm is shown in Figure 2.4.

Besides the entities constituting the domotics system in the strict sense (i.e. the Client, the SHG, and a DSAN), the ESC is also involved. The operations start with the UAC sending a `SUBSCRIBE` request to the notification server, which first acknowledges the reception of the message and then sends the first notification to inform the client about the current state of the event the user is interested in. Successive notifications may be sent either periodically or on the occurrence of some event. The figure shows both of them. The first one is a periodic update of the information, whereas the second one is an asynchronous notification due to the occurrence of the monitored event on one of the sensors of the domotics network.

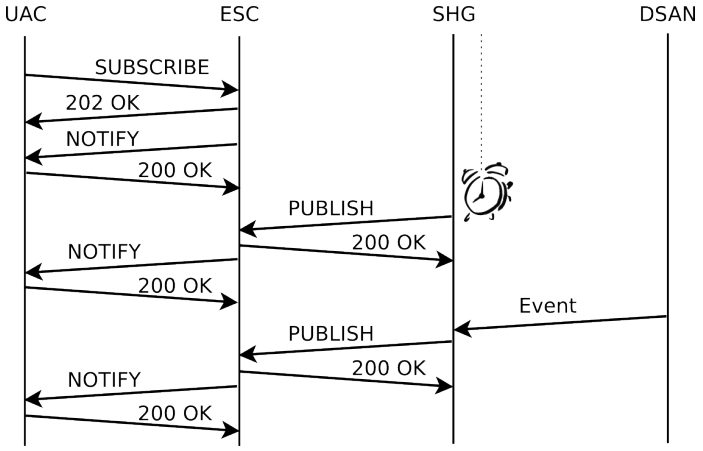

Figure 2.4: *SIP message flow for publish, subscribe, and notify operations.*

Note that the ESC is a logical entity, which can physically reside in diverse parts of the system. As far as our domotics architecture is concerned, we identified three reasonable places where the ESC functions can be performed:

- a server of the SIP core network (either integrated with a Proxy or a Registrar, or standalone); in particular, operators may be interested in deploying their own ESCs to offer customized services;
- the SHG;
- another home server; this may be the case for a solution that puts all SIP server functions (registration, event notification, etc.) on a dedicated device, leaving on the SHG just the domotics-specific SIP procedures.

Registration

When a user wants to start a session with another user (either human or not), he/she generally refers to that user with the SIP URI. The URI does not hold any information about the user location, point of attachment to the network, or device capabilities. Hence, the SIP framework must discover the actual host at which the destination user is currently reachable. This process is fulfilled by SIP network elements such as the proxy servers. They receive the connection request, look for the location of the user, and then forward the request there.

Some mechanism is therefore necessary to create bindings in a location service that associates a SIP URI with one or more "physical" addresses. SIP User Agents can create such binding by sending a `REGISTER` message to a registrar server [46]. The message holds the current location (e.g. the IP address) of the user, so that the registrar can store the binding between the user and its present address in the location service, which is then queried by the proxy servers to locate the user.

This method, which is part of the standard SIP specifications, makes the mobility transparent to both the user and the SHG. The user has just to register every time he/she changes the point of attachment to the network (in fact, this is performed automatically by the UAC) and then the SIP infrastructure will take care of keeping the connection with the SHG alive.

In the outlined domotics architecture, only two elements require to be registered: the user and the SHG. All sensors and actuators of the various sub-systems are managed by the SHG via the specific DSAN technologies (they can be completely unaware of the SIP control plane), and the user refers to them through what we have called the "functional address" (see Section 2.2). As a result, the user can interact with the system by knowing just the SIP URI of the SHG. This makes the system extremely user-friendly and also highly scalable. No matter how many devices are in the house, the user can control them through invariably the same URI (the SHG one), from anywhere he/she is and from any SIP-enabled device he/she is using.

Note that this is neither an intrinsic feature of SIP nor of the domotics concept itself. In fact, many other works exist that deal with these aspects, but rarely do they succeed in achieving scalabilty and ease of use (see Section 3.2 for a detailed review). Rather, this is a notable advantage of the way we built our architecture. Clearly, the positive impact of this approach is greater as the network grows larger.

2.5 Proof of Concept

To put the ideas expressed in the previous Sections into practice, we have realized a small testbed involving all the elements of the architecture. The SHG, being the core and most innovative element, has been built from scratch. A DSAN has been implemented using a series of ZigBee devices. A SIP client, which is the so-called Enhanced Client, has also been developed. Finally, to illustrate the potentials of expansion and customization of our architecture, we have defined and implemented a specific SIP event package for the domotics framework.

2.5.1 The ZigBee sensor and actuator network

ZigBee is a standard for low-cost and low-power wireless sensor and control networks developed by the ZigBee Alliance [55], a non-profit association including more than 150 organizations from large semiconductor companies to government agencies. The ZigBee specifications ensure interoperability among devices from different manufacturers and enable the deployment of large wireless networks.

The ZigBee architecture relies on the IEEE 802.15.4 standard, which provides the radio interconnection of the devices. It defines the Physical (PHY) and Medium Access Control (MAC) layers for short range (tens of meters) and low data rate (up to 250 kbps) wireless connectivity, targeting applications with very low energy consumption. To cover large areas with a single and fully connected network, ZigBee supports a multi-hop communication paradigm based on message routing.

A ZigBee network is composed of three types of devices:

- ZigBee End Device (ZED): this is a reduced function device that can send or receive messages, join or leave a pre-existing network, and does not participate to message routing; it can go to sleep mode to save battery power.
- ZigBee Router (ZR): this node can route data between the ZigBee devices, allow devices to join or leave the network, and perform all the functions of a ZED.
- ZigBee Coordinator (ZC): this is the node that starts the network and lets other nodes join or leave the network; the ZC also performs all the functions of a ZR.

To meet the requirements of the most common application scenarios, ZigBee defines a series of Public Application Profiles (in short: PAPs or just profiles). A PAP contains specific

details about what information a device should communicate and how this device should interact with other devices on the ZigBee network. Multiple applications, also belonging to different profiles, are allowed to reside on a specific node.[2]

To speed up the implementation of devices belonging to the various PAPs and to help the developers and manufacturers in building interoperable devices, ZigBee introduces the concept of *cluster*. A cluster is a set of attributes, data, and commands exchanged between endpoints. An endpoint is a logical wire used to connect distributed applications residing on different nodes. The ZigBee Cluster Library (ZCL) is a collection of clusters and cross-cluster commands that is common across the various profiles.

ZigBee then provides for a mechanism, known as *binding*, to connect endpoints on different nodes. Binding creates logical links between endpoints on devices and maintains this information in a binding table. The binding table also has information about the services offered by the devices on the network. The ZC typically holds the binding table for the whole network.

A notable advantage of this structure (clusters, endpoints and binding tables) is that it allows the implementation of the *service discovery* procedure via bindings. The services available inside the ZigBee network can thus be discovered directly within the ZigBee domain, without resorting to any additional software or external entities. With specific reference to the SIP control plane, this means that there is no need to port the SIP registration procedure to the ZigBee network, since this would be just a duplication of the ZigBee service discovery.

Going now to the implementation details, the nodes of the ZigBee sensor network are based on the Freescale MC1322x board [25], which integrates a 32-bit ARM-7 MCU and a low power 2.4 GHz transceiver. The fully compliant ZigBee stack provided by Freescale was installed on the nodes. A custom application that supports environmental data collection, remote control, and message routing has been developed on top of the ZigBee stack by means of the ZCL functions.

In our testbed we arranged for the collection of ambient data (temperature and pressure), which is then cached on the coordinator node (ZC). The SHG periodically retrieves the data from the ZC and stores it internally, to avoid a rapid outage of the scarce memory space on the node. Temperature and pressure are collected both on regular time basis and on-demand. Both approaches are available to the user, who can either subscribe to this event or ask the

[2] As already outlined in the Introduction, the fact that more profiles can coexist on a node does not imply that they can talk to each other.

SHG to check a specific sensor value (or all readings of all sensors).

As for remote control, in our testbed it takes the form of light control. The MC1322x boards are equipped with an array of three LEDs, which was used to mimic a multi-level light. The user can thus raise or lower the lighting power by simply typing some text in a chat application on his terminal. To this purpose we defined a set of textual commands (see Table 2.1 at page 37). Combining them with the name of a room allows the user to set the desired light level. After the command has been sent, the user receives a feedback from the end node, consisting of the updated light level. If the sensor node is unreachable, or if the command is wrong, the user receives an error message.

Of course, the data collection and remote control implemented in our testbed are just two examples, useful to give a proof-of-concept of the capabilities and feasibility of the proposed architecture. As for data collection, many other request formats are possible, depending only on the fantasy and needs of the application developer. Similarly, the architecture does not set any limit to the number and type of sensors, which can be extended to fit any feasible application scenario. The same applies to remote control. Since a General Purpose Input Output (GPIO) is available on the MC1322x platform, it is easy to understand that it can do much more than controlling simple LEDs, and this paradigm can be extended to as many applications as needed by the system designer.

The last task the ZigBee DSAN is expected to fulfill is a "backstage" activity: connecting all sensor nodes to the coordinator node, and therefore to the SHG. We thus configured all nodes to act as ZigBee Routers. Note that this is a pure networking operation, which is transparent to both the SIP architecture and the user. For an indoor topology, this is an almost indispendsable feature to enable, because of the low a-priori knowledge of the installation premises and the short ranges that can be reached in indoor environments.

2.5.2 A domotics event framework

The SIP Event Notification Framework (ENF) standardized in RFC 3265 [44], and later augmented by RFC 3903 [42], provides just the procedures that enable notification of events (as outlined in Section 2.4.5), but do not define any specific "event package". In other terms, these RFCs do not describe an extension that can be used directly, but only an empty framework that must be completed by other documents that define the concrete semantic. Guidelines for creating these extensions are also provided. Event packages based on the ENF are thus allowed to define customized events and arbitrarily elaborate rules to handle the

subscription, publication, and notification of the events they describe.

Indeed, a few packages for the SIP ENF have currently been ratified. Among them, the Presence package [45] is probably the most popular, and also the one that is implemented in some widely available SIP clients. However, this package does not suit well the needs of a domotics environment, as it provides just a single elementary functionality (the presence of a given user). In addition, the Presence package refers to the bare ENF, without taking advantage of the `PUBLISH` method.

To test our domotics system with a complete and flexible Publish/Subscribe-Notify paradigm, we built a new package, named "Home Automation". The basic features of Home Automation are similar to the ones of Presence, but our package employs the `PUBLISH` method too. We designed it to embed a customized XML text, like the one illustrated in Figure 2.5, which contains domotics specific data (such as the values read by some ambient sensors). In this particular example, the XML snippet is sent from the SHG to an Enhanced Client to report about the readings of the sensors in the *Lab* room and also the current light level. Note that the tags implement a possible functional naming abstraction of the DFA layer. Clearly, this XML scheme can be replaced with any other kind of text format, like REST or SOAP (the ones employed by the ZigBee Gateway [56]).

In order to correctly handle this package, we also built a customized ESC server. We employed Kamailio [3], an open source SIP Server released under GPL, to which we made some modifictations. The changes mainly consisted in adding the specific Home Automation keywords to let it recognize the Home Automation package in a similar fashion to any other package.

Note that the XML text is not touched by the ESC (only a formal check is done), but is passed directly to the subscribers by means of the `NOTIFY` messages. Hence, SIP is immediately able to deliver this information using the existing infrastructure.

Further arrangements had to be taken on the SHG and the clients as well; these are described in the respective sections.

2.5.3 SIP clients

The client is a somewhat critical element of the system. In most of the deployed SIP-clients, very few SIP functionalities are available. Apart from the `REGISTER` and `MESSAGE` methods, which are present in all terminals[3], a number of SIP-clients also features the Presence package (e.g. Jitsi, Bria). This implies the support of the `SUBSCRIBE` and `NOTIFY` methods, but only in

```
<device room="Lab"
                name="dimmablelight">
        <attrib name="Level"
                        value="33">
        </attrib>
</device>
<device room="Lab"
                name="ambientsensor">
        <attrib name="Temperature"
                        value="20">
        </attrib>
        <attrib name="Pressure"
                        value="308">
        </attrib>
</device>
```

Figure 2.5: *Sample XML for the Home Automation event package.*

strict connection with the presence service. From the perspective of the multiple applications and services that a domotics system may offer, these capabilities can not be deemed adequate.

Therefore we developed a test SIP client that supports the full Publish/ Subscribe-Notify paradigm and the Home Automation package described in Section 2.5.2. One such client is in all aspects a SIP-compliant software, but with the extra feature that it can control the DSAN with its native semantic. We call this terminal an Enhanced Client. In contrast, a Legacy Client is any SIP client that understands just the `MESSAGE` method. Both clients have been used in the testbed.

2.5.4 SIP-based Home Gateway

To build the SHG, the only two requirements are the sufficient processing power and memory to run the software, and the capability to interface with the technologies of the particular

[3] In fact, other methods, such as `INVITE`, are present, but they are not relevant for the present work.

sensor networks to control.

A generic single board computer (SBC) with a Texas Instrument OMAP 3530 processor was used. Since the management operations are not particularly CPU-intensive, this hardware is more than adequate. The SHG was connected to a ZigBee coordinator node via a USB interface. This provides the SHG with the physical interface to the ZigBee network. Figure 3.2 shows the prototype SHG.

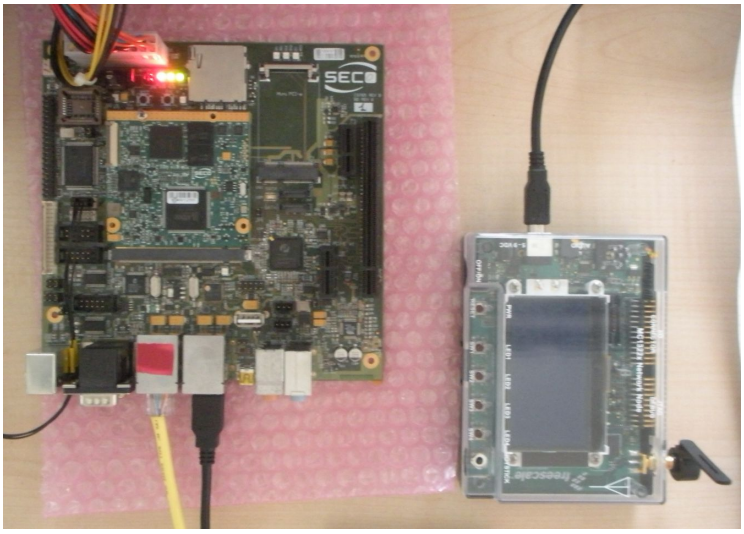

Figure 2.6: *The prototype SHG; the ZC can be seen on the right.*

The SHG software was built on top of Linux, which provides the necessary support and development tools (e.g. a SIP library, the interface drivers). The software that implements the DFA layer and translates the SIP semantics into the ZigBee messages and commands has been written from scratch using the C++ language.

Besides the standard SIP operations on the external side, the translations of the messages, the communication with the various DSANs, the SHG performs also a very sensitive operation. It ensures the compatibility with existing and new clients.

The interoperability with the Enhanced Client is unequivocal, as the Home Automation profile has been defined and implemented by us in both the SHG and the client.

To ensure the backward compatibility with the numerous and already deployed terminals (i.e. the Legacy Clients), instead of acting on the clients, we opted for developing a workaround on the SHG. We conveyed subscriptions and event notifications by means of the `MESSAGE`

method, which is always supported. A special text message, such as "subscribe_home_auto" (see Table 2.1), was used in place of the SUBSCRIBE method. Similarly, on event occurrence, the SHG notifies the user by sending a MESSAGE rather than a NOTIFY. All the background routines of the SIP ENF are then handled by the SHG. This can be accomplished in two ways. Either the SHG becomes itself a notification server or it relays the subscription to a proper ESC. Figure 2.7 illustrates the two options. In the first case (left in the figure), the SHG internally handles the SIP notification procedures. From a logical point of view, this means that the SIP messages are passed from the software implementing the SIP interface to the ESC software; from a practical perspective, the developer may optimize the implementation of the SHG and the SUBSCRIBE and NOTIFY methods might not be generated at all. Alternatively (right in the figure), the SHG performs the subscription and receives the notification from the external ESC on behalf of the client. Thus it must translate the "compatibility" MESSAGE methods to and from the SIP standard methods. In other terms it behaves as both a proxy and a gateway for the Publish/Subscribe-Notify paradigm.

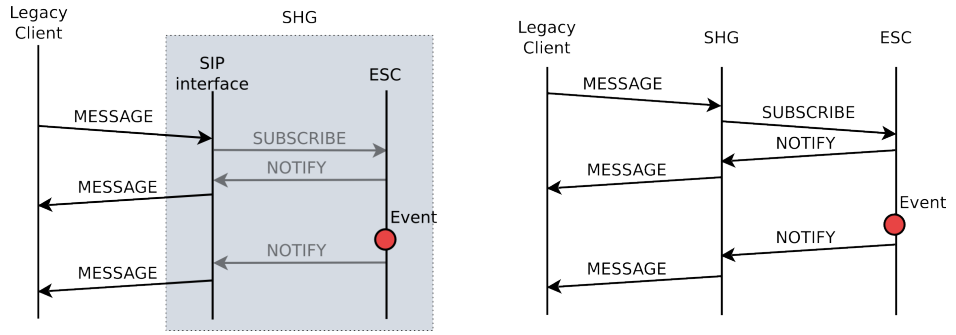

Figure 2.7: *The two solutions for using the Publish/Subscribe-Notify paradigm with a Legacy Client. The 200 OK messages have not been drawn.*

Both solutions are viable and the choice may depend on several factors such as flexibility, overhead, or whether to build a self-contained system or not. In our testbed we opted for the former solution. Apart from the conveying mechanism, this behaviour is coherent with the basic SIP ENF.

In summary, two operating modes coexist on the implemented SHG. When the user is controlling the system through our Enhanced Client, the system works with the extended Home Automation event package; when the user employs a Legacy Client, the MESSAGE workaround is used. Section 2.6.3 shows both of them at work.

2.5.5 SIP servers

Servers build the necessary infrastructure for SIP to work properly. In our proof-of-concept we employed three different servers, which represents the three possible situations for our domotics system.

An external registrar and proxy server provided by *iptel.org* [6] was used as a sample of a pre-existing SIP network element. This server is compliant to exisiting SIP standards, and is completely unaware of the nature of our domotics testbed.

A customized SIP server was built in our lab by means of the Kamailio open source software. As explained in Section 2.5.2, this was necessary to provide support for our Home Automation event package. Hence, this server is representative of a domotics-aware element in the SIP infrastructure. We called this server Enhanced Notification Server (ENS). Note that the ENS besides executing the ESC functions, can also perform the registration and proxy procedures as well.

Finally, a SIP server was also set up within the SHG. This solution is the archetype for a self-contained domotics system, in which the gateway provides all the necessary SIP functionalities. For the sake of conciseness, however, we report only the tests involving the compatibility with a Legacy Client, since the other functions (registration, location, proxy) are in all aspects identical to the ones performed by the *iptel.org* server and therefore do not add any useful contribution.

2.6 Performed tests

We performed two kinds of test, one involving the Request-Response paradigm for remote control and the other the Publish/Subscribe-Notify semantic for event management. The next section describes the deployed sensor network and the environment where the tests have been carried out. Then we focus on the remote control test, implemented via the SIP `MESSAGE` method. And finally we report on the event notification framework implemented via the `PUBLISH`, `SUBSCRIBE`, and `NOTIFY` methods. The messages exchanged on the various interfaces have been captured by means of the Wireshark analyzer [54] and synthetically reported in the form of diagrams.

2.6.1 Network topology

All tests have been carried out within the premises of the Department of Information Engineering of the University of Pisa, Italy. This might indeed constitute a good test environment, as applications such as smart energy, building automation, and intrusion detection systems fits well this kind of structures. The realized testbed is made of five ZigBee sensor nodes, including the ZigBee Coordinator, which is connected to the SHG via a USB cable, as already shown in Figure 3.2. The physical location of the nodes is illustrated in Figure 3.4. The ZC is named *Gateway*. All ZigBee nodes have a wireless path to the ZC. Due to the indoor environment, the nodes *Stairs*, *Office*, and *Corridor* use a multihop path.

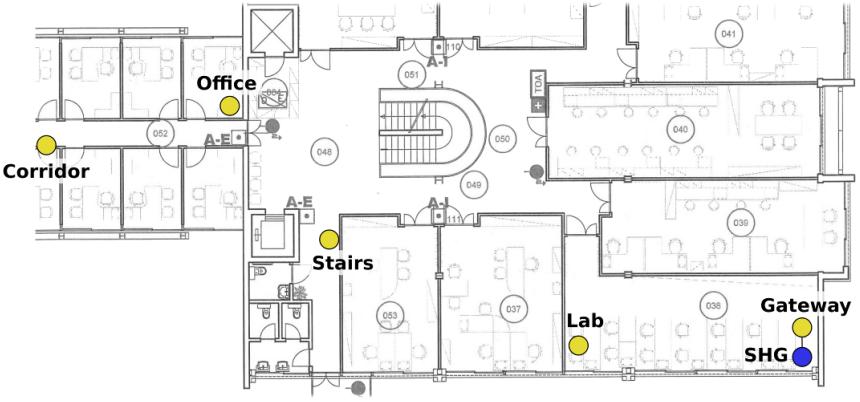

Figure 2.8: *Map of the Department of Information Engineering with the position of the ZigBee nodes (yellow discs); the SHG is also shown (blue disc).*

2.6.2 Remote control

The remote control test consists in a user controlling a light from his/her laptop or smartphone from a remote location. In this test, the Client can be either Legacy or Enhanced, since all operations are performed through the `MESSAGE` method. The user's terminal and the SHG are located on two different networks (though both belonging to our University), so that we can indeed test a control procedure from a remote location. For the registration procedures we have used the *iptel.org* standard SIP server.

The user can make use of a set of commands to pilot the light, as detailed in Table 2.1, and can address the end nodes with the name of the room where they are located. The

mnemonic names of the rooms are reported in Figure 3.4. The relationship between the SIP messages (expunged of the registration and authentication procedures, as well as of the `200 OK` answers), the SHG translations, and the ZigBee frames is schematically illustrated in Figure 2.9. The test can be split in three phases (1, 2, and 3), which are indicated on the left.

Table 2.1: *List of test commands.*

Command	Effect
switch_on_light	turn on the light (at the maximum level)
switch_off_light	turn off the light
set_light_level *N*	set the light level to *N*
subscribe_home_auto	subscribe the Home Automation events (in backward compatibility mode)

The first SIP messages of our test (not shown) are related to the SHG registration procedure with the *iptel.org* server. Once the registration is completed, the SHG is reachable through its mnemonic address (`sip.home.gate- way@iptel.org`). A similar operation has been performed by the Client too. The final Client URI is `lorisgazzarrini@iptel.org`.

Now, let us examine the user interaction (phase 1 at first). To control a specific device the user opens a chat window on his SIP Client, in this case Jitsi, and types one of the previously listed commands preceded by the room name. For example, assuming the user wishes to turn on the *Office* light at an intermediate level, the resulting text would be something like "Office set_light_level 2". Figure 2.10 shows the text messages that appear on the Client interface. The SIP UAC encapsulates this text into a `MESSAGE` and sends it to the SHG.

At the reception of the `MESSAGE`, the SHG performs a syntax check. Since the syntax is correct, a `200 OK` is sent back to the Client via the *iptel.org* server. Then, another verification is made to ensure that a valid command is received. Being the command and room correct, the SHG proceeds to transform the command into a DFA Request. This is in turn translated into a ZigBee command, containing the MAC address of the *Office* node, which is sent to the embedded ZigBee Coordinator. The ZC builds a proper ZigBee frame (the ZCL "command" frame shown in Figure 2.9) and casts it over the air. The specified sensor node is thus reached by the command, possibly through some intermediate router nodes. It verifies that the command is correct and executes the action (i.e. the node would change the light level).

Since no explicit acknowledgement is sent by the sensor node[4], the SHG proceeds to check

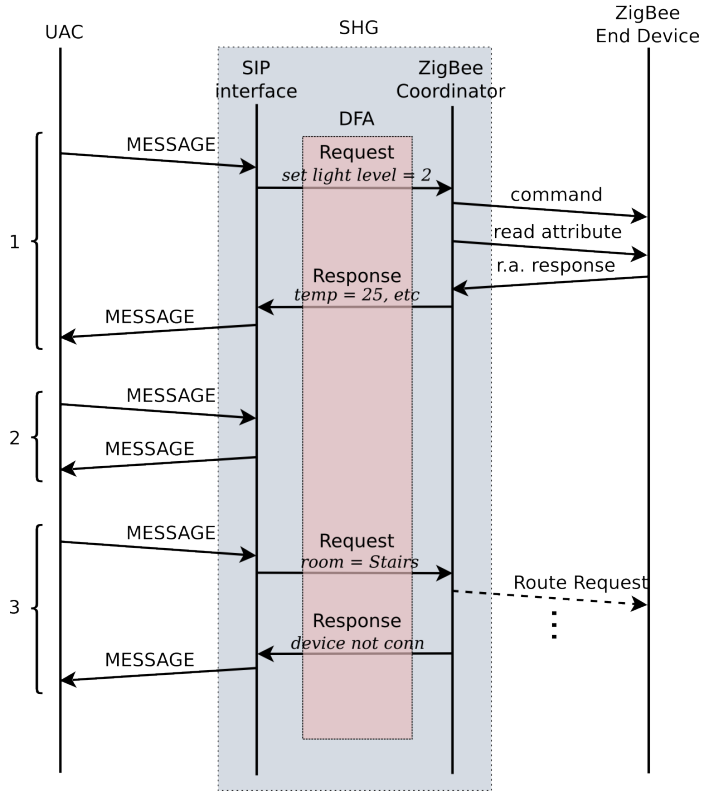

Figure 2.9: *Messages on the SHG and ZigBee domains exchanged during the remote control tests.*

for the correct execution of the command. It thus instructs the ZC to ask for an update of the sensor status, which, in our example, comprises the current light level, temperature, and pressure values. The ZigBee Coordinator builds and sends a ZCL "read attribute" frame to the sensor node, which replies with a ZCL "read attribute response" frame. The response frame is received by the ZigBee Coordinator, which updates the information stored in its local memory and also informs the SHG with a Response message. The SHG can thus build a new `MESSAGE` packet, with the new light level and sensor readings, and send it to the user. The user terminal will then display a confirmation message, as shown in Figure 2.10. In our case, the light has now reached the second light level that can be read in the fourth line of the chat screenshot.

[4] According to the ZCL specifications, the sensor is not required to confirm neither the reception nor the execution

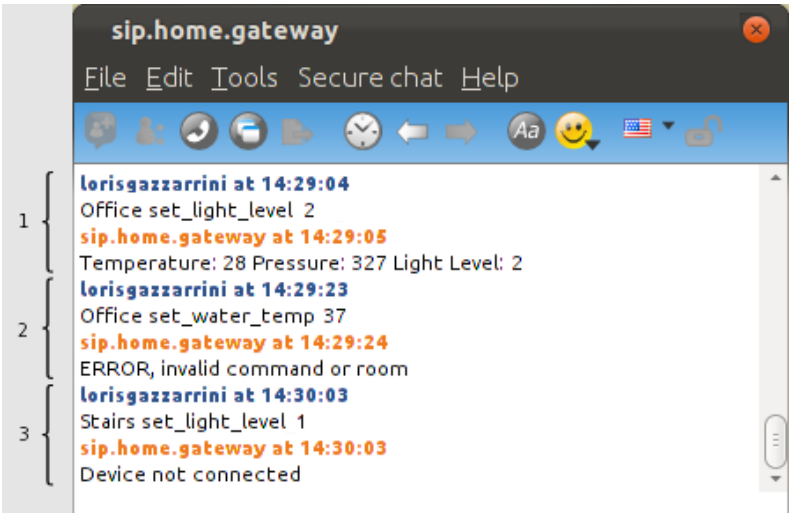

Figure 2.10: *Screenshot of the Jitsi chat window taken during the remote control test.*

To verify and give an example of the behaviour in case of a `MESSAGE` containing a wrong command, we let the user issue a non-existent command ("set_water_temp"). The SHG replies at first with a `200 OK`, because the syntax of the `MESSAGE` element is correct, but after parsing the message, it detects a wrong command, and therefore builds a `MESSAGE` to inform the user that the typed command is wrong. Note that no message translation occurs and no traffic is sent on the ZigBee network (as illustrated in Figure 2.9).

To complete the panorama of the possible cases, we report a test in which a user is trying to control a node that is not connected to the network[5]. Since the command is correct, the SHG passes the request to the DSAN. However, when the ZigBee Coordinator looks for the ZigBee network address of the device in its binding table, nothing is found. The ZC may then try to find this device, e.g. by means of a discovery procedure, and, after a specified number of failed attempts, an error message is sent from the ZC to the SHG and then from the SHG to the Client.

---

of the command. Note that a MAC layer acknowledgement frame is indeed sent back, but obvioulsy this does not carry any application layer information.

[5]We assume that the room name is correct, otherwise the SHG would immediately reply with an error message.

2.6.3 Event Notification

In this section we show how the domotics testbed works for the event notification paradigm. Since the test involves the use of the PUBLISH, SUBSCRIBE, and NOTIFY methods together with the newly defined Home Automation event package, we also employed the ENS. The "legacy" *iptel.org* server is still used as both registrar and proxy for the Clients.

The test consists in a remote user interested in following the variations of the ambient values of a specified room, in our case the *Lab*. As explained in Section 3.4, we have two different possibilities, depending on the client employed by the user (either Legacy or Enhanced). Both of them are shown in Figure 2.11, which reports the SIP transactions occurred during the test. To keep the figure clean, we do not show the acknowledgement messages, the Client registration procedures, and the ZigBee devices. The messages exchanged on this side of the system, however, are quite essential: every time an event is generated by a ZigBee node, a ZCL frame is sent to the ZC and hence to the SHG. The reception at the SHG is indicated by a red disc on the SHG line. In our testbed, events are generated on a regular time basis.

At startup (phase 0), the SHG registers with the ENS, so that it can publish the Home Automation events. A PUBLISH is sent immediately after the reception of the 200 OK message to perform a first update of the DSAN data. This is illustrated in Figure 2.11. Both clients register instead with the *iptel.org* server (not shown).

The interaction with the Enhanced Client embraces phases 1 and 2. When the user wants to control the variations in the ambient values, he/she types the mnemonic address of the device (i.e. the room name, e.g "Lab") and the duration of the subscription (e.g. 100 seconds). A subscription to the Home Automation event regarding the *Lab* room is requested to the ENS by the UAC. This message traverses the SIP network (dotted line in Figure 2.11) and reaches the ENS. This is an essential aspect of the test, because it confirms the compatibility of our proprietary Home Automation event package with the existing SIP infrastrucure.

If the command is correct, the server accepts the subscription and provides the client with the current status of the *Lab* readings via a NOTIFY message. A green shadow on the Enhanced Client line indicates that the notification service is active for this device. Therefore, when the ZigBee network sends an event to the SHG, this casts a PUBLISH to the ENS which in turn notifies the Enhanced Client with a NOTIFY message. A screenshot of our Enhanced Client is reported in Figure 2.12.

When the user wants to control the ambient values from a Legacy Client (in our testbed an iPhone featuring the Bria software), he/she must perform the subscription by typing a string

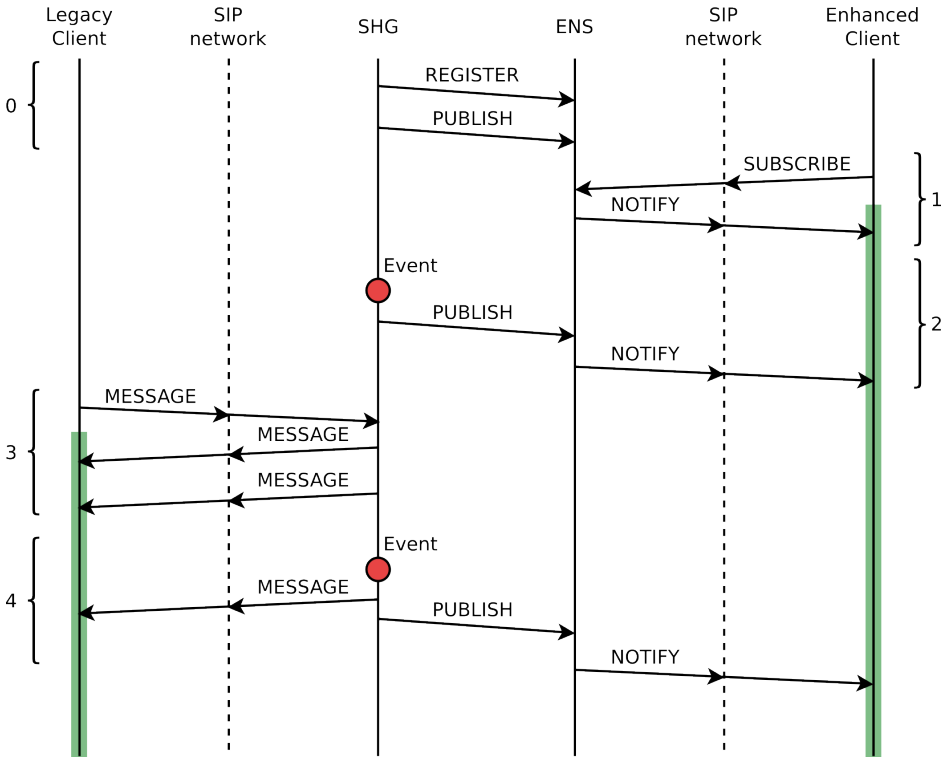

Figure 2.11: *Messages on the SIP domain exchanged during the event notification test.*

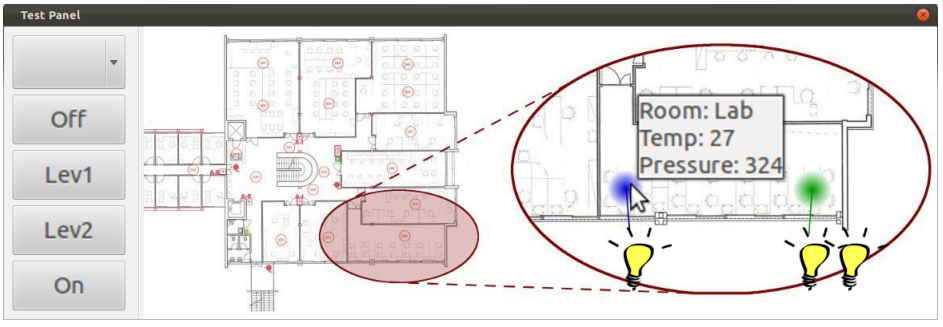

Figure 2.12: *Screenshot of the developed Enhanced Client window taken during the event notification test.*

of characters in the chat window. In our example, the "subscribe_home_auto" command, see Table 2.1, followed by the room name (e.g. Lab), and the duration (e.g. 100 seconds). The UAC then encapsulates this text into a `MESSAGE` and sends it to the SHG, possibly via the SIP infrastructure (phase 3). Indeed, from the analysis of the traces captured on the SIP network, we could see that the message reaches the SHG via *iptel.org* and the ENS. The mechanism is therefore transparent to both the existing and new SIP network elements.

If the command is correct, the SHG replies with another `MESSAGE` in which the user is notified of the acceptance of the subscription. A further `MESSAGE` with the observed values immediately follows (to mimic the `NOTIFY` message sent to the Enhanced Client). The user can now take advantage of the notification service managed directly by the SHG. Note that the notification setup procedure for the Legacy Client involves three transactions (in place of the two for the standard procedure). When an event occurs (phase 4), the SHG sends a `PUBLISH` to the ENS and also a `MESSAGE` to the Legacy Client. Note that all `MESSAGE` messages are sent from the SHG directly to the client. The outcome of the event notification service on the Bria chat window is shown in Figure 2.13 (where two event on two different rooms have been subscribed).

2.7 Background

As outlined in the Introduction, domotics may combine many different sub-systems and technologies, from the application to the physical layers. As a result, realizing a comprehensive survey of all the literature that somehow intersects the domotics scenario is indeed a huge task. In this Section we just draw a sketch on the current state of the art, whereas a comparison with the works whose topic is most similar to ours is presented in the next subsection. As a general taxonomy, we can distinguish works that dealt with sensor and actuator networks only, works that addressed the home automation paradigm, and works that tried to merge the various sub-systems under a common framework.

As for the Wireless Sensor Network (WSN) area, several papers have been presented that deal with making the information of a WSN remotely available and/or conveying commands or data to the WSN nodes from anywhere on the global network. Among the most interesting solutions proposed so far, two are worth mentioning: IrisNet and MQTT-S. IrisNet [28] is a software infrastructure that lets users query globally distributed collections of high-bit-rate sensors. The communication, however, is only unidirectional, i.e. data can only be retrieved from the sensors. MQTT-S [30] is an extension to the WSNs of the open publish/subscribe

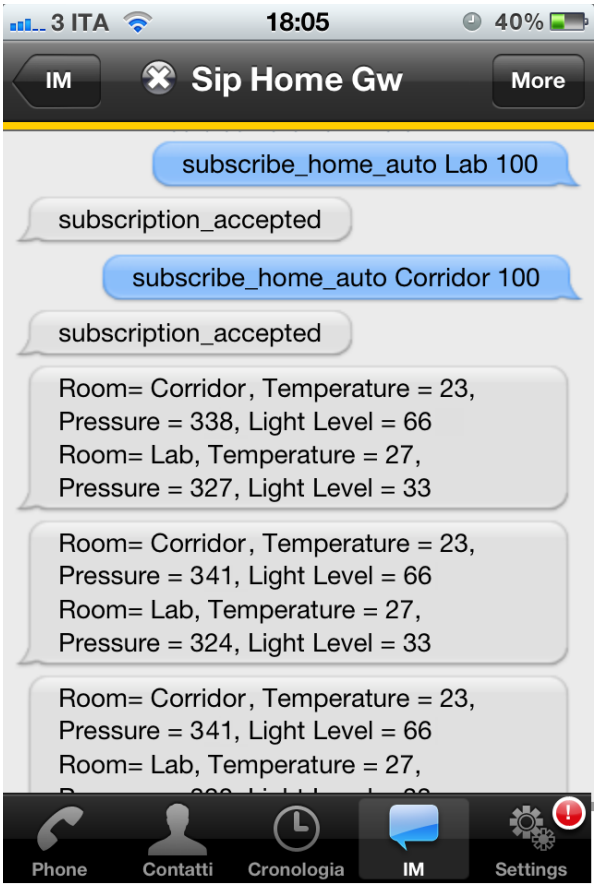

Figure 2.13: *Screenshot of the Bria chat window taken during the event notification test.*

Message Queuing Telemetry Transport protocol. The drawback of MQTT-S is that it relies on protocols and architectures that do not enable the interaction between different systems.

A more comprehensive approach has been proposed in [27], in which diverse WSN technologies are integrated into a special device called WSN-Center (WSN-C), in turn integrated into the Service Oriented Architecture (SOA) of the operator. In this way, the WSN-C provides, in the form of services, a set of application-independent functionalities, e.g. for the configuration, data acquisition, and management of logical names. Hence, the WSN-C plays an intermediary role between applications and the specific WSN technology. The authors adopt the Web Services paradigm based on the REpresentational State Transfer (REST)

protocol. From the perspective of the operator, this approach is the most suitable to the characteristics of the SOA, since it is able to traverse firewalls, is scalable, and minimizes the communication overhead. Examples of applications of this architecture are presented in [14].

REST has also been the tool for the authors of [40], who used it to provide clients connected to the Internet with the ability to directly interact with MICAz sensors using POST, GET, and SUBSCRIBE requests. However, an interface between the clients and the sensors is still necessary and is provided by an HTTP-2-TinyREST gateway. The gateway maps every HTTP request to a TinyREST message and forwards the message to the sensors. Therefore, the implementation of such system implies, besides the gateway development, also the onerous task of bringing REST compliant software on constrained-resource sensor nodes.

The authors of [31] and [32] propose TinySIP, an extension of the SIP for the WSN world. TinySIP supports the concept of session (to allow receiving a continuous video or audio stream), as well as the publish/subscribe and instant messaging approaches which are most renowned in the WSN domain. TinySIP offers to multiple clients the access to sensor-based information via SIP. The advantage is that the user interacts with a client he/she is already familiar with. Similarly to [40], this solution needs a gateway for mapping SIP messages to and from TinySIP. Moreover, a "zone manager" is necessary for sensor node registration.

On the side of home entertainment and content storage systems, despite their commercial growth and increasingly automated and simple mechanisms for content access, the heterogeneity of the different solutions that are on the market is still an open issue. Studies on how to design a platform for domestic services have been precisely focused on how to handle the heterogeneity of the devices, and how to integrate them to create value-added services and applications.

In this scenario, the need to put on the same platform classical communication services (e.g. telephony) and the new entertainment applications has resulted in the definition of solutions based on SIP, such as [33] and [16]. Though following the SOA paradigm, they lack the flexibility and openness towards other systems, like home automation and sensor platforms, thus making it impossible to develop new SIP services that integrate these worlds.

The authors of [47] showed how ubiquitous computing can be integrated into home networks by means of SIP. The choice of using SIP is based on the assumption that in the future every building will be equipped with a full featured IP network.

An approach that follows the philosophy of making an open and flexible service platform can be found in [10]. The proposed Device Integration Framework is based on the creation of a middleware with a two-levels architecture. In the lower level, the middleware provides a set

of functions that allow integrating the different technologies used in the devices that handle the multimedia services. In the higher level, a framework that interacts using SIP has been realized to allow connecting the various SIP and non-SIP components to create advanced services.

The authors of [9] presented a concept of a ubiquitous home and facility control solution exploiting the IP Multimedia Subsystem (IMS), a SIP-based control architecture considered by mobile network operators. In particular, SIP and the Presence Service can be used to realize a near-real-time push solution to manage the actuators and sensors via a mobile device, or to develop a communication platform for existing bus-systems for Smart Home Control (e.g. KNX).

More recently, SIP has been tested as a universal communication bus for home automation environments [12] [11]. A gateway and a series of SIP adapters and interpreters have been deployed to make all devices SIP compliant. An experimental platform was realized in order to estimate the adaptation work and to collect some figures on the performance of the gateway.

2.7.1 Comparison with our work

After having illustrated our domotics architecture and sketched an outline of the related literature, we can summarize the differences that make our contribution innovative.

The approaches that tried to bring customized or reduced versions of SIP [31] or REST [40] on the sensor nodes suffer from a series of drawbacks. Since the device are resource constrained, the protocols must be stripped of many functionalities. Due to the particular operative system running on the sensor nodes, the development times might be non-negligible. Also, given the high heterogeneity of the devices, it might be necessary to repeat and modify the customization and development steps for every technology that is going to be integrated into the system. Finally, compatibility with deployed hardware and software is not retained. Conversely, our framework moves the development effort onto a single high-end device (the SHG), allowing faster implementation times and full compatibility with both existing sensor and actuator devices and also with the SIP protocol.

The greatest disadvantage of proposals like [9], [10], and [12] is the necessity for all home devices to register with their own URI. When the number of devices increase (heavily monitored and automated buildings may have hundreds of nodes), the user capability of handling them all through their URIs is clearly hampered. The same shortcoming applies to the zone manager solution proposed by [31], in which the majority of the communications are possible

only by knowing the address of each gateway to which the sensors of interest refer. Under this constraint, it is not possible to change the gateway-sensor association online. In other terms, these approaches lacks scalability and/or flexibility. On the other hand, in our system the sole SHG must register to an external SIP server (unless the SHG itself implements a registrar) and we can mask the multitude of DSAN nodes by means of the "functional addressing" method.

Alia *et al.* started their work from an architecture [10] similar to ours. However, they followed a top-down approach with main focus on the software and middleware aspects of the system, whereas we cared more about the integration at the functional level. Also, their scope was primarily the multimedia home entartainement area, with the consequence that their approach cannot be esily extended to other domotics sub-sytems. In this respect, our architecture covers a much wider panorama. Furthermore, the two-level architecture of Alia *et al.* implies a rather complex set of elements and interactions, whereas our system features only the DFA-based translation, which is a much thinner layer.

The work closest to ours is perhaps the one by Bertran *et al.* [12] [11], who also implemented a SIP gateway between a SIP-native world and a set of heterogeneous home automation systems. However, there are some aspects that may put our framework one step ahead. Bertran *et al.* did not consider the issues with addressing and reachability of the single DSAN devices. Conversely, we designed a functional addressing scheme that greatly simplifies the user interaction and does not require the DSAN nodes to register to any SIP server or other additional entities. Then, we devised a way of keeping the compatibility not only with the DSAN elements, but also with the user terminals. This allowed us to provide the user also with fucntionalities that are not natively supported by his/her device. This paradigm can even be extended to ensure forward compatibility with new domotics services. Conversely, Bertran *et al.* did not pay much attention to this seemingly secondary, but in our opinion quite substantial, aspect. A third distinguishing point is in the adaptation between the SIP and the DSAN worlds. While Bertran *et al.* design a single software module to be put in the gateway, we perform this operation in two steps, via the DFA layer. This allows to decouple the implementation of the two domains, making the system more flexible. Finally, we studied in much more detail the integration with a possible DSAN, namely ZigBee, and showed how it is possible to exploit its features to simplify the integration into the system. In [11], the main focus of the experimental platform was on the performance figures of the gateway (which, if we think about the current hardware technology, might not be the most relevant hurdle to the domotics development).

A further advantage of our solution is that the SHG can be conveniently merged with the ZigBee Gateway, a device recently standardized by the ZigBee Alliance to enable the interaction of the ZigBee WSN with IP networks using REST and SOAP protocols [56]. Enhancing the ZigBee Gateway with SIP features is therefore a conceptually easy operation that would offer the user a larger set of interfaces (REST, SOAP, and SIP) and the properties of SIP: real time interaction with the end devices and a large and fully functional infrastructure. The fact that our idea moves in the same direction of the standardization bodies directly involved in the home and building automation arena might indeed be a confirmation of its validity.

2.8 Conclusions

The chapter presented an architecture for realizing a domotics system with heterogeneous devices and user terminals. The architecture is based on the use of SIP as the common control plane and is centered on the SIP-based Home Gateway. In our vision, the proposed architecture has several aspects that might contribute in advancing the current state of the art in domotics systems.

A functional addressing scheme and an abstract translation layer (called DFA) are used to make the underlying technology transparent to the user. In addition, by exposing a single SIP URI to the user, the system can be easily extended to large deployments. Heterogeneous sensor and actuators devices (DSANs) are supported through different translation entities embedded into the SHG. The DFA is the glue between these entities and the SIP world, and simultaneously allows to separate the implementation of the SIP and DSAN interfaces. Compatibility with existing SIP devices and network elements have also been guaranteed.

We have proved the feasibility of our architecture by means of a working prototype, which includes the SHG, a standard ZigBee network, an Enhanced and a Legacy Client, a newly defined SIP event package, and a customized Event State Compositor. The possible synergies between ZigBee and SIP have also been illustrated.

CHAPTER 3

EXPERIMENTAL EVALUATION OF THE SIP-BASED HOME GATEWAY

In modern houses, the presence of sensors and actuators is increasing, while *communication services* and *entertainement systems* had long since settled into everyday life. The utilization of wireless communication technologies, such as ZigBee, Wi-Fi and Bluetooth, is attractive because of their short installation times and low costs. The research is moving towards to the integration of the various home appliances and devices into a single domotics system, able to exploit the cooperation among the diverse subsystems and offer the end-user a single multi-service platform. In this scenario, the chapter presents the experimental evaluation of a domotics framework centered on a SIP-based Home Gateway (SHG). While SIP is used to build a common control plane, the SHG is in charge of translating the user commands from and to the specific domotics languages. The analysis has been devoted to assess both the performance of the SHG software framework, and the negative effects produced by the simultaneous interference among the three widespread wireless technologies.

3.1 Introduction

Domotics refers to a system that control several (or all) home "services", such as lighting, HVAC (heating, ventilation, and air conditioning), communications, security, healthcare, and entertainment, in a integrated and automatic or semi-automatic way, allowing the user to manage them from a series of heterogeneous devices (e.g. touch panels, remotes, mobile handsets, smartphones), either at home or from anywhere in the world. In the domotics archetype, all subsystems are able to talk to each other and interact in a seamless manner,

realizing an intelligent structure that improves the quality of life, reduces the costs, and achieves energy savings. To put this paradigm into practice, the communication among the single devices and between the various subsystems is the fundamental operation. Hence, wired and wireless networks will be one of the building blocks of the present and future domotics solutions. On top of this somewhat "physical" element, a common control plane is also necessary, in order to unify the management operations into a single and portable user interface.

One of the major components of a domotics system is the set of sensors and actuators. These usually come in the form of one or more networks, backed either by a single technology or by differerent ones. Not always, however, do the specifications define a common control plane that is suitable to contemporarily manage devices belonging not only to different standards, but even to different application profiles. As a result, the burden of coordinating and making devices interoperate is often left entirely to the system implementer. Indeed, a scenario with mixed profiles and technologies is not so uncommon, especially in those environments where multiple services might be requested. One such example is exactly the "smart home" or domotics concept, in which several profiles and technologies (e.g. ZigBee's Home Automation, Smart Energy, and Telecom Services, or KNX's Lighting, Heating, Energy management – just to cite the most appealing ones) might all be present.

From the user perspective, the devices belonging to the diverse subsystems of the home services platform can be typically controlled through dedicated appliances located in the house (e.g. a touch panel, a smart telephone, a TV remote). However, this paradigm no longer holds for remote control operations that occur when the user is far from home. In this case the user would normally have a single device at hand, such as a notebook or a smartphone, by means of which he/she would like to control any device in the home, not just those belonging to a specific profile or technology, and possibly without complex configuration or selection procedures.

In addition to the need for a coordinating system for the DSANs, in today's houses we already find interpersonal communication and multimedia entertainments systems. Hence, the design of a domotics platform should also consider the integration of communication and multimedia applications with the DSANs-based services.

Among the various domotics sensor and actuator networks (DSANs), wireless sensor[1] networks (WSNs) are the version that is growing faster, due to shorter deploying times and simplified configuration. Several technologies and standards are nowadays available for the implemen-

tation of a WSNs [50]. Especially the ones based on open or widely adopted standards, such as ZigBee, Bluetooth, Z-Wave, and KNX-RF, can undeniably be regarded as the most interesting ones. This is beacause they allow the deployment of large and almost self-configuring networks in relatively short times and at reduced costs. Two of these standards, ZigBee and Bluetooth, have been embedded into the SHG.

On the other hand, the current trend in multimedia and communication home systems is to move the physical transport services over the Wi-Fi technology. Thus, we have equipped our SHG also with a Wi-Fi interface, used for providing the above-mentioned "wideband" services.

The majority of these wireless standards operate into the unlicensed 2.4 GHz ISM band, which can be exploited by multiple users and networks at the same time. However, due to the mutual interference, the coexistence of different devices operating in proximity of each other can be troublesome. As proved by many authors [41, 49], this is especially true for ZigBee networks, whose performance is heavily influenced by the presence of Wi-Fi devices. While it is sometimes feasible to avoid the interference among devices sharing the same spectrum and implementing the same standard (e.g. collision avoidance schemes might work across separate networks), the use of incompatible modulations and channel access schemes makes it virtually impossible to ensure the coexistence among devices belonging to different technologies.

In summary, the design and implementation of a domotics gateway must face two key issues: integrating heterogeneous indoor devices and networks, allowing the composition of dynamic and pervasive services (including interpersonal communications and multimedia), and assuring the physical coexistence of the interfaces located on the gateway apparatus.

3.1.1 Contribution

A customized SIP event package and a notification server have also been developed to validate a possible extension to new services. The SHG was interfaced with an actual ZigBee network and a Bluetooth PAN, in addition to a generic Wi-Fi connection. We experimentally evaluated the performance of the SHG prototype, proving its ability to support large domotics systems.

The chapter reports an experimental study involving Wi-Fi, ZigBee, and Bluetooth networks. The goal of this study is to characterize the performance of the SHG in terms of the

[1] Note that the term "sensor" is often used for both sensors in the strict sense and for actuators too.

coexistence of the three systems, especially because they are all active in the same time and space, i.e. in the prototype SHG board, and thus subject to strong mutual interference.

3.2 Background

For the coexistence of multiple wireless interfaces, we can find numerous analytical and simulation studies, especially about the performance of ZigBee under the interference of Wi-Fi and Bluetooth (such as [48], just to cite one). The major shortcoming of these approaches is that, due to the very complex nature of the wireless channel and environment, there is no measure of their agreement with the reality, and thus their actual utility is somehow limited.

Sikora and Groza experimentally obtained the PER of a ZigBee system under the interference of Wi-Fi devices, Bluetooth devices, and also of a microwave oven [49]. However, the study is limited to a single source of interference (e.g. either Wi-Fi or Bluetooth), and also the analysis of the coexistence of ZigBee and Bluetooth is not complete, since the (actually very few) results have been collected in one direction only (i.e. Bluetooth over ZigBee). Neverhteless, an interesting observation in Sikora and Groza's paper is about the presence of notable discrepancies between the collected experimental data and the simulation results provided by the IEEE 802.15.4 task group.

A similar experimental study was led by Musăloiu and Terzis, who evaluated the loss rate of a ZigBee system under Wi-Fi interference [41]. Starting from this result, they developed interference estimators and distributed algorithms to dynamically change the ZigBee operating channel. This approach was proved to drastically reduce the loss rate of ZigBee networks.

The authors of [29] present the results of an empirical study on the coexistence between IEEE 802.11b and Bluetooth devices. However, the primary objective was to develop an analytical model to estimate the mutual interference, rather than characterizing it in real world scenarios. Hence, to build such models, the experiments were controlled through the use of attenuators, signal generators, and coaxial cables, thus resulting in a rather idealistic environment.

From the analysis of the cited works, it emerges that in all cases, even in [49], the authors studied the interference of no more than two systems at a time. A two-way experimental analysis of the simultaneous interference among Wi-Fi, Bluetooth, and ZigBee can be found in [26], which confirms the weakness of ZigBee and also shows that some supposed interference-free ZigBee channels are in fact affected by the presence of Wi-Fi transmissions.

However, in all cited works, the interfering sources are always placed in physically disjoined

devices. On the contrary, devices such as the domotics gateways are expected to embed several wireless interfaces onto the same board. In such cases, the interference effect might be even greater, due to the electrical couplings on the board. The experimental measurement we carried out over our prototype SHG was aimed at filling this gap.

3.3 Wireless technologies in the SHG

In this section we give a quick overview of the three wireless technologies, i.e. Wi-Fi, ZigBee, and Bluetooth, that we have selected for the Home Network and hence integrated onto the SHG prototype board. We also outline how these standards exploit the 2.4 GHz band and interact in this region of the spectrum.

3.3.1 Wi-Fi

The latest IEEE 802.11 standard [8] defines a CSMA/CA (carrier sense multiple access with collision avoidance) scheme as the mandatory medium access scheme. According to CSMA/CA, every Wi-Fi device shall listen to the medium before transmitting. The transmission is allowed only if the medium has been sensed idle for a pre-defined time period. In case the medium is sensed busy, or after a collision, the device shall refrain from transmission for a period whose length is determined by a random variable (exponential backoff).

A IEEE 802.11 network can operate over one of the 11, 13, or 14 channels defined for the 2.4 GHz ISM band (the exact number depends on the local regulations). Each channel is 22 MHz wide, and the channels are partially overlapped (since the overall ISM bandwidth is just above 80 MHz). Therefore, no more than three networks can be contemporaneously operated in the same area in order to keep the transmissions of each free from interference from the others.

The operative channel and the transmission power are generally set statically (e.g. by the manufacturer or by the user at configuration time), even though dynamic channel selection (DCS) and transmit power control (TPC) routines have been defined for operations in the 5 GHz band. In the 2.4 GHz band the maximum transmission power is 100 mW (20 dBm) in Europe, and 1 W (30 dBm) in North America; in Japan, where power is measured in relationship to bandwidth, the maximum allowed power is 10 mW/MHz.

Finally, the modulation scheme is either a DSSS (direct sequence spread spectrum) for the lower bit rates, or an OFDM (orthogonal frequency division multiplexing) for the higher ones.

3.3.2 ZigBee

The IEEE 802.15.4 standard [7] specifies the physical and medium access control layers for low-rate wireless PANs, targeting a 10 meter communication range with a transfer rate of up to 250 kb/s.

Similarly to Wi-Fi, 802.15.4 devices employ a CSMA/CA channel access algorithm and the DSSS modulation (actually, the latest release of the standard defines four modulation schemes, but in the 2.4 GHz band only the DSSS modulation is allowed).

Sixteen channels are defined for worldwide use in the 2.4 GHz band. However, differently from 802.11, they are much narrower (just 2 MHz) and do not overlap, so that up to sixteen 802.15.4 networks can easily coexist in the same area. When starting a new network, an Energy Detection (ED) functionality is used to determine the activity of other systems and thus decide the operating channel; yet there is no support for dynamic channel selection.

The latest ZigBee release has introduced the support for frequency hopping in the ”ZigBee Pro” standard. In this way a PAN coordinator can move the whole PAN to another channel if the one in use is overloaded. However this is not a fast, reliable, and energy saving way to solve the problem. In addition it is not mandatory to implement.

3.3.3 Bluetooth

Bluetooth is a standard communication protocol designed for connection-oriented services such as voice, with low power consumption and short range operations. The output power depend on the device Class, spanning from 1 to 100 mW. Accordingly, the expected range should go from 1 to 100 meters, even though the practical range is highly variable.

Bluetooth transmits on up to 79 channels in the 2402-2480 MHz range. Each channel is 1 MHz wide, and one guard channel is used at the lower and upper band edges. In order to reduce the interference from external sources, frequency hopping (FHSS) is used to spread the signal across all channels. Thus a single Bluetooth network uses the full available 2.4 GHz ISM band. Different networks can coexist in the same area by employing different hopping patterns or a time-shifted version of the same pattern. Since specification v1.2, Bluetooth also includes an adaptive frequency hopping (AFH) scheme, which reduces the number of employed channels to improve its robustness against the interference.

The Bluetooth channel access procedure is based on a master-slave scheme, which is built on top of a time division duplex (TDD) transmission scheme. The basic modulation is Gaussian frequency-shift keying (GFSK), which allows a transfer rate of up to 1 Mb/s. Since

the introduction of the enhanced data rate (EDR) with specification v2.0, π/4-DQPSK (differential quadrature phase shift keying) and 8-DPSK modulations may also be used, bringing the data rate to 2 and 3 Mb/s respectively.

3.3.4 Channels, Frequencies and Modulations

Figure 3.1 shows the allocation of the ZigBee and Wi-Fi channels over the 2.4 GHz ISM band. Note that a single 802.11 channel completely overlaps with four ZigBee channels. Bluetooth channels are not reported, as the FHSS covers the whole available spectrum.

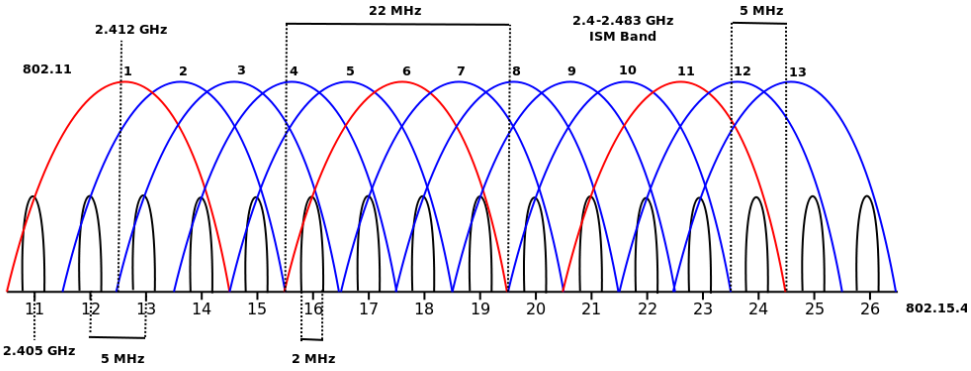

Figure 3.1: *Channel Occupancy of 802.11 and 802.15.4 systems.*

The three most used non-overlapping Wi-Fi channels are 1, 6, and 11. In this case, two ZigBee channels should be free from interference from Wi-Fi transmissions, i.e. channels 25 and 26 (the two rightmost ones). However, there is no assurance that using channels 25 and 26 solves the interference problem. For example, two channels might not be enough to allow the coexistence among several geographically overlapping PANs. In addition, though in North America ZigBee channels 25 and 26 can be really assumed free from Wi-Fi transmissions, in other regions such as Europe and Asia all Wi-Fi channels can be used, thus covering the complete set of ZigBee channels.

A further aspect making the coexistence of Wi-Fi and ZigBee difficult is the different allowed transmission power. In fact, the maximum Wi-Fi output power can be up to 100 times higher than the maximum allowed ZigBee transmission power (100 mW vs. 1 mW). The same consideration holds for Wi-Fi and Bluetooth devices belonging to Classes 2 and 3.

3.4 Proof of Concept

To put the ideas expressed in the previous Sections into practice, we have realized a small testbed involving all the elements of the architecture. The SHG, being the core and most innovative element, has been built from scratch. Two DSANs have been implemented using two sets of ZigBee and Bluetooth devices. Finally, to illustrate the potentials of expansion and customization of our architecture, we have defined and implemented the "Home Automation" package, a specific SIP event package for the domotics framework.

3.4.1 SIP-based Home Gateway

The only requirements for building the SHG are the sufficient processing power and memory to run the software, and the capability to interface with the technologies of the particular sensor networks to control.

With regard to the former aspect, we used a generic single board computer (SBC) with a Texas Instrument AM 3730 processor (ARM Cortex-A8) running at 720 MHz with 256 MB of DRAM and 256 MB of NAND flash memory. As it will be shown in Section 3.5.2, this hardware is more than adequate. To give the SHG the physical interfaces towards the wireless networks, a ZigBee module has been embedded into the board and connected to the main processor via a serial interface; then a Hama Bluetooth adapter and a Wi-Fi card were inserted into the two USB ports. Figure 3.2 shows the prototype SHG.

The SHG software was built on top of Linux (with kernel 2.6.36), which provides the necessary support and development tools (e.g. a SIP library, the interface drivers). The software that implements the SHG functionalities has been written from scratch using the C++ language, and then cross-compiled for the ARM platform. A multi-threaded approach has been followed. Each user request is handled in parallel by a different thread. This helps improving the scalability performance of the SHG.

The internal software architecture of the SHG is reported in Figure 3.3. Starting from the top, the first object we meet is the SIP interface. This is nothing more than the SIP software (the GNU oSIP and eXosip libraries), which extracts the user's commands from the SIP messages and passes them to the next module in the form of plain text strings. These are then translated into the proper DFA actions by the translation module, which fetches the set of available DFA actions from the DFA Library. The output of this module is fed to the SHG engine, where we have placed the intelligence for executing the user's directives in the proper way. This is tipically achieved via the creation of a series of elementary DSAN commands

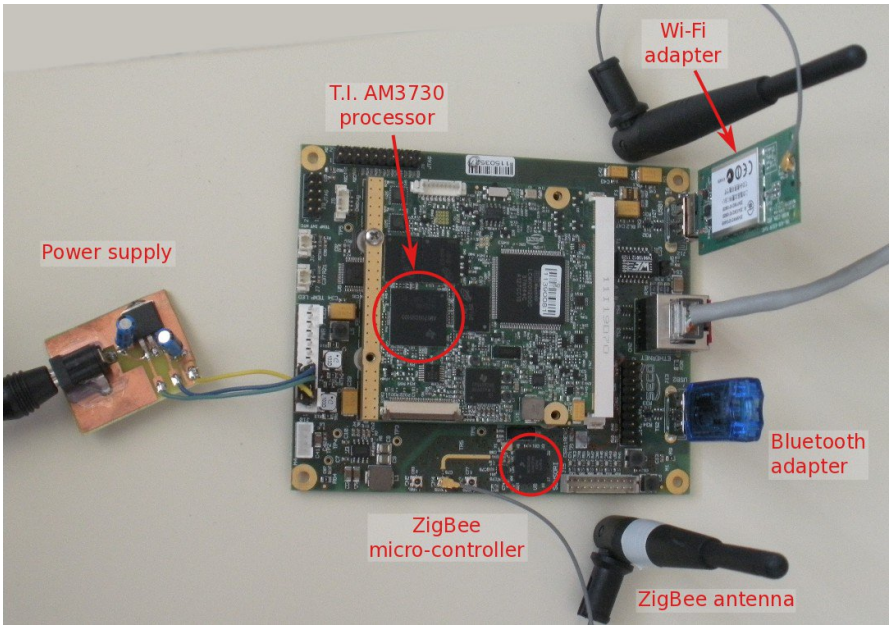

Figure 3.2: *A photo of the SHG prototype, with the indication of the main components.*

to be delivered to the various DSAN elements. The set of available elements and commands is retrieved from the device database. The SHG engine then passes the DSAN commands to the various DSAN managers, which are in charge of translating them into the technology-specific commands and performing all the operations to ensure that the specified actions are fulfilled. Finally, the ZigBee, Bluetooth, and Wi-Fi interfaces are the software modules (a custom software for ZigBee, the BlueZ stack for Bluetooth, the Linux drivers and tools for Wi-Fi) that pilot the physical objects that are connected to the various sensors and actuators.

The device database (DdB) holds the set of available DSAN objects, with the related properties (e.g. commands, location, technology). The DdB is filled and kept up to date by the DSAN managers, which are aware of the number and types of devices connected through the various DSAN interfaces. Further information, such as the physical location of each device, can be inserted at configuration time either by the user or by the service provider.

The operations in the reverse direction, i.e. from the DSAN networks to the SIP interface, are analogous to the ones mentioned above. The notifications from the sensors are passed, by means of the DSAN managers, to the SHG engine, which decides what actions are to be

taken. For example, a new command might be issued towards the DSAN, or an information message can be sent to the user (or both). In the latter case, the message is passed to the translation module, and finally to the SIP interface.

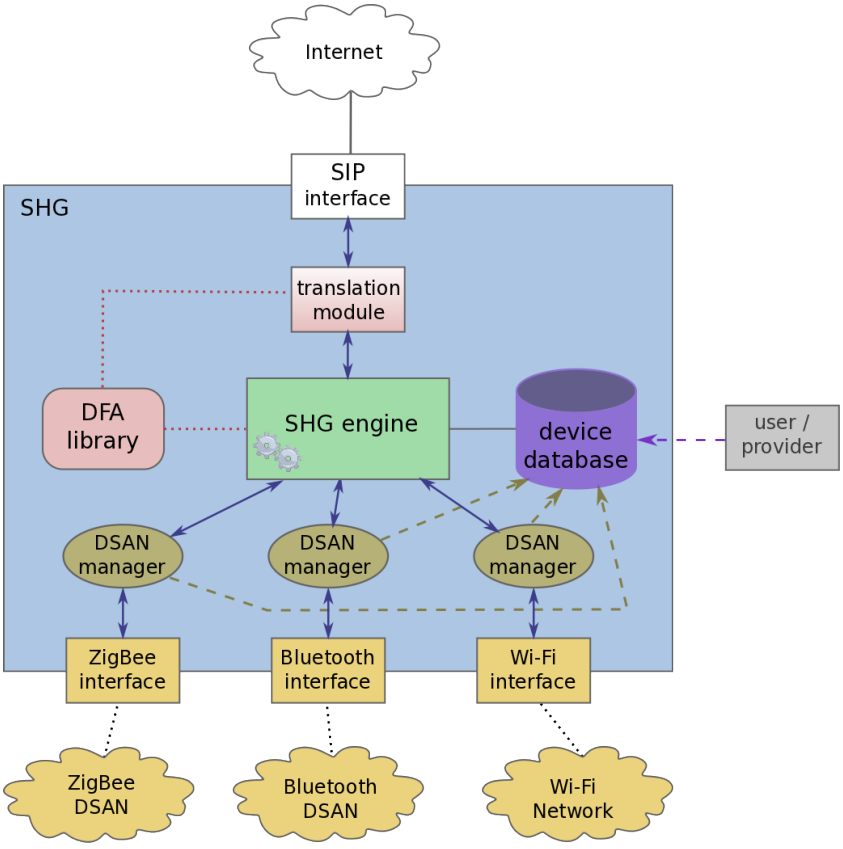

Figure 3.3: *The software modules building the prototype SHG.*

3.4.2 The wireless networks

This section briefly describes the set up of the two DSANs and of the Wi-Fi local area network. A few essential technical details are also given.

The ZigBee DSAN

The nodes of the ZigBee sensor network are based on the Freescale MC1322x board, which integrates a 32-bit ARM-7 MCU and a low power 2.4 GHz transceiver. The fully compliant ZigBee stack provided by Freescale was installed on the nodes.

A custom application that supports environmental data collection (temperature and pressure), remote light control, and message routing has been developed on top of the ZigBee stack by means of the ZigBee Cluster Library (ZCL) functions. The APS ACK feature (an end-to-end acknowledgment mechanism) was enabled to make the ZigBee transmissions reliable.

Ambient data is retrieved both on regular time basis and on-demand, and both approaches are available to the user, who can either subscribe to this event or ask the SHG to check a specific sensor value. As for remote light control, the MC1322x boards are equipped with an array of LEDs, which was used to mimic a multi-level light. For both ambient data collection and light control we defined a set of textual commands. Combining them with the name of a room allows the user to set the desired light level or retrieve the sensor reading.

An important aspect of the ZigBee system is that it provides for a mechanism, known as *binding*, to connect endpoints[2]on different nodes. Binding creates logical links between endpoints and maintains this information in a binding table. The binding table also has information about the services offered by the devices on the network. The ZigBee Coordinator (ZC) typically holds the binding table for the whole network. A notable advantage of this structure is that it allows the implementation of the *service discovery* procedure via bindings. The services available inside the ZigBee network can thus be discovered directly within the ZigBee domain, without resorting to any additional software or external entities. With specific reference to the SIP control plane, this means that there is no need to port the SIP registration procedure to the ZigBee network, since this would be a duplication of the ZigBee service discovery.

The Bluetooth DSAN

The Hama Bluetooth adapter connected to the SHG board embeds a version 2.0 compliant chipset supporting the EDR feature. It is a Class 2 device, with 2.5 mW (4 dBm) of output power allowing for an approximate range of 10 meters and a physical bit rate of 3 Mbps.

[2] An "endpoint" is a logical wire connecting distributed applications residing on different nodes.

To operate this device we took advantage of the Bluetooth Linux stack (BlueZ), which gives support for basic operations such as scanning and pairing. On top of these basic functionalities we built the Bluetooth interface, which is capable of listing and managing the connected devices.

For the purposes of validating the domotics system, we set up an audio streaming test. A Bluetooth-enabled headset (Sony DR-BT101) was used as the client device. The audio streaming was handled directly by the BlueZ (on the SHG side) and the headset, by means of the A2DP profile.

On the SIP-based control plane, we defined and implemented some simple commands, such as listing of the available content and playing an audio stream on the Bluetooth headset.

The Wi-Fi local area network

To build the Wi-Fi LAN we used two adapters based on the Ralink RT3572 chipset, a IEEE 802.11a/b/g/n compliant card. One of the adapters was installed on the SHG and the other on a common laptop PC. The drivers from the latest "compat-wireless" package have been uset to pilot the card.

We set up a private IBSS network on one of the 2.4 GHz channels. The use of an IBSS topology rather than an "infrastructure" one is justified by the shorter setup times (mostly in terms of driver and software configuration), but has no impact neither on the traffic at the transport and application layers nor on the physical layer mechanisms. The SHG and the PC are therefore two "peer" stations.

Traffic on the Wi-Fi connection has been generated by means of common test applications, such as FTP or iperf[3].

3.5 Performed tests

The performed tests can be divided in two sets. The first series was aimed at assessing the performance of the prototype SHG in terms of capacity, scalability, and processing delay. The objective of the second set instead was to verify the amount of interference among the wireless interfaces on board the SHG, and the impact on the SHG performance.

Before discussing the tests, we outline the deployed networks and the environment where the tests have been carried out.

[3] http://sourceforge.net/projects/iperf/

3.5.1 Network topology

All tests have been carried out within the premises of the Department of Information Engineering of the University of Pisa, Italy. This might indeed constitute a good environment for both kinds of tests: applications such as smart energy, building automation, and intrusion detection systems fits well this kind of structures; and we might indeed expect to find in the Department several devices using different radio technologies working at the same time.

The realized testbed is made of five ZigBee sensor nodes, including the ZigBee Coordinator (ZC), which is embedded on the SHG board as already shown in Figure 3.2. The physical location of the nodes is illustrated in Figure 3.4. All ZigBee nodes have a wireless path to the ZC. Due to the indoor environment, the nodes *Stairs*, *Office*, and *Corridor* use a multihop path. The Bluetooth headphones (*bths*) are placed in the same room of the SHG, approximately 8 meters apart; the PC acting as a Wi-Fi station (*sta1*) is placed in a room adjacent to the one with the SHG. We checked that the Bluetooth and Wi-Fi devices, as well as the *Lab* node, are within the operation range of the SHG. *sta0* represents the Wi-Fi adapter connected to the USB port of the SHG.

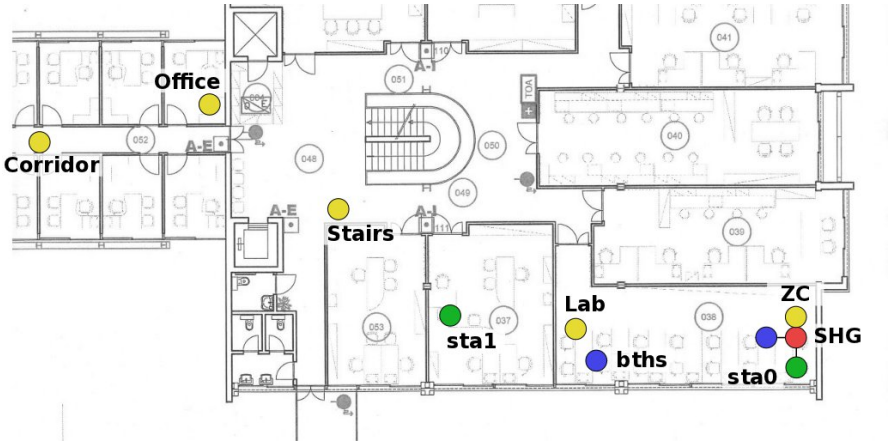

Figure 3.4: *Map of the Department of Information Engineering with the position of the ZigBee, Bluetooth, and Wi-Fi nodes (yellow, blue, and green discs, respectively); the SHG is also shown (red disc).*

3.5.2 Performance of the SHG

We assessed the performance of the SHG in terms of two metrics: the number of served user requests per second (in short: SURPS) and the average response time.

To compute the first metric, we connected the SHG to a varying number of clients through our 100 Mbps local area network. Every client was programmed to send a continuous flow of 100 requests using the `MESSAGE` method. Each request is cast as soon as the previous response is received from the SHG (we recall that a response is implemented with a distinct `MESSAGE` transaction). In this way the SHG always has a pending request to process for each client. The auxiliary SIP procedures, such as registration, have been excluded in order to measure the raw SHG capacity. For the same reason, we did not connect the SHG to any real DSAN, but implemented a fake interface that returns a response as soon as it receives a command. In practice, with reference to Figure 3.3, the processing path stops at the ZigBee interface. The Bluetooth and Wi-Fi networks were left inactive.

The collected numbers of total SURPS and mean SURPS per client, averaged over ten experiments, are reported in Figure 3.5 as a function of the number of connected clients. Focusing on the red lines (labelled "eXosip"), we can see that when a single client is connected, this can enjoy a service rate that is around 89 requests per second. This number can undoubtedly be deemed adequate not only for any human-based activity but also for any sensible automated application (see e.g. [11]). In case of two clients, the number of total SURPS is almost doubled, but when further clients are added there is a sudden performance drop. When four clients are connected the total SURPS are even less than the single client case.

We have explored the reasons for such a tremendous degradation, and found that it was due to the eXosip library integrated into the SHG. Without delving into the software details, this library presents some structures and timeouts that slow down the entire system when eXosip is called to serve many requests at the same time. We thus devised a simple patch that bypasses these shortcomings and repeated the SURPS test.

The results for the amended version are also shown in Figure 3.5 (the blue lines, labelled "patched"). The performance of the SHG has improved for almost any number of connected clients (it has slightly worsened for the two-client case only). More remarkable, however, is the fact that the trend in the number of SURPS is now much smoother, and, above all, that the total number of SURPS reaches a stable level – it floors to about 146 SURPS. This means that the performance of the SHG is not appreciably influenced by the number of clients. Hence

we can reasonably affirm that the SHG can scale to serve many requests from different clients at the same time.

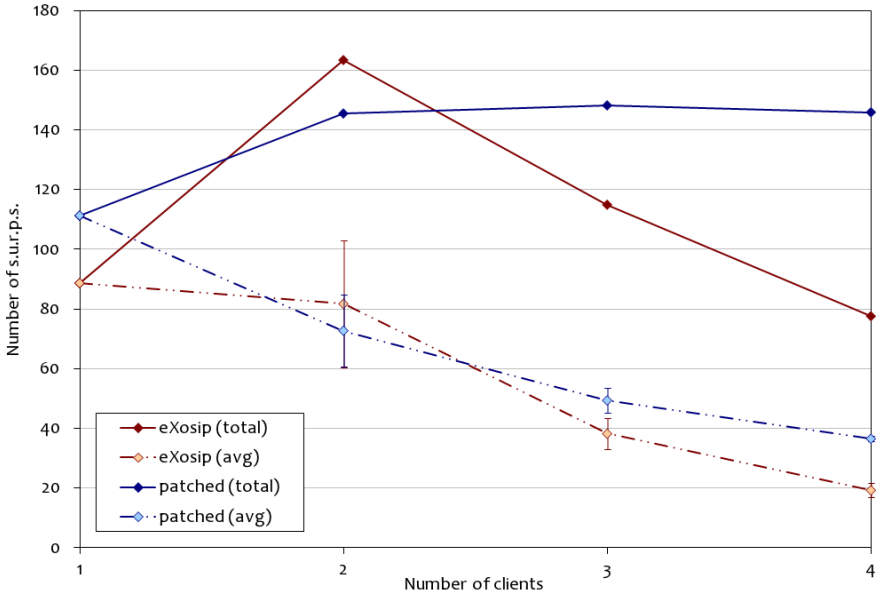

Figure 3.5: *Total and average number of user requests per second served by the SHG; the standard deviation among the clients is also reported.*

With the second performance test we analyzed the behavior of the SHG from an internal point of view. We measured the time that elapses between the reception of a request `MESSAGE` and the issue of the response `MESSAGE`. The measurement was carried out with a single client, with the auxiliary SIP procedures at work, but still with the Bluetooth and Wi-Fi networks kept idle. We used two configurations. The first one is still based on the fake interface, whereas the second setting is an operative scenario with a real ZigBee DSAN attached. To keep the things simple, however, the ZigBee DSAN is composed of two nodes only: the coordinator (physically soldered to the SHG board) and a device node in direct communication range. The ZigBee network operated on channel 25, which is the most free from external interference sources.

The first row of Table 3.1 shows the processing times of the SHG in the two configurations. The difference is apparent, being them apart by almost two orders of magnitude. This result does not leave much room for speculation, and clearly identifies the bottleneck of the system

with the domotics sensor network.

Table 3.1: *Processing delay of the SHG.*

hardware	fake interface	real ZigBee DSAN
SHG prototype	11.0 ms	177 ms
i3-based PC	34.8 ms	103 ms

In the second row of Table 3.1 we have reported the performance of the SHG software when run on a generic PC based on a Intel Core i3 processor running at 2.66 GHz with 4 GB of RAM. The purpose of these figures is to provide a comparison with a "high-end" hardware. The PC is somewhat slower in running the software, but is faster when the actual ZigBee network is attached. The "software" gap can be ascribed to the scarce optimization of both the code and the hardware, whereas the "DSAN" gap comes from the different connection with the ZigBee coordinator: serial (slower) on the SHG prototype, USB (faster) on the PC.

3.5.3 Effect of interference

To check the effect of the interference on the domotics system, we set up a sort of "use-case" scenario. We assumed that the Wi-Fi and Bluetooth networks are used to deliver different but massive data to the user(s). Specifically, a FTP or HTTP transfer is conveyed over the Wi-Fi connection, whereas an audio streaming is performed by means of the Bluetooth devices. The SHG thus acts as the source of the data, and its Wi-Fi and Bluetooth interfaces works mainly as transmitters. Such a scenario can be mapped, for example, to a file download form the Internet (the FTP transfer) and a user listening to a song retrieved from a local repository (the audio streaming). As for the ZigBee DSAN, this is used to send commands to the sensors spread across the house. We exploit the *request/response* paradigm implemented via the double SIP `MESSAGE` transactions (as illustrated in Section 2.4.5). Hence the SHG and the sensors alternate the roles of source and destination of the traffic.

Since the weak link in the chain is the ZigBee connection, our effort was mostly targeted at measuring the effect of the two "stronger" technologies, i.e. Wi-Fi and Bluetooth, on the performance of the ZigBee sub-system, and consequently on the capability of the user to control and have feedback from the ZigBee DSAN.

The test was organized in a similar manner as the performance experiment described in the previous Section. A SIP client sends a continuous flow of 100 requests to the SHG via

the local 100 Mbps Ethernet LAN. Each request is sent as soon as the previous response is received. No requests nor responses are lost in this segment of the system. The auxiliary SIP procedures (registration, publish, etc.) have been disabled, as they do not have any influence on the radio interference. The SHG then translates and casts the requests over the ZigBee network. Only one ZigBee sensor node is used for the test. This is sufficient for the purposes of the test, as the interference mostly occurs in the first wireless hop.

As for the wireless segment, we placed the Wi-Fi network on channels 1 or 11, and the ZigBee DSAN either on channel 25 or on channel 15. We did not test the system under overlapped ZigBee and Wi-Fi channels because we believe it is logical to assume that a deployed system will be smart enough to avoid such clearly troublesome kind of allocation. Also, we did not test the full range of possibile combinations, which is not the purpose of the present work – the reader interested in this kind of analysis can refer to [26].

We collected four performance parameters: the average command service time registered on the client (in short: service delay), the average and the peak command execution time on the ZigBee interface (in short: execution delay and peak delay), and the number of lost commands (i.e. either lost requests or lost responses; we made ourselves certain that the losses can only occur on the ZigBee network).

To monitor the activity on the 2.4 GHz spectrum, including possible external interference sources (e.g. other Wi-Fi networks), we used the AirView2-EXT ISM-band spectrum analyzer[4]. A screenshot of the power level in the test area has been taken before performing every experiment, to check whether strong external interferences are present, and thus avoiding biased results.

Table 3.2 reports the outcome of the tests. The first test, labelled "#0", is a preliminary test, used to benchmark the system when solely the ZigBee network is active (on channel 25). We can see that no commands are lost, and that the peak execution delay is just a few milliseconds greater than the average. This indicates that the behavior of the ZigBee network is quite stable. Also, the average service and execution delays differ only by 3 ms.

In the next test (#1) we activated both the Bluetooth and the Wi-Fi networks, with Wi-Fi placed on channel 1, i.e. the farthest possible from the ZigBee one. In this case, the interference is mostly due to Bluetooth, which covers the whole 2.4 GHz band. The performance drop is apparent, with an increase of 12% in the average times. The peak delay is the value that changed most, as it is now almost seven times the average execution delay. Thus,

[4] http://www.ubnt.com/airview

Table 3.2: *Interference performance of the SHG.*

Metric	test #0	test #1	test #2	test #3
service delay	179.4 ms	202.0 ms	835.9 ms	937.5 ms
execution delay	176.5 ms	197.2 ms	824.4 ms	913.6 ms
peak delay	181.4 ms	1268 ms	5538 ms	5291 ms
lost commands	0	0	0	0

the ZigBee network can still bring all commands to completion, but its response time has become quite unpredictable. In absolute terms, however, even the highest values (1.268 s) can be deemed acceptable.

In test #2 we moved the Wi-Fi emissions closer to the ZigBee ones, i.e. we put Wi-Fi on channel 11. In theory, there is still no overlapping between the ZigBee and Wi-Fi channels. But in fact the ZigBee segment is heavily penalized, as proved by the values in Table 3.2. The average delays reach almost 1 second, with the peak execution delay going beyond 5 seconds. For some applications these values might be critical, for the user annoying. Note, however, that no commands are lost.

The reason for these figures lies in the long timeouts and the numerous retries that are allowed at the ZigBee application and MAC layers. For example, the default application retry timeout is 1.5 seconds, and the allowed number of retries is 3, both at the MAC and at the application layer. Thus, the ZigBee network, which is highly hampered by Wi-Fi, can take advantege of several attempts to deliver each packet, and consequently the overall transmission times grows very large.

To have a confirmation that Wi-Fi interferes with ZigBee even in non-overlapping channels, we repeted the test by moving Wi-Fi to channel 1 and ZigBee to channel 15. The numbers of this test (#3), which are very similar, even worse than the previous ones, indeed corroborate this fact.

3.6 Conclusions

The performance of the SHG has been assessed in terms of served user requests per second, processing delay, average and peak service delay. The effect of having the three wireless interfaces on the same board that operate on the same frequency band has also been evaluated.

The results proved the SHG ability to support a considerable number of requests per second,

also from a different number of clients. Thus, the developed prototype can indeed be employed for large deployments, as it does have the ability to scale to any realistic requirement.

On the interference side, it emerged that ZigBee suffers the presence of both Bluetooth and Wi-Fi. Yet, while the former technology produces just a relatively small performance degradation, the presence of Wi-Fi is definitely more cumbersome, as the ability of the ZigBee network to accomplish its task in short times is heavily hampered. Though the weakness of ZigBee is well known, it is remarkable that this occurs even when Wi-Fi and ZigBee operate on channels that are nominally separated from each other. Our experiments showed a tremendous performance degradation when ZigBee and Wi-Fi are on adjacent channels. Nevertheless, by means of a proper configuration, we have also proved that it is possible to avoid command losses.

CHAPTER 4

IMPROVING PCA-BASED ANOMALY DETECTION BY USING MULTIPLE TIME-SCALES ANALYSIS AND K-L DIVERGENCE

The increasing number of network attacks causes growing problems for network operators and users. Thus, detecting anomalous traffic is of primary interest in IP networks management.

In this chapter we address the problem considering a method based on PCA (Principal Component Analysis) for detecting network anomalies. In more detail, this chapter presents a new technique that extends the state of the art in PCA-based anomaly detection. Indeed, by means of multi scale analysis and Kullback-Leibler divergence we are able to obtain great improvements with respect to the performance of the "classical" approach. Moreover we also introduce a method for identifying the flows responsible for an anomaly detected at the aggregated level.

The performance analysis, presented in this chapter, demonstrates the effectiveness of the proposed method.

4.1 Introduction

In the last few years the Internet has experienced an explosive growth. Along with the wide proliferation of new services, the quantity and impact of attacks have been continuously increasing. The number of computer systems and their vulnerabilities have been rising, while the level of sophistication and knowledge required to carry out an attack have been decreasing, as much technical attack know-how is readily available on Web sites all over the world.

As a consequence, many research groups have focused their attention on developing novel detection techniques, able to promptly reveal and identify network attacks. Among these, some of the most promising approaches are based on the use of the Principal Component Analysis (PCA).

PCA is a dimensionality-reduction technique that returns a compact representation of a multi-dimensional data set, by projecting the data onto a lower dimensional subspace. Indeed, it allows the reduction of the data set dimensionality (number of variables), while retaining most of the original variability in the data.

Another important element is that most of the works proposed in the literature, comprising most of the ones based on PCA, analyzes the single traffic flows, resulting to be unscalable and thus not applicable in modern backbone networks (that is the scenario considered in our work).

For such reasons, in this chapter, we have focused on the development of an anomaly based Network Intrusion Detection System (IDS), which applies PCA to traffic aggregates.

The starting point for our work is represented by the works by Lakhina et al. [34], [35], [36]. Indeed, we have taken the main idea of using PCA to decompose the traffic variations into their normal and anomalous components, thus revealing an anomaly if the anomalous components exceed an appropriate threshold.

Nevertheless, our approach introduces several novelties, allowing great improvements in the system performance. Specifically, the main contributions of the chapter are:

- analysis on four distinct levels of aggregation, namely ingress router, origin-destination flows, input link, and random aggregation, so as to detect anomalies that could be masked at some aggregation level.

- introduction of sketch [38] for identifying the anomalous flows inside the aggregates, once an anomaly has been detected. To be noted that previous works are only able to

detect the anomalous aggregate, without providing any information at the flow level.

- application of PCA at different time-scales. In this way the system is able to detect both sudden anomalies (e.g. bursty anomalies) and "slow" anomalies (e.g. increasing rate anomalies), which cannot be revealed at a single time-scale.

- use of the Kullback-Leibler divergence, together with the entropy, for detecting anomalous behavior, showing that our choice results in better performance and more "stability" for the system.

The remainder of this chapter, which extends the preliminary study described in [17], is organized as follows: Section 2 presents some relevant related works. Then Section 3 summarizes the theoretical background, briefly describing sketch and PCA, while Section 4 provides a detailed description of the implemented system. Then in Section 5 we analyze the experimental results, focusing on the improvements offered by our approach, and finally Section 6 concludes the chapter with some final remarks.

4.2 Background

PCA is the most commonly used technique to analyze high dimensional data structures.

Originally applied in the framework of image compression, in the last years it has been widely used in the domain of intrusion detection to solve the problem of high dimensionality of typical IDS data sets. Recent papers in networking literature have applied PCA to the problem of traffic anomaly detection with promising initial results.

In [39] the author first proposed an anomaly detection scheme based on the use of PCA. In this work, the method based on PCA is compared with a more "classical" approach based on clustering and Local Outlier Factor (LOF).

The theoretical bases for the application of PCA to the anomaly detection field are provided in [37], where the authors perform an analysis of several traffic measurements taken over two backbone networks (Abilene and Geant) demonstrating, by means of PCA, that these measurements have a *small intrinsic dimension*. This conclusion allows the authors to think that PCA could be suitable for anomaly detection, hence they represent the "theoretical" justification of all the works that propose a PCA-based anomaly detection algorithm, including our own.

Starting from this previous work, the same authors, in [34], introduce the subspace method that allows the separation of a high dimensional space occupied by a set of network traffic measurements into disjoint subspaces, respectively representative of the normal and anomalous components of the original data. The authors also define a method for revealing the anomalies and for pinpointing the traffic aggregates responsible for such anomalies.
In [35], the subspace method is then applied to three different metrics (packet count, byte count, and IP-flow count) for the detection of different kinds of anomalies.
A step forward in the method is given in [36], where the previous traffic volume metrics are substituted by the entropy of the empirical distribution of the traffic features, enabling the detection of a larger set of anomalies. These three works (namely, [34], [35], and [36]) represent the most advanced work based on the use of PCA for network anomaly detection. Starting from these papers, our work proposes several improvements to the method that allows to obtain significant improvements in the system performance.

A recent work by Ringberg et al. [43] applies the method described in [36], highlighting some intrinsic difficulties in tuning the method parameters and the consequent system instability. Nevertheless, our experimental results demonstrate that our method presents a good stability, demonstrating to be suitable for "real-world" application.

Other notable techniques that employ PCA methodology, not strictly related to the method presented in this chapter, include the works done by Wang et al. [52], Bouzida et al. [15] and Wang et al. [53].

Finally, a recent paper [20] has extended the method, by introducing a multi metric multi link analysis, which implies to analyze several traffic features taken on several links at the same time. Nonetheless this paper is not directly connected to our work, even tough, in principle, our contributions could be also applied to the method presented in the paper.

4.3 Theoretical Background

In this Section, we briefly describe the two key elements of the proposed approach (i.e., sketch and PCA), providing an intuitive reason for their application to network anomaly detection.

4.3.1 Sketch

Sketches are a family of data structures that use the same underlying hashing scheme for summarizing data. They differ in how they update hash buckets and use hashed data to derive

estimates. Among the different sketches, the one with the best time and space bounds is the so called count-min sketch [23], which is basically the one used in this work.

Specifically, the sketch data structure is a two-dimensional $d \times w$ array $T[l][j]$, where each row l ($l = 1,\ldots,d$) is associated with a given hash function h_l. In our work, we choose to use functions belonging to the 4-universal hash family[1] [51]. These functions give an output in the interval $(1,\ldots,w)$ and these outputs are associated to the columns of the array. As an example, the element $T[l][j]$ is associated to the output value j of the hash function l.

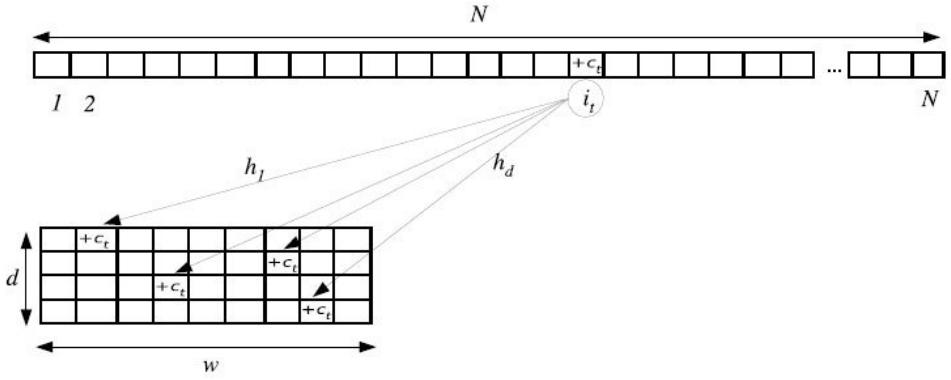

Figure 4.1: *Sketch: Update Function*

The input data are viewed as a stream that arrives sequentially, item by item. Each item consists of an *hash key*, $i_t \in (1,\ldots,N)$, and a *weight*, c_t. When new data arrive, the sketch is updated as follows:

$$T[l][h_l(i_t)] \leftarrow T[l][h_l(i_t)] + c_t \tag{4.1}$$

The update procedure is realized for all the different hash functions as shown in figure 4.1.

To be noted that, given the use of hash functions, it is possible to have some *collisions* in the sketch table. In this work we have taken advantage of this fact, indeed having collisions allows us to randomly aggregate the traffic flows, namely all the IP flows that collide in the

[1] A class of hash functions $H : (1,\ldots,N) \rightarrow (1,\ldots,w)$ is a *k-universal hash* if for any distinct $x_0,\ldots x_{k-1} \in (1,\ldots,N)$ and any possible $v_0,\ldots v_{k-1} \in (1,\ldots,w)$:

$$\Pr_{h\in H} = \Pr\{h(x_i) = v_i; \forall i \in (1,\ldots,k)\} = \frac{1}{w^k}$$

same bucket will be considered as an aggregate. To be noted that in this way each IP flow will be part of d distinct random aggregates, each of which will be analyzed to check if it presents any anomaly. This means that, in practice, any flow will be checked more than once, thus, it will be easier to detect an anomalous flow. Indeed, an anomalous flow could be masked in a given traffic aggregate, while being detectable in another one.

4.3.2 Principal Components Analysis

The Principal Components Analysis (PCA) is a linear transformation that maps a coordinate space onto a new coordinate system whose axes, called Principal Components (PCs), have the property to point in the direction of maximum variance of the residual data (i.e., the difference between the original data and the data mapped onto the previous PCs).

Specifically, the first PC captures the greatest degree of data variance in a single direction, the second one captures the greatest degree of variance of data in the remaining orthogonal directions, and so on.

Thus, the PCs are ordered by the amount of data variance they capture. Typically, the first PCs contribute most of the variance in the original data set, so that we can describe them with only these PCs, neglecting the others, with minimal loss of variance.

In mathematical terms, to calculate the PCs is equivalent to compute the eigenvectors. Thus, given the matrix of data $B = \{B_{i,j}\}$, with $1 < i < t$ and $1 < j < m$ (a data set of m samples captured in t time-bins), each PC, v_i, is the $i-th$ eigenvector computed from the spectral decomposition of $B^T B$, that is:

$$B^T B v_i = \lambda_i v_i \qquad i = 1, \ldots, m \tag{4.2}$$

where λ_i is the "ordered" eigenvalue corresponding to the eigenvector v_i.

In practice, the first PC, v_1, is computed as follows:

$$v_1 = \underset{\|v\|=1}{\operatorname{argmax}} \|Bv\| \tag{4.3}$$

Proceeding recursively, once the first $k-1$ PCs have been determined, the $k-th$ PC can be evaluated as follows:

$$v_k = \underset{\|v\|=1}{\operatorname{argmax}} \left\| (B - \sum_{i=1}^{k-1} B v_i v_i^T) v \right\| \tag{4.4}$$

where $||\cdot||$ denotes the L^2 norm.

Once the PCs have been computed, given a set of data and its associated coordinate space, we can perform a data transformation by projecting them onto the new axes.

4.4 System architecture

In this Section, we present the architecture of the system we have implemented to detect anomalies in the network traffic.

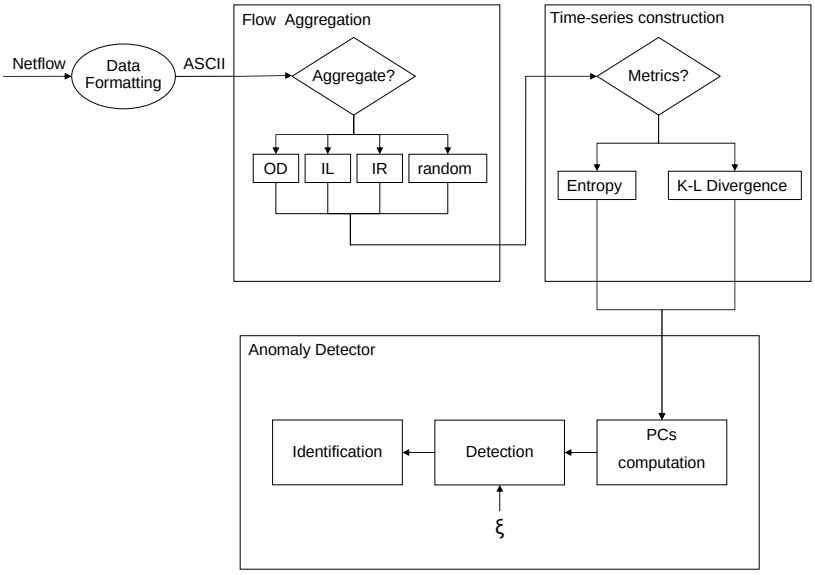

Figure 4.2: *System Architecture*

4.4.1 System Input

First of all the input data are processed by a module called, in Figure 4.2, Data Formatting. Indeed, this module is responsible for reading the Netflow [21] traces and transforming them into ASCII data files, by means of the Flow-Tools [1].

In more detail, in our implementation we have in input Netflow data, measuring the traffic gone through a given router over given time-bin (in the following we have T distinct time-bins).

To be noted that our system works on three distinct time-scales, namely it uses time-bins of 5, 10, and 15 minutes.

Moreover, to facilitate the detection of correlation and periodic trends in the data, we have studied different levels of aggregation. Specifically, the block called Flow Aggregation realizes four different levels of aggregation:

- Ingress Router (IR)
- Origin Destination (OD) flows
- Input Link (IL)
- Random Aggregation

Using the IR aggregation, data are organized according to which router they entered the network. The OD flow aggregates traffic based on the ingress and egress routers used by an IP flow. Instead, the IL aggregates IP flows with the same ingress router and input interface. Finally, the random aggregation is performed by means of the sketch (as described in the previous Section).

The output of this block is given by $4T$ distinct files, each file corresponding to a specific time-bin and a specific aggregate.

Note that the modularity of the system allows great flexibility. Indeed, instead of considering the number of bytes sent by a given IP, the system administrator can easily choose of using another traffic descriptor that better allows her to detect the different attacks.

4.4.2 Time Series Construction

After the data have been correctly formatted, they are passed as input to the third module, responsible for the construction of the time series.

In this work, we have taken into consideration the quantity b_{ij}^t, that is the number of bytes sent by the j^{th} IP source address of the i^{th} aggregate in the time-bin t. The feature distribution has been estimated with the empirical histogram. Thus, in each time-bin for each aggregate we have evaluated the histogram as follows:

$$B_{i,\cdot}^t = \{\beta_{i,j}^t\}_{j=1}^M \tag{4.5}$$

where

$$\beta_{i,j}^{t} = \frac{b_{i,j}^{t}}{\sum_{j=1}^{M} b_{i,j}^{t}} \tag{4.6}$$

Unfortunately, the histogram is an high-dimensional object quite difficult to handle with low computational resources. For this reason we have tried to concentrate the information taken by the histogram in a single value, able to hold the most of the useful information, that, in our case, is the trend of the distribution. Previous works [34] [35] [36] have emphasized the possibility to extract very useful information from the *entropy* of the histogram, which provides a computationally efficient metric for estimating the degree of dispersion or concentration of a distribution.

Given the empirical histogram, X^t, we can evaluate the entropy value as follows:

$$y_{it} = -\sum_{i=1}^{M} \beta_{i,j}^{t} \log_2 \beta_{i,j}^{t} \tag{4.7}$$

Nevertheless, the entropy is only able to capture the information related to a single time-bin, while from our point of view it would be much more important to capture the the difference between packet feature distributions of two adjacent time-bins.

For this reason, in this work we have also used another metric that is the Kullback-Leibler (K-L) divergence.

Thus, given two histograms X^t (captured in time-bin t) and X^{t-1} (captured in time-bin $t-1$), the K-L divergence is defined as follows:

$$D_{t,i} = \sum_{j=1}^{M} \beta_{i,j}^{t} \log \frac{\beta_{i,j}^{t}}{\beta_{i,j}^{t-1}} \tag{4.8}$$

Despite of the method used, this module will output a matrix for each type of aggregation in which, for all the aggregates, the values of the metric (entropy or K-L divergence) evaluated in each time-bin are reported.

This matrix has the following structure:

$$Y = \begin{pmatrix} y_{11} & \cdots & y_{1N} \\ \vdots & & \vdots \\ y_{T1} & \cdots & y_{TN} \end{pmatrix} \tag{4.9}$$

where N is the number of aggregates and T the number of time-bins.

4.4.3 PCs computation

After the time-series have been constructed, they are passed to the module that applies PCA. The computation of the PCs can be performed using the Equation (4.3) and (4.4). As described before, typically there is a set of PCs (called *dominant* PCs) that captures most of the variance in the original data set. The idea is to select the dominant PCs and describe the normal behavior only using these ones. It is worth highlighting that the number of dominant PCs is a very important parameter, and needs to be properly tuned when using PCA as a traffic anomaly detector.

In our approach the set of dominant PCs is selected by means of the scree-plot method. As a result, we separate the PCs into two sets, dominant and negligible PCs, that will be then used to distinguish between normal and anomalous variations in traffic.

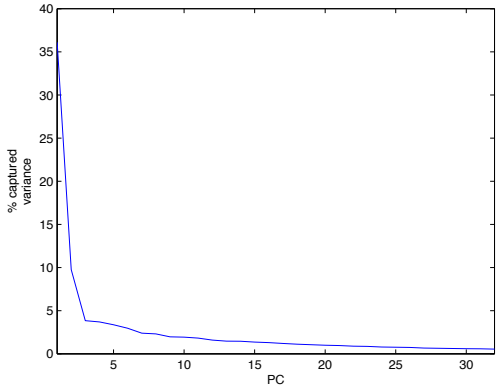

Figure 4.3: *Scree-plot*

In more detail, a scree-plot is a plot of the percentage of variance captured by a given PC. Figure 4.3 reports a scree-plot of a particular data set used in our study. Visually, from the graph we can observe that the first $r = 5$ PCs are able to correctly capture the majority of the variance. These PCs are the dominant PCs:

$$P = (v_1, \ldots, v_r) \tag{4.10}$$

The method is based on the assumption that these PCs are sufficient to describe the normal behavior of traffic.

4.4.4 Detection Phase

The detection phase is performed by separating the high-dimensional space of traffic measurements into two subspaces, which capture normal and anomalous variations, respectively.

Specifically, once the matrix P has been constructed, we can partition the space into a normal subspace ($\hat{S}$), spanned by the dominant PCs, and an anomalous subspace ($\tilde{S}$), spanned by the remaining PCs. The normal and anomalous components of data can be obtained by projecting the aggregate traffic onto these two subspaces. Thus, the original data, in the time-bin t, Y_t are decomposed into two parts as follows:

$$Y_t = \hat{Y}_t + \tilde{Y}_t \tag{4.11}$$

where $\hat{Y}_t$ and $\tilde{Y}_t$ are the projection onto $\hat{S}$ and $\tilde{S}$, respectively, and can be evaluate as follows:

$$\hat{Y}_t = PP^T Y_t \tag{4.12}$$

$$\tilde{Y}_t = (I - PP^T)Y_t \tag{4.13}$$

To be noted that, when anomalous traffic crosses the network, a large change in the anomalous component ($\tilde{Y}_t$) occurs. Thus, an efficient method to detect traffic anomalies is to compare $\|\tilde{Y}_t\|^2$ with a given threshold ξ.

Particularly, if $\|\tilde{Y}_t\|^2$ exceeds ξ, the traffic is considered anomalous, and we mark the time-bin t as an anomalous time-bin (Figure 4.4).

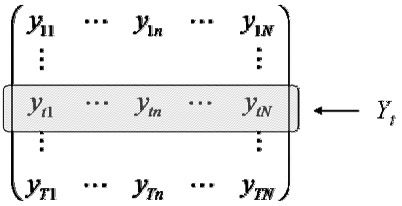

Figure 4.4: *Data matrix*

If we use a random aggregation the detection phase can be improved performing a further step.

Indeed, in this case, since we have used d different hash functions h_j we have d data matrices, Y^j (one for each function). So, the previously described analysis (performed for each Y^j) returns d different responses. Thus, a voting analysis is performed. We evaluate the number of produced alarms and we decide if the time-bin is anomalous or not according to the following rule:

$$\begin{cases} \text{anomalous} & if \quad \text{number of alarms} > \dfrac{d}{2} - 1 \\ \text{normal} & \text{otherwise} \end{cases}$$

4.4.5 Identification Phase

If an anomaly has been detected, the system performs a new phase called anomaly identification.

To be noted that PCA works on a single time-series, so in the detection phase we are able to identify the time-bin during which traffic is anomalous. But, at this point we do not know the specific network event that has caused the detection. In fact, it is worth noticing that an anomalous time-bin may contain multiple anomalous events, and that a single anomalous event can span over multiple time-bins.

In this phase we want to identify the specific flow responsible for the revealed anomaly. In more detail, at first, we search the specific traffic aggregation in which the anomaly has occurred. Since an anomaly is detected when $\|\tilde{Y}\|^2$ exceeds the threshold ξ, the identification method consists of searching the particular aggregate that, if removed, from the aforementioned statistics, would bring it under threshold.

In this way we identify an element y_{tn} of the vector Y_t that corresponds to a set A of candidate IP flows responsible for the detected anomaly. To be noted that if the traffic aggregation is performed by means of IR, OD flow, and IL we are not able to determine the specific flow responsible for the revealed anomaly. On the contrary, when we use random traffic aggregation, we have introduced a method that allows us to correctly detect the specific flows involved in the anomalous traffic.

For this analysis we can use the information stored inside the $\{Y^j\}_{j=1}^d$ data matrices that contain d analysis of the same traffic data using different hash functions. At this point, for each of these matrices we can detect the anomalous time-bin and the anomalous aggregate, so we can identify an element $\{y_{tn}^j\}_{j=1}^d$ of each matrix. Each of these elements correspond to

an aggregate (A_j) of candidate anomalous IP flows. Given that, the responsible IP addresses can be found by simply evaluating the intersection of all these aggregates $I = \bigcap_{j=1}^{d} A_j$.

4.5 Experimental results

As it is known a serious issue in testing the IDSs is represented by the lack of complete data sets provided with a ground truth. Indeed, the only one (DARPA IDEVAL) dates back to 1999 and it is not representative of real traffic. Thus, it is a common choice to use a real data set and to synthetically add some anomalies.

Following the approach presented in [36], the proposed system has been tested using a publicly available data set, composed of traffic traces collected in the Abilene/Internet2 Network [5], a hybrid optical and packet network used by the US research and education community.

The used traces consist of the traffic related to nine distinct routers, collected in one week, and are organized into 2016 files, each one containing data about five minutes of traffic (netflow data). To be noted that the last 11 bits of the IP addresses are anonymized for privacy reasons; nevertheless we have more than 220000 distinct IP addresses.

Since the data provided by the Internet2 project do not have a ground truth file, we are not capable of saying *a priori* if any anomaly is present. Because of this reason we have performed a manual verification of the data (according to the method presented in [36]), analyzing the traces for which our system reveals the biggest anomalies. Moreover we have synthetically added some anomalies in the data, so as to be able to correctly interpret the offered results (more details in Appendix A).

As already stated in the previous section, we have decided to give in input to the system the number of bytes sent by a given IP address. This choice is supported by the obtained experimental results. Nevertheless, it is possible to feed the system with another metric, just simply modifying the first block of the system, if the "new" metric can result more suitable for detecting some attacks.

Before detailing the performance achieved by the system in terms of number of detected anomalies and detection rate, let us analyze its sensitivity to two key parameters, i.e., the dimension of the normal subspace (number of dominant PCs, r) and the sketch size w.

It is important to say that the presented performance has been obtained varying the value of the threshold ξ in a range chosen on the basis of the values of $||\tilde{Y}||^2$.

As previously said, for the selection of an appropriate number of PCs we have used the

scree-plot method. In Figure 4.3 we report the scree-plot related to random aggregation for a sketch size w of 64 and bin-size of 5 minutes. From the plot we can easily notice that most of the data energy is captured by the first five PCs and that after the eighth PC, the contribution of the remaining PCs is less than 4%. Thus we have decided to perform our analysis with a number of PCs $r \in [2,7]$.

Figure 4.5 shows the detection rate (computed over the synthetically added anomalies plus the ones revealed by the manual inspection of the traces) when varying the number of PCs (note that, given the nature of the data set, we cannot plot an "exact" ROC curve).

It is worth noticing that a very low value of r takes to correctly detect a good number of anomalies, also raising a big number of false alarms, due to the fact that also the "normal" components are considered in the anomalous subspace. Vice-versa, considering a high value of r takes to a bad detection rate, behavior due to the fact that considering a high number of PCs implies to insert in the normal subspace some anomalous components. Given this, it is evident that r is an important parameter and it has be chosen so as to obtain a good trade-off between detection rate and false alarm rate.

Concerning the sketch size, it is important to highlight that in our implementation we have used $d = 8$ distinct hash functions, which give output in the interval $[0, w-1]$; so the resulting sketch tables will be $\in \mathbb{N}_{8\times w}$, where w is a parameter to set. As far as the hash functions are concerned, we have used 4-universal hashes, obtained as:

$$h(x) = \sum_{i=0}^{3} a_i \cdot x^i \ mod \ p \ mod \ w$$

where the coefficients a_i are randomly chosen in the set $[0, p-1]$ and p is an arbitrary prime number (we have considered the Mersenne numbers).

The choice of w is very important, since this parameter determines the number and the composition of the aggregates, significantly influencing the detection rate. For this reason, we have studied the detection rate achieved by the system when varying the sketch size. Figure 4.6 shows the results of such analysis, when r has been fixed equal to 5. From an analysis of the achieved results we have concluded that the best performance is achieved when $w = 64$. Indeed, even though the graph shows better performance for $w = 32$, in that case there are too few traffic aggregates taking to a huge number of false alarms.

Before detailing the achieved performance, it is also worth highlighting that the proposed method introduces some computational overhead with respect to the state of the art method described in [34][35][36] (in the following referred to as "classical" PCA anomaly detection

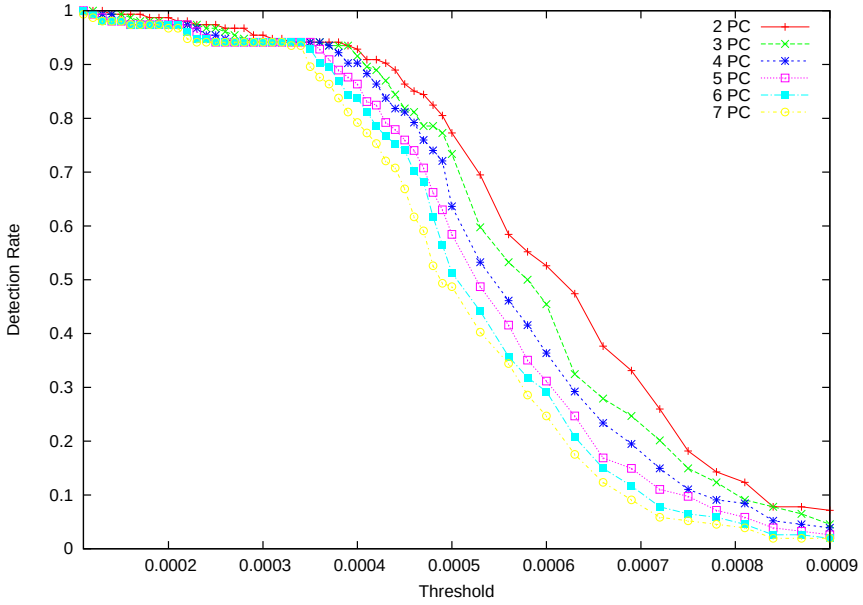

Figure 4.5: *Detection Rate vs. Number of PCs*

method and used as performance benchmark). Nonetheless, the system input is represented by Netflow traces that are produced by the router every five minutes. This basically means that the system has five minutes to elaborate the data, before new data are produced. In such a scenario, the overhead introduced by our work results to be negligible.

In the following subsections we show the performance achieved by our system, in terms of detection rate (computed over the synthetically added anomalies plus the ones revealed by the manual inspection of the traces) and total number of detected anomalies. It is important to highlight that the system stops analyzing a given time-bin, once an anomaly has been detected, this implies that the number of detected anomalies is equal to the number of anomalous time-bins. The presence of multiple anomalies in a time-bin will eventually be detected during the identification phase.

Such results discussion is divided into three distinct parts:

- performance achieved with the different types of traffic aggregations

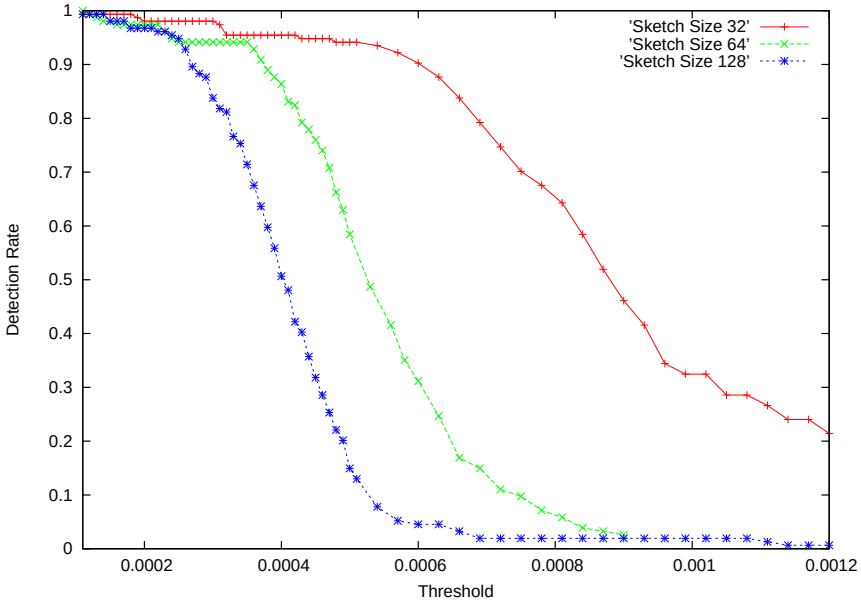

Figure 4.6: *Detection Rate vs. Sketch Size*

- performance achieved by adding multi time-scale analysis to the "classical" PCA anomaly detection method
- performance achieved by adding K-L divergence to the "classical" PCA anomaly detection method

4.5.1 Traffic aggregations

The aim of this first performance evaluation is to establish if considering different traffic aggregation criteria can somehow modify the detection rate achieved by the system, and in particular to evaluate the effectiveness of the random aggregation provided by the use of sketch. The graphs presented in this section have been obtained by considering time-bin of 5 minutes, sketch size $w = 64$, and the use of entropy as detection method.

Figures 4.7 and 4.8 respectively show the detection rate and the total number of detected anomalies when the OD aggregation is used. The different curves have been obtained varying the number of PCs. As expected the best detection rate is obtained when considering a very low number of PCs, but this implies to have a high number of false positives. By manual inspecting the data set, it appears that the best trade-off is achieved when using 5 PCs.

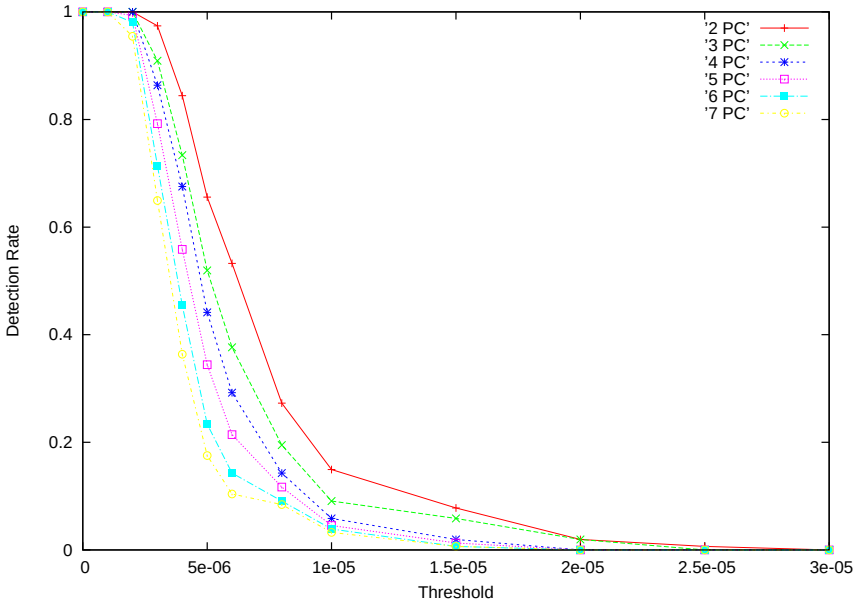

Figure 4.7: *Detection Rate for OD traffic aggregation*

From the graphs we can see that the performance do not strongly depend on the number of PCs and that the detection rate and the total number of detected anomalies decrease following the same trend, when varying the threshold.

Regarding the IR aggregation the achieved performance is shown in Figures 4.9 and 4.10. Differently from the previous graphs we can notice that, in this case, the performance strongly depend on the number of PCs, and that we have a strong worsening when we consider more than 3 PCs.

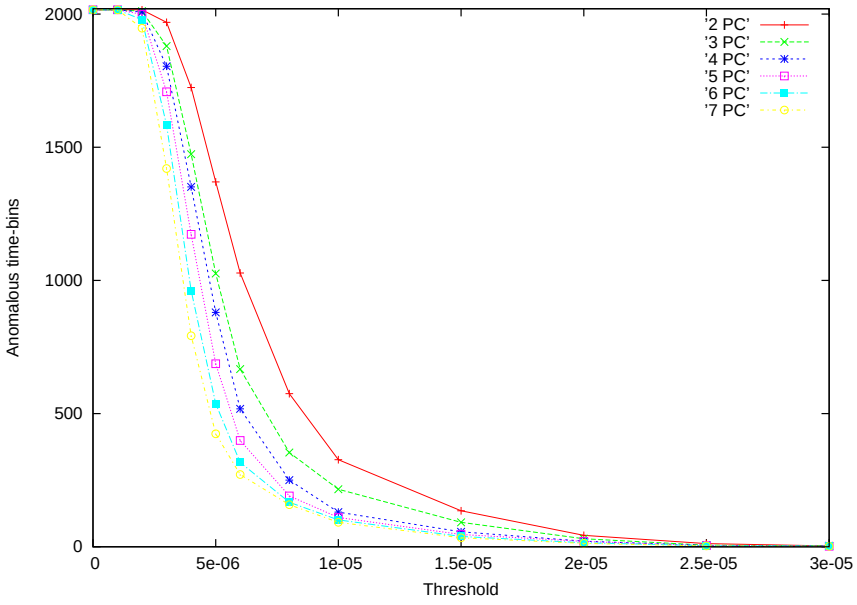

Figure 4.8: *Detected Anomalies for OD traffic aggregation*

It is also important to notice that, when using IR aggregation, the total number of detected anomalous events is much lower than in the previous case. This last consideration can take us to conclude that the IR aggregation behaves better than the OD one, but the strong dependence of this aggregation on the number of PCs makes the method hard to be tuned.

From Figures 4.11 and 4.12 we can see that the IL aggregation presents almost the same behavior of the OD aggregation. Indeed the performance do not strongly depend on the number of PCs and present the same trend when varying the threshold.

To conclude this analysis, Figures 4.13 and 4.14 present the performance achieved when using random aggregation.

As we can see this method allows us to obtain better performance in terms of detection rate: indeed the detection rate is stable over the 95%, for several values of the threshold, while the total number of detected anomalies decreases faster.

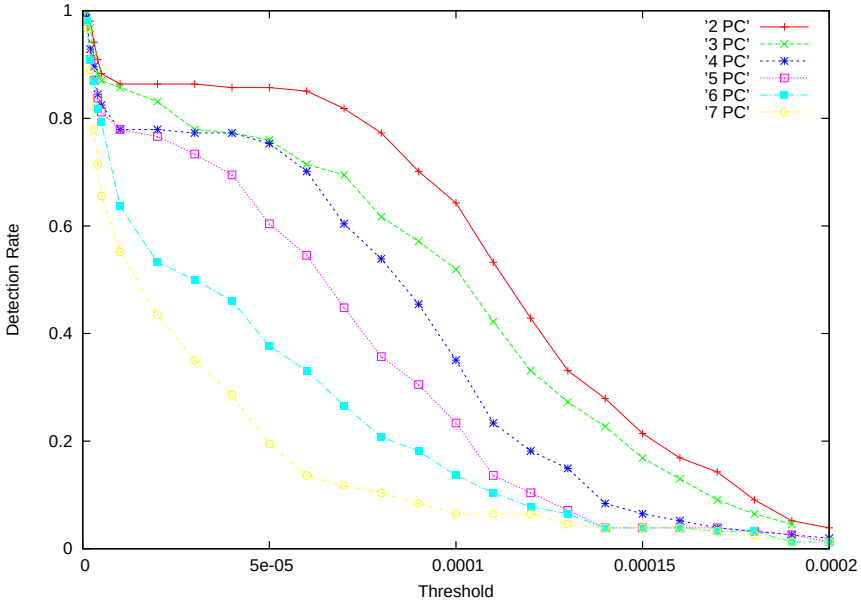

Figure 4.9: *Detection Rate for IR traffic aggregation*

To better compare the presented results, in Table 4.1 we show the detection rate achieved by using the different aggregations, once fixed the total number of detected anomalies to 910. It is easy to conclude that the best performance is given by the IR and random aggregations.

	OD	IR	IL	Random
Detection Rate	51%	90%	43%	90%

Table 4.1: *Detection Rate - Anomalous Time-Bins = 910*

Moreover, if we consider that in a given time-bin there can be more than an anomaly, it is clear that OD, IR, and IL aggregation do not allow us to distinguish between them, since they just reveal an anomalous time-bin, while using the sketch we can also distinguish between

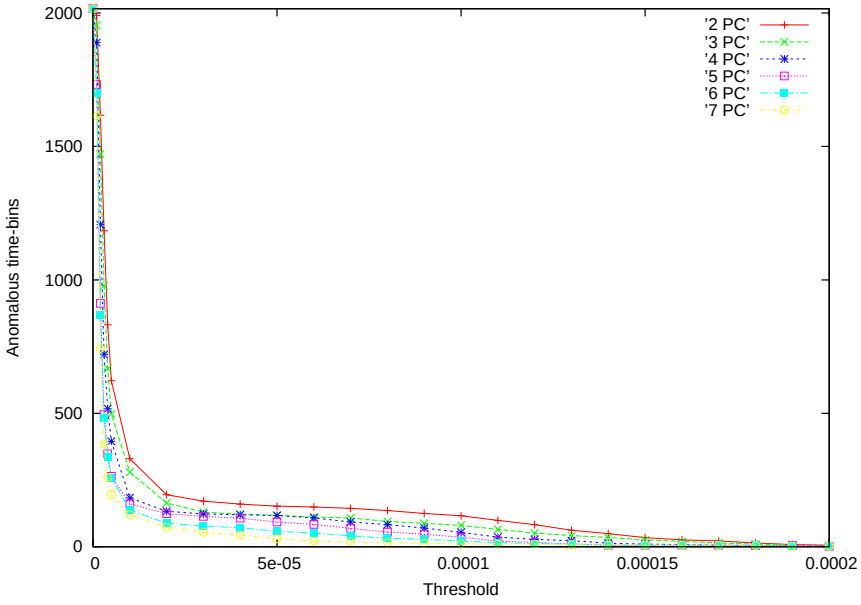

Figure 4.10: *Detected Anomalies for IR traffic aggregation*

contemporary anomalies, since each flow is part of several aggregates.

It is also important to highlight that we can take profit from this fact, so as to be able to identify the flows that contribute to the anomaly in a given time-bin, as previously described. Figure 4.15 shows the percentage of anomalous flows correctly identified. The performance depends on the threshold value. In more detail, it is worth noticing that in correspondence of too low values the identification is not able to correctly work, since it is not possible to find any flow that, if removed, would bring $||\tilde{y}||^2$ under the threshold.

To sum up we can conclude that the best performance is obtained with the use of the random aggregation, which is also the only one that allows the identification of the anomalous flows.

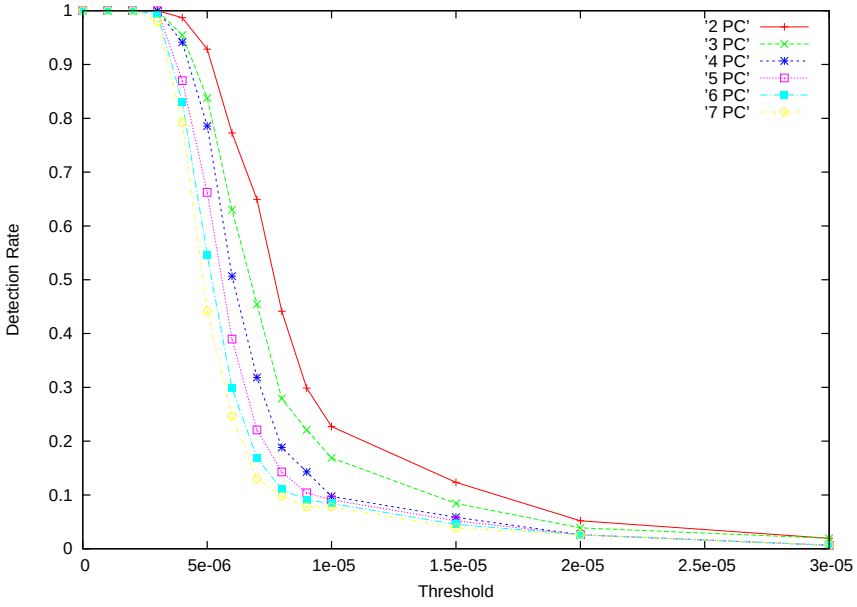

Figure 4.11: *Detection Rate for IL traffic aggregation*

4.5.2 Multi time-scale analysis

The second session of experimental tests has been carried out to evaluate the effectiveness of using multiple time-scales, namely, in our implementation, we have taken into account time-bins of 5, 10, and 15 minutes.

The graphs of this section have been realized taking into account a sketch size w equal to 64 and the use of entropy. Regarding the traffic aggregation, given the results previously discussed, the presented plots are related to the use of sketch.

Figures 4.16 and 4.17 respectively show the detection rate and the total number of detected anomalies when considering time-bins of 10 minutes. If we compare these results with those obtained with time-bins of 5 minutes (Figures 4.13 and 4.14) we can notice that the behavior is quite different.

It is important to highlight that the system analyzes each time-bin independently of the

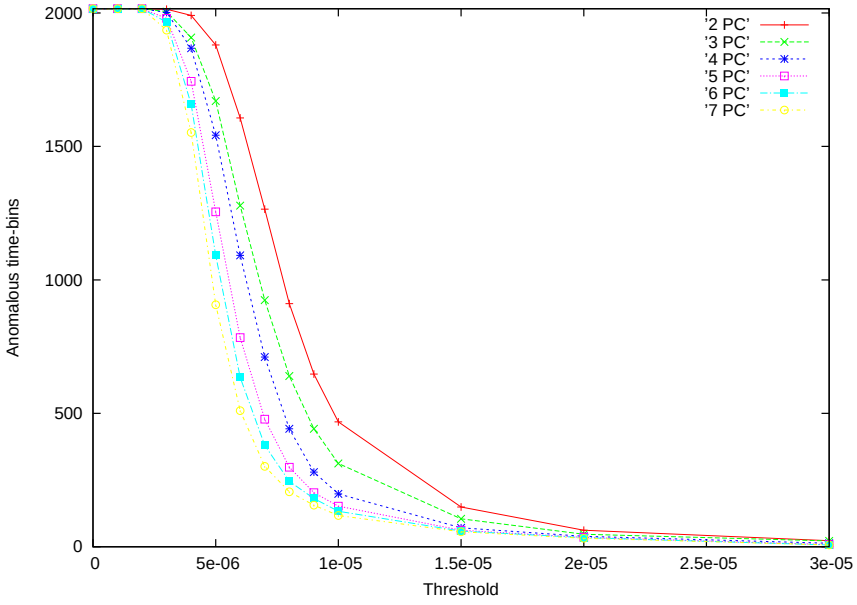

Figure 4.12: *Detected Anomalies for IL traffic aggregation*

others (i.e., in a stateless way). This choice, due to scalability reasons, implies that a single anomaly spanning over multiple time-bins, will be counted more than once. Specifically, an anomaly that lasts two adjacent time-bins of 5 minutes will be counted as two distinct anomalies when using time-bins of 5 minutes, while it will be counted only once, when using time-bins of 10 minutes. Given that, it is not possible to directly compare the presented graphs, without taking into account the tables inserted at the end of the section.

The same consideration holds when considering time-bins of 15 minutes (see Figures 4.18 and 4.19).

To be able to perform a more significant comparison between the results obtained with the use of multiple time-scales and to realize if, in fact, it takes to some improvements we have evaluated some more parameters.

Table 4.2 presents the increase in the detection rate offered by the use of the two time scales

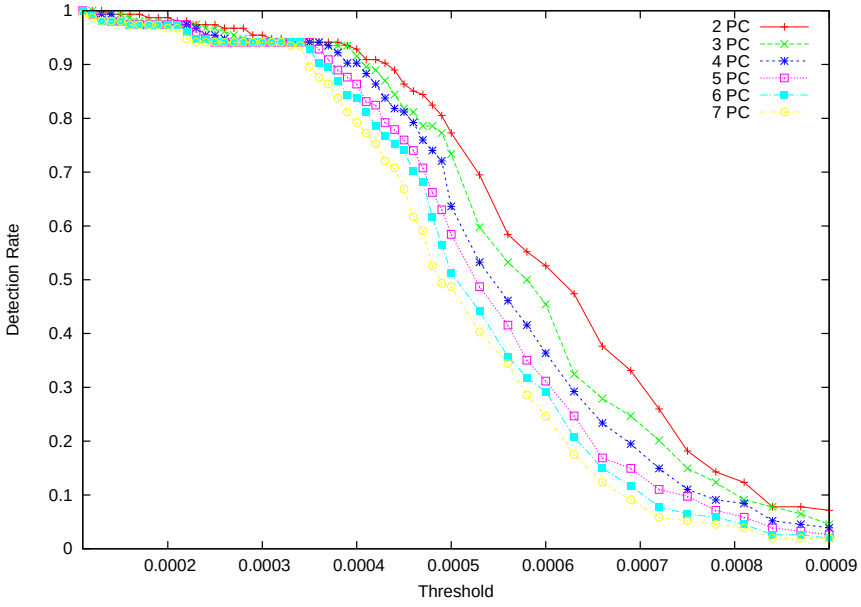

Figure 4.13: *Detection Rate for Random Aggregation*

(5 and 10 minutes). In more detail for a given number of PCs, we have fixed the value of the detection rate and we have evaluated how many anomalies are revealed at both the time-scales and how many only by one of the two.

It emerges that, as an example, when using 3 PCs, fixing the detection rate at 73%, 68% of the detected anomalies are commonly detected at both the time-scales, while the other 5% detected with time-bins of 5 minutes are not detected with time-bins of 10 minutes and vice-versa. As a result the use of 2 distinct time-scales (with 3 PCs) takes in this case to improve the performance from a detection rate of 73% to 78%.

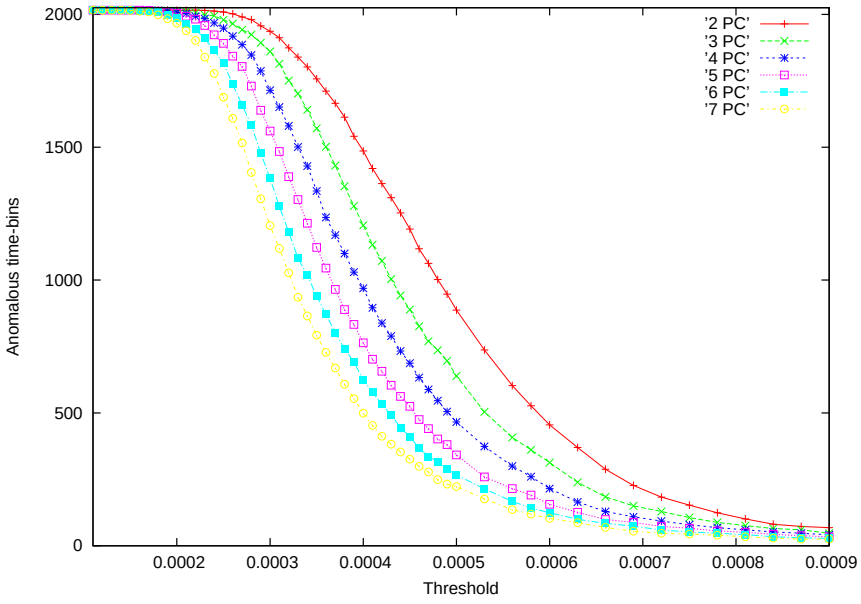

Figure 4.14: *Detected Anomalies for Random Aggregation*

PC Number	Detection Rate	Additive Detections
3	80%	1%
4	80%	2%
5	80%	2%
3	73%	5%
4	72%	3%
5	72%	4%

Table 4.2: *Multi Time-Scale Additive Detections - time-bins of 5 and 10 minutes*

Analogous considerations can be done observing the results presented in tables 4.3 and 4.4, where we present the increase in the detection rate respectively offered by the use of the 5

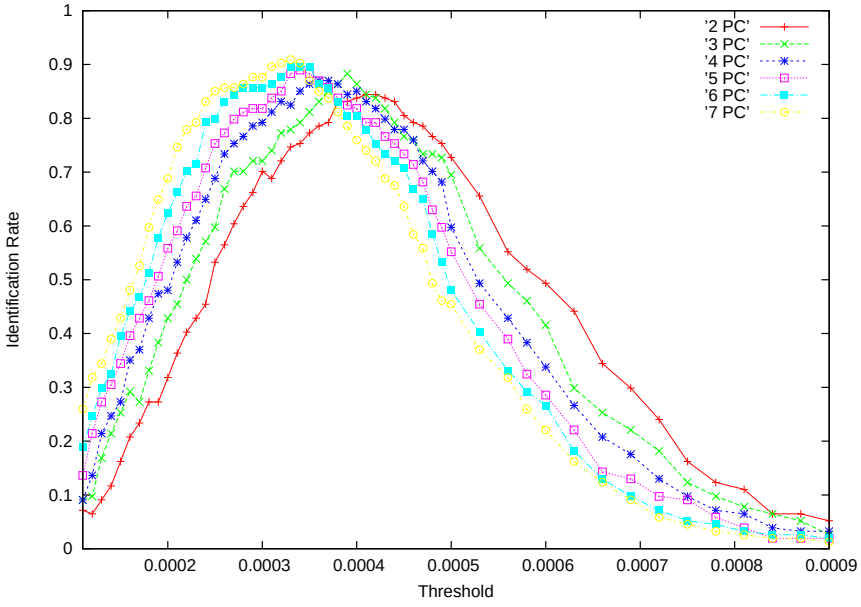

Figure 4.15: *Flows Identification Rate - Random Aggreagation*

and 15 minutes time-bins and 10 and 15 minutes time-bins.

As a general consideration we can easily conclude that the use of multiple time-scales significantly improves the performance.

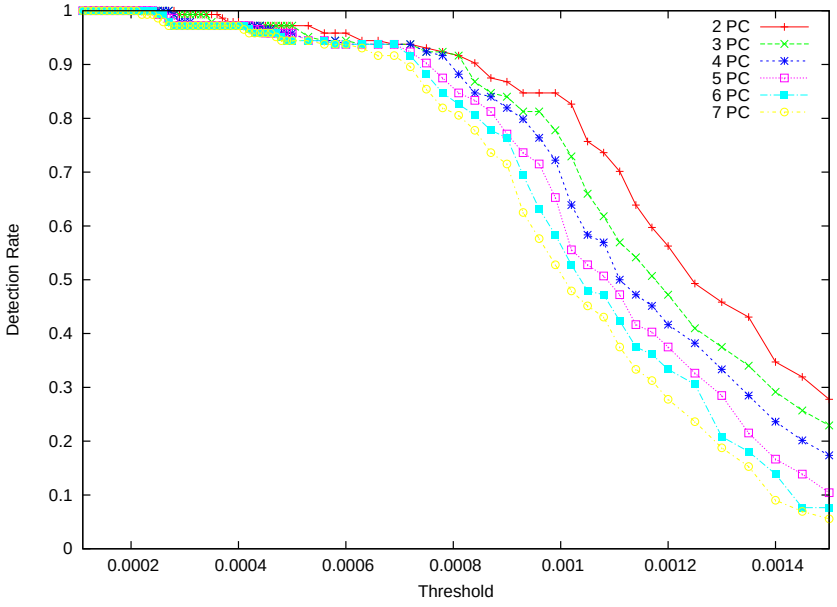

Figure 4.16: *Detection Rate for Random Aggregation, binsize 10 min*

PC Number	Detection Rate	Additive Detections
3	80%	3%
4	79%	4%
5	80%	4%
3	73%	5%
4	70%	5%
5	71%	6%

Table 4.3: *Multi Time-Scale Additive Detections - time-bins of 5 and 15 minutes*

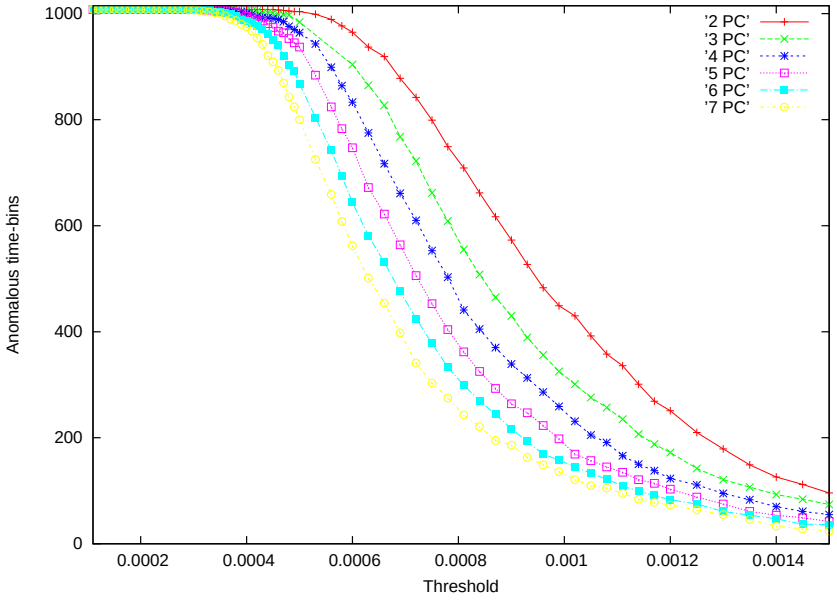

Figure 4.17: *Detected Anomalies for Random Aggregation, binsize 10 min*

PC Number	Detection Rate	Additive Detections
3	79%	2%
4	80%	3%
5	81%	1%
3	73%	5%
4	70%	3%
5	71%	3%

Table 4.4: *Multi Time-Scale Additive Detections - time-bins of 10 and 15 minutes*

The effects related to the use of multiple time-scale are also highlighted by Figures 4.20, 4.21, and 4.22. From these graphs we can easily notice like different dimensions of the time-

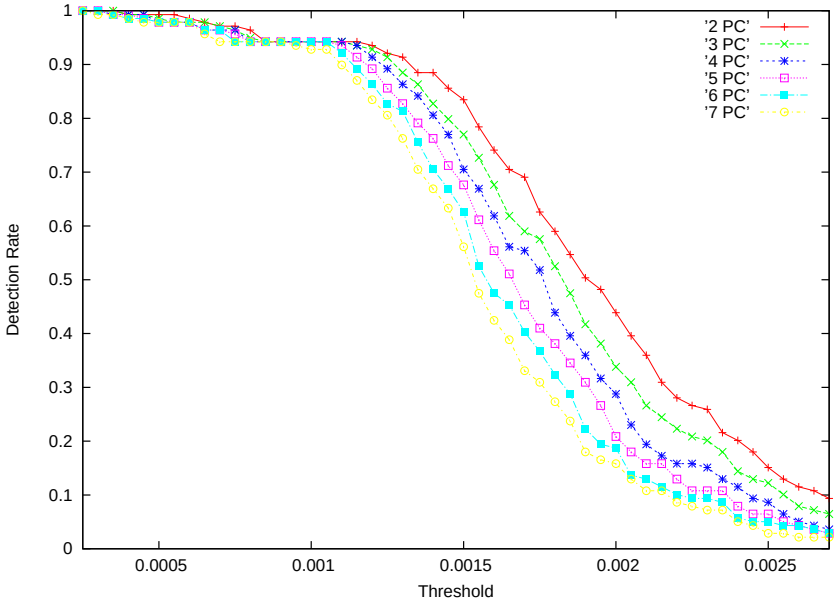

Figure 4.18: *Detection Rate for Random Aggregation, binsize 15 min*

bin take to obtain very different results when identifying long as well as short anomalies.

4.5.3 K-L divergence

The last session of experiments has been conducted to evaluate the effectiveness of using K-L divergence instead of the entropy.

Figures 4.23 and 4.24 respectively show the detection rate and the number of detected anomalies achieved when using K-L divergence. By comparing these graphs with those related to the use of entropy (Figures 4.13 and 4.14), it is easy to conclude that the performance is quite different.

Nevertheless, as in the case of the multi time-scales, it is not easy to directly compare these results. For this reason in Table 4.5 we present an analysis similar to the one presented in Tables 2-4.

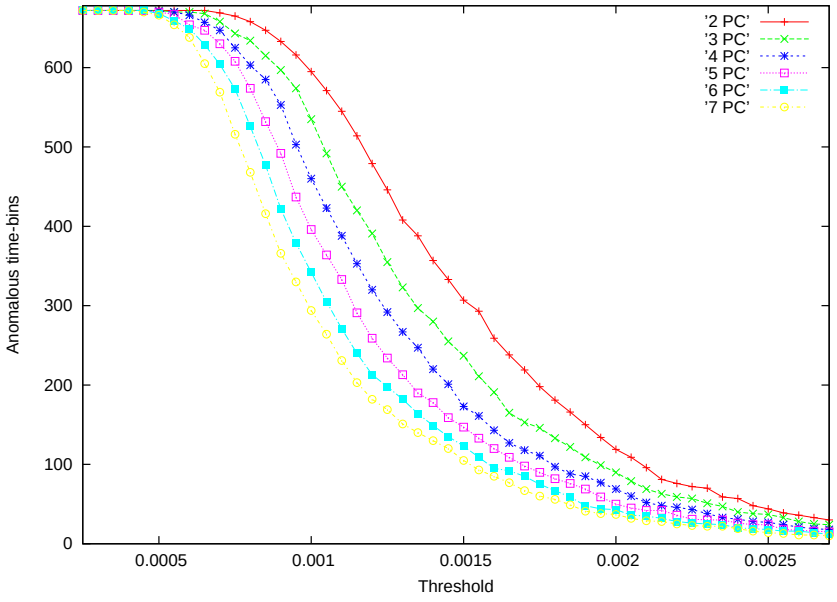

Figure 4.19: *Detected Anomalies for Random Aggregation, binsize 15 min*

In this case we can easily notice that the combined use of both entropy and K-L divergence takes to great improvements in the performance. As an example, if we consider to use 5 PCs with a detection rate of 70%, we can see that the 55% of the anomalies are detected by both the systems, while the 15% detected by using the entropy is different from the 15% detected by using K-L divergence, thus in this case the detection rate improves from 70% to 85%.

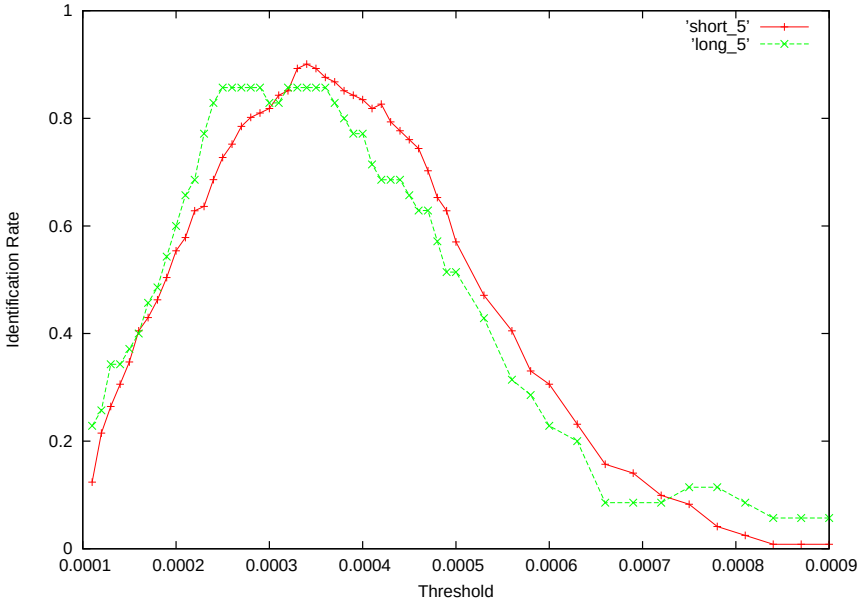

Figure 4.20: *Identification Rate - time-bins 5 minutes*

PC Number	Detection Rate	Additive Detections
3	82%	10%
4	81%	9%
5	81%	10%
3	73%	15%
4	72%	16%
5	70%	15%

Table 4.5: *K-L and entropy Additive Detections*

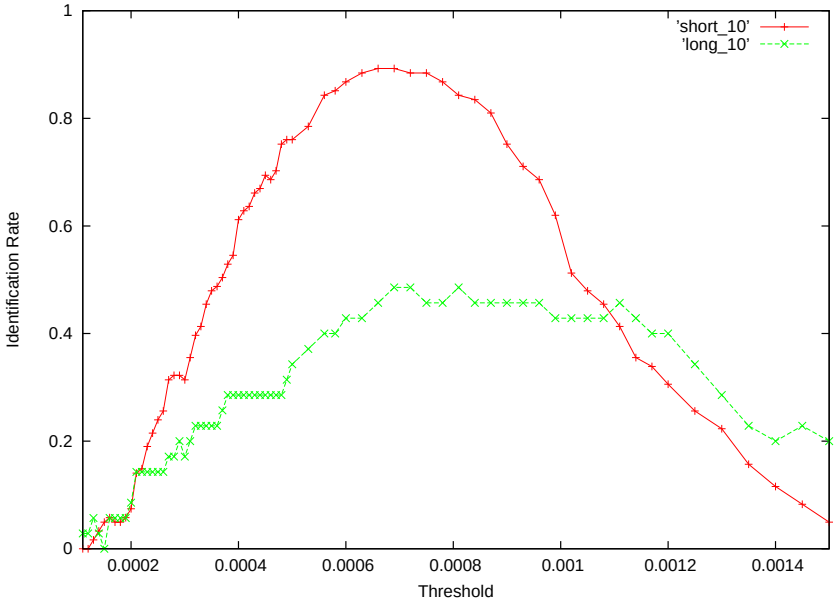

Figure 4.21: *Identification Rate - time-bins 10 minutes*

4.6 Conclusions

In this chapter we have presented a novel anomaly detection method, based on the use of PCA. With respect to the "classical" approaches, our system achieves several improvements due to the use of K-L divergence and multi time-scale analysis. Moreover our system also goes beyond the state of the art, by introducing a method for identifying the traffic flows responsible for an anomaly detected at the aggregate level.

To assess the validity of the proposed solution, we have tested the system over traffic collected in the Internet2/Abilene network. The performance analysis has highlighted that, for a proper choice of the tuning parameter, the implemented system obtains very good results, detecting all the synthetic anomalies.

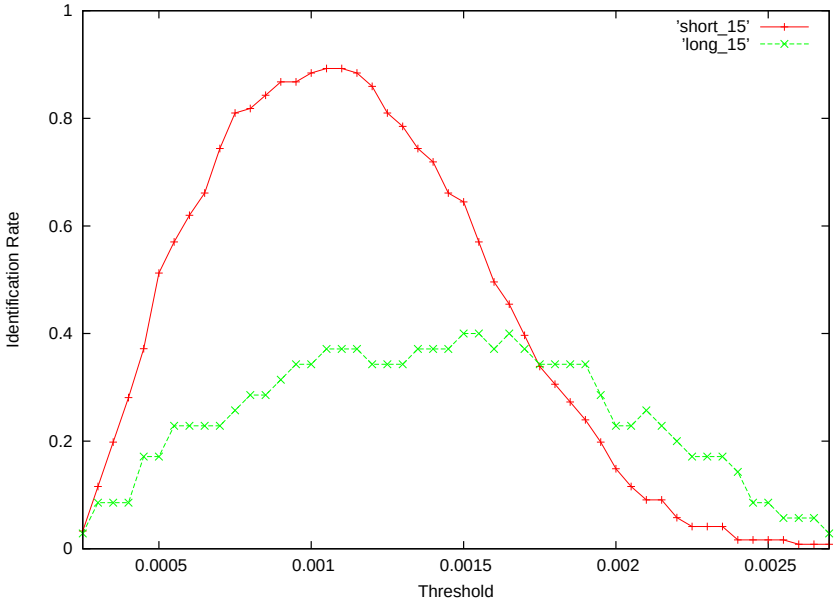

Figure 4.22: *Identification Rate - time-bins 15 minutes*

Appendix A

The dataset has been realized by adding some synthetic anomalies to the Abilene/Internet2 Network traffic traces. Specifically, we have used the traffic traces collected in one week and associated to the nine routers of the backbone networks.

Given such traces we have added anomalies that can be associated to DoS and DDoS attacks, either spanning a single or multiple time bins. To introduce anomalies that are plausible in the dataset, we have at first performed a manual verification of the data (according to the method presented in [36]), analyzing the traces for which our system reveals the anomalies.

Then we have computed the average volume V of the detected anomalies (in terms of associated traffic) and we have synthetically produced 155 distinct anomalies, spanning either a single or multiple (up to three) time-bins. These anomalies are represented by n traffic

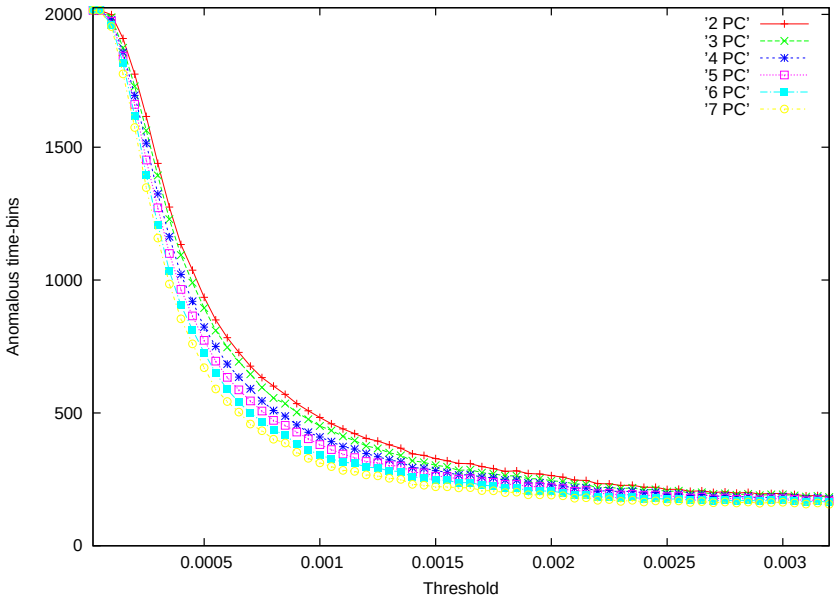

Figure 4.23: *Anomalies Detected - K-L divergence*

flows (n is randomly chosen between 1 and 5), characterized by a volume of traffic given by V multiplied by a scaling factor k that randomly assumes a value in the range $[0.9, 1.1]$.

Eventually, the obtained anomalies have been randomly inserted in the traffic traces (according to a uniform distribution).

The presented results represent the average of ten independent runs of this generation procedure.

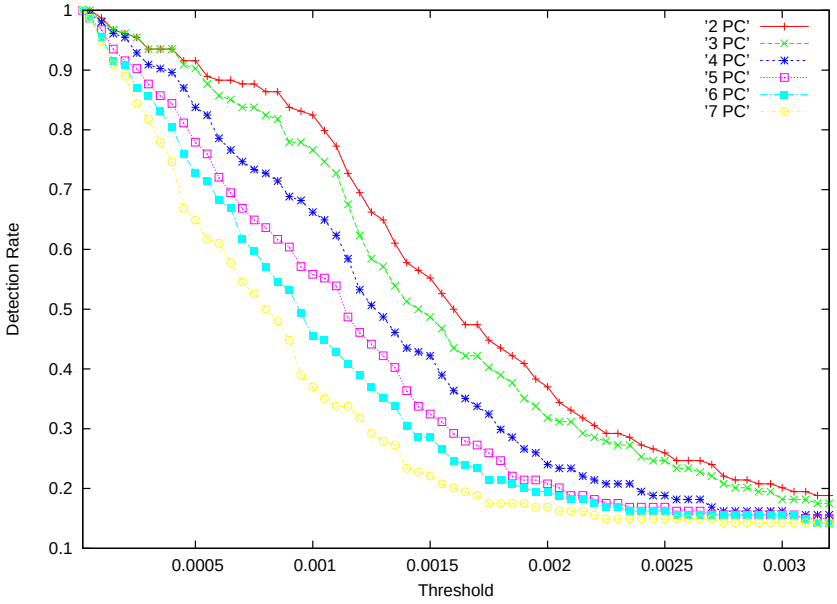

Figure 4.24: *Detection Rate - K-L divergence*

CHAPTER 5

CONCLUSIONS AND PERSPECTIVES

In this thesis we have shown a view over the arising technologies used to build the IoT, involving new perspectives for Home Automation and IoT security. Chapter 2 addressed the design and implementation of an SIP-based Home Gateway for remote control of *Smart Object* in a domotic environment. The chapter presented an architecture for realizing a domotics system with heterogeneous devices and user terminals. The architecture is based on the use of SIP as the common control plane and is centered on the SIP-based Home Gateway. In our vision, the proposed architecture has several aspects that might contribute in advancing the current state of the art in domotics systems. Heterogeneous sensor and actuators devices (DSANs) are supported through different translation entities embedded into the SHG. The DFA is the glue between these entities and the SIP world, and simultaneously allows to separate the implementation of the SIP and DSAN interfaces. Compatibility with existing SIP devices and network elements have also been guaranteed. We have proved the feasibility of our architecture by means of a working prototype, which includes the SHG, a standard ZigBee network, an Enhanced and a Legacy Client, a newly defined SIP event package, and a customized Event State Compositor. The possible synergies between ZigBee and SIP have also been illustrated.

Chapter 3 asses the performances of such a device. Performance evaluation is performed considering two main problems for such a device: scalability to a large number of request per seconds and interference/coexistence of devices belonging to different technologies/standard (ZigBee, BLuetooth, and Wi-Fi). The performance of the SHG has been assessed in terms of served user requests per second, processing delay, average and peak service delay. The effect of having the three wireless interfaces on the same board that operate on the same frequency

band has also been evaluated.

The results proved the SHG ability to support a considerable number of requests per second. Our experiments showed a tremendous performance degradation when ZigBee and Wi-Fi are on adjacent channels. Nevertheless, by means of a proper configuration, we have also proved that it is possible to avoid command losses.

The latter chapter 4 faces the previously introduced problem security for the IoT. In particular we have developed an Anomaly Based Intrusion Detection System. To asses the performances of the proposed method we used freely available datasets of traffic captured over the Internet. In this chapter we have presented a novel anomaly detection method, based on the use of PCA. With respect to the "classical" approaches, our system achieves several improvements due to the use of K-L divergence and multi time-scale analysis. Moreover our system also goes beyond the state of the art, by introducing a method for identifying the traffic flows responsible for an anomaly detected at the aggregate level.

BIBLIOGRAPHY

[1] "Flow-Tools Home Page," http://www.ietf.org/rfc/rfc3954.txt.

[2] "Jitsi." [Online]. Available: http://jitsi.org/

[3] "Kamailio (OpenSER) the Open Source SIP Server." [Online]. Available: http://www.kamailio.org/

[4] "Linphone, open-source voip software." [Online]. Available: http://www.linphone.org/

[5] "The Internet2 Network," http://www.internet2.edu/network/.

[6] "The IP Telecommunications Portal." [Online]. Available: http://www.iptel.org/

[7] "IEEE Standard 802.15.4-2006," Sept. 2006.

[8] "IEEE Standard 802.11-2007," Dec. 2007.

[9] R. Acker, S. Brandt, N. Buchmann, T. Fugmann, and M. Massoth, "Ubiquitous home control based on SIP and presence service," in *Proceedings of the 12th International Conference on Information Integration and Web-based Applications and Services (ii-WAS)*. ACM, 2010, pp. 759–762.

[10] M. Alia, A. Bottaro, F. Camara, and B. Hardouin, "On the Design of a SIP-Based Binding Middleware for Next Generation Home Network Services," in *Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE.*, 2008, pp. 497–514.

[11] B. Bertran, C. Consel, W. Jouve, H. Guan, and P. Kadionik, "SIP as a Universal Communication Bus: A Methodology and an Experimental Study," in *IEEE International Conference on Communications (ICC)*, May 2010.

[12] B. Bertran, C. Consel, P. Kadionik, and B. Lamer, "A SIP-based home automation platform: an experimental study," in *13th International Conference on Intelligence in Next Generation Networks (ICIN)*, Oct. 2009.

[13] D. Bonino, E. Castellina, and F. Corno, "Automatic domotic device interoperation," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 499–506, May 2009.

[14] C. Borean, G. Fici, and S. Guerra, "Wireless Sensor Network Center: a ZigBee Network Management System - How to connect ZigBee world to enable the Internet of Things," in *4th European ZigBee Developers' Conference*, 2010.

[15] Y. Bouzida, F. Cuppens, N. Cuppens-Boulahia, and S. Gombault, "Efficient intrusion detection using principal component analysis," in *3ème conférence sur la sécurité et architectures réseaux, La Londe, France, juin*. RSM - Dépt. Réseaux, Sécurité et Multimédia (Institut Télécom-Télécom Bretagne), 2004.

[16] A. Brown, M. Kolberg, D. Bushmitch, G. Lomako, and M. Ma, "A SIP-based OSGi device communication service for mobile personal area networks," in *3rd IEEE Consumer Communications and Networking Conference (CCNC)*, Jan. 2006, pp. 502–508.

[17] C. Callegari, L. Gazzarrini, S. Giordano, M. Pagano, and T. Pepe, "A novel multi timescales pca-based anomaly detection system," in *Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2010 International Symposium on*, july 2010, pp. 156 –162.

[18] C. Callegari, R. G. Garroppo, S. Giordano, and M. Pagano, "Security and delay issues in SIP systems," *International Journal of Communication Systems*, vol. 22, no. 8, pp. 1023–1044, 2009.

[19] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging," RFC 3428, Internet Engineering Task Force, Dec. 2002.

[20] V. Chatzigiannakis, S. Papavassiliou, and G. Androulidakis, "Improving network anomaly detection effectiveness via an integrated multi-metric-multi-link (m3l) pca-based approach," *Security and Communication Networks*, vol. 2, no. 3, pp. 289–304, 2009. [Online]. Available: http://dx.doi.org/10.1002/sec.69

[21] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), Internet Engineering Task Force, Oct. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3954.txt

[22] D. J. Cook, J. C. Augusto, and V. R. Jakkula, "Ambient intelligence: Technologies, applications, and opportunities," *Pervasive and Mobile Computing Journal*, vol. 5, no. 4, pp. 277–298, 2009.

[23] G. Cormode and S. Muthukrishnan, "An improved data stream summary: the count-min sketch and its applications," *Journal of Algorithms*, vol. 55, no. 1, pp. 58 – 75, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/B6WH3-4BM8Y1G-1/2/71b7980bb85b570bc57ee73f8afcd62f

[24] CounterPath Corporation, "Bria 3 softphone." [Online]. Available: http://www.counterpath.com/bria.html

[25] Freescale Semiconductor Inc., "MC1322x Platform in a Package." [Online]. Available: http://www.freescale.com

[26] R. Garroppo, L. Gazzarrini, S. Giordano, and L. Tavanti, "Experimental assessment of the coexistence of Wi-Fi, ZigBee, and Bluetooth devices," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, Lucca, Italy, june 2011, pp. 1–9.

[27] F. Genova, F. Bellifemine, M. Gaspardone, M. Beoni, A. Cuda, and G. P. Fici, "Thermal and energy management system based on low cost Wireless Sensor Network Technology, to monitor, control and optimize energy consumption in Telecom Switch Plants and Data Centres," in *4th International Conference on Telecommunication-Energy Special Conference (TELESCON)*, May 2009.

[28] P. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan, "IrisNet: an architecture for a worldwide sensor Web," *IEEE Pervasive Computing*, vol. 2, no. 4, pp. 22–33, Oct.-Dec. 2003.

[29] I. Howitt, V. Mitter, and J. Gutierrez, "Empirical study for IEEE 802.11 and Bluetooth interoperability," in *IEEE Vehicular Technology Conference, Spring*, vol. 2, May 2001, pp. 1109–1113.

[30] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S: A publish/subscribe protocol for Wireless Sensor Networks," in *3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE)*, Jan. 2008, pp. 791–798.

[31] S. Krishnamurthy, "TinySIP: Providing Seamless Access to Sensor-based Services," in *3rd Annual International Conference on Mobile and Ubiquitous Systems*, July 2006.

[32] S. Krishnamurthy and L. Lange, "Enabling Distributed Messaging with Wireless Sensor Nodes Using TinySIP," in *Ubiquitous Intelligence and Computing*, ser. Lecture Notes in Computer Science, J. Indulska, J. Ma, L. Yang, T. Ungerer, and J. Cao, Eds., vol. 4611, 2007, pp. 610–621.

[33] B. Kumar and M. Rahman, "Mobility support for universal plug and play (UPnP) devices using session initiation protocol (SIP)," in *Proceedings of the 3rd IEEE Consumer Communications and Networking Conference (CCNC)*, vol. 2, Jan. 2006, pp. 788–792.

[34] A. Lakhina, "Diagnosing network-wide traffic anomalies," in *In ACM SIGCOMM*, 2004, pp. 219–230.

[35] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *In ACM Internet Measurement Conference*, 2004, pp. 201–206.

[36] ——, "Mining anomalies using traffic feature distributions," Tech. Rep., 2005.

[37] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *In ACM SIGMETRICS*, 2004, pp. 61–72.

[38] X. Li, F. Bian, M. Crovella, C. Diot, R. Govindan, G. Iannaccone, and A. Lakhina, "Detection and identification of network anomalies using sketch subspaces," in *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2006, pp. 147–152.

[39] M. ling Shyu, S. ching Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," in *In IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with ICDM03*, 2003, pp. 172–179.

[40] T. Luckenbach, P. Gober, S. Arbanowski, A. Kotsopoulos, and K. Kim, "TinyREST: a protocol for integrating Sensor Networks into the Internet," in *Workshop on Real-World Wireless Sensor Networks (REALWSN)*, June 2005.

[41] R. Musaloiu-Elefteri and A. Terzis, "Minimising the effect of WiFi interference in 802.15.4 wireless sensor networks," *IJSNet*, vol. 3, no. 1, pp. 43–54, 2008.

[42] A. Niemi, "Session Initiation Protocol (SIP) Extension for Event State Publication," RFC 3903, Internet Engineering Task Force, Oct. 2004.

[43] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of pca for traffic anomaly detection," *SIGMETRICS Perform. Eval. Rev.*, vol. 35, no. 1, pp. 109–120, 2007.

[44] A. B. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification," RFC 3265, Internet Engineering Task Force, June 2002.

[45] J. Rosenberg, "A Presence Event Package for the Session Initiation Protocol (SIP)," RFC 3856, August 2004.

[46] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261, Internet Engineering Task Force, June 2002.

[47] H. Schulzrinne, X. Wu, S. Sidiroglou, and S. Berger, "Ubiquitous computing in home networks," *IEEE Communications Magazine*, vol. 41, no. 11, pp. 128–135, 2003.

[48] S. Y. Shin, H. S. Park, S. Choi, and W. H. Kwon, "Packet Error Rate Analysis of ZigBee Under WLAN and Bluetooth Interferences," *IEEE Transactions on Wireless Communications*, vol. 6, no. 8, pp. 2825–2830, 2007.

[49] A. Sikora and V. Groza, "Coexistence of IEEE 802.15.4 with other Systems in the 2.4 GHz-ISM-Band," in *IEEE Instrumentation and Measurement Technology Conference (IMTC)*, vol. 3, May 2005, pp. 1786–1791.

[50] K. Sohraby, D. Minoli, and T. Znati, *Wireless Sensor Networks: Technology, Protocols, and Applications*. John Wiley & Sons, Inc., May 2007.

[51] M. Thorup and Y. Zhang, "Tabulation based 4-universal hashing with applications to second moment estimation," in *SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2004, pp. 615–624.

[52] W. Wang and R. Battiti, "Identifying intrusions in computer networks with principal component analysis," in *ARES '06: Proceedings of the First International Conference on Availability, Reliability and Security*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 270–279.

[53] W. Wang, X. Guan, and X. Zhang, "A novel intrusion detection method based on principal component analysis," in *in Computer Security, Advances in Neural Networks, International IEEE Symposium on Neural Networks*. Lecture, 2004, pp. 657–662.

[54] Wireshark Foundation, "Wireshark network protocol analyzer." [Online]. Available: http://www.wireshark.org

[55] ZigBee Alliance, "ZigBee Specification," 2007. [Online]. Available: http://www.zigbee.org/

[56] ——, "ZigBee Gateway Standard," 2010. [Online]. Available: http://zigbee.org/Standards/ZigBeeNetworkDevices/Overview.aspx