Mohamed Khider University
of Biskra Faculty of Sciences
and Technology Department
of Electrical engineering

MASTER MEMORY

Electrical Engineering
Telecommunications
Networks and Telecommunications

Submitted and Defended by:

Gherbia Amel

On: Saturday, 6 July 2019

The implementation of security and privacy in IoT device

Board of Examiners:

Mr.	Guesbaya Tahar	**MCA**	University of Biskra	**President**
Ms.	Ouarhlent Saloua	**MCB**	University of Biskra	**Examiner**
Ms.	Zehani Soraya	**MCB**	University of Biskra	**Supervisor**

Academic Year: 2018 - 2019

الجمهورية الجزائرية الديمقراطية الشعبية

People's Democratic Republic of Algeria

وزارة التعليم العالي و البحث العلمي

Ministry Of Higher Education and Scientific Research

Mohamed Khider Biskra University

Faculty of Sciences and Technology

Electrical Engineering Department

Field: Telecommunication

Option: Networks and Telecommunication

A Dissertation for the Fulfillment of the Requirement of a

Master's Degree

The implementation of security and privacy in IoT device

Presented by:

Gherbia Amel

Favorable opinion of the supervisor:

Ms. Zehani Soraya

Favorable opinion of the Jury President

Dr.Guesbaya Tahar

Stamp and signature

الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic Republic of Algeria
وزارة التعليم العالي و البحث العلمي
Ministry Of Higher Education and Scientific Research

Mohamed Khider Biskra University
Faculty of Sciences and Technology
Electrical Engineering Department
Field: Telecommunication
Option: Networks and Telecommunication

Theme:

The implementation of security and privacy in IoT device

Proposed by: Gherbia Amel
Directed by: Ms. Zehani Soraya

ABSTRACT

IoT devices are poised to become more pervasive in our lives than mobile phones and will have access to the most sensitive personal data such as social security numbers and banking information. As the number of are also exponentially multiplied. While security solutions won't even run on most embedded devices. It must secure the data stored by the device, secure communication and protect the device from cyber-attacks.

The goal of our project is discuss how to secure our IoT device which presented in that research as a Raspberry Pi by using Firewall, IDS, and SSL/TLS. First it will connect to Losant platform (IoT platform) and we will make sure it works perfectly.

Then simply we will take all the work on the Raspberry Pi, which takes on three steps: installing Firewall, IDS, and prove the default existing of SSL/TLS protocol.

Finally we tried to scan the traffic all between them to prove that no one will try to hack our Raspberry Pi.

Keywords: IoT, Firewall, IDS, SSL/TLS.

DIDICATION

For who be a part of my life even with a moment

For who feed my passion to be a special person just as I'm

For who stand and fight for Ambition, faith and the last hope of last second that changes everything.

ACKNOWLEDGEMENT

First and foremost, praises and thanks to the God, my creator, my source of inspiration, knowledge and understanding, the Almighty, for His blessings throughout my research work to complete the research successfully.

I would like to express my deep and sincere gratitude to my research supervisor, Dr.Zehani Soraya for her sincerity and motivation have deeply inspired me. She has taught me the methodology to carry out the research and to present the research works as clearly as possible. I would also like to thank her for her friendship, empathy. And I thank all my teachers and Jury members to have accepted to participate in this defense of the master.

I am extremely grateful to my two parents for their love, prayers, caring and sacrifices for educating and preparing me for my future, and for the women who made me as I am "mama sadia", Also I express my thanks to my lovely sisters; Amina and Fairouz, rare brothers; Ayoub, Haithem, Saad and Youcef. Because they are just the motivation part from all my days, Allah bless them.

My completion of this project could not have been accomplished without the support of those special people in my life: Zineb, Hadjer, Sofia, Hafsa, Amoula, Mona, Salma, Rim, Imane and all my classmates.

My special thanks go to all the people who have supported me, encouraged me and just help me even with a few words to stand up again and again all the time.

TABLE OF CONTENTS

Chapter I: An overview of internet of things (IoT)

Chapter II: IoT security

Chapter III: implementation part

TABLES LIST

FIGURES LIST

ABBREVIATIONS LIST

BLE: Bluetooth Low Energy.

COAP: Constrained Application Protocol.

DTLS: Datagram Transport Layer Security.

EDI: Electronic Data Interface.

EU GDPR: European Union's General Data Protection Regulation.

FTP: File Transfer Protocol.

GPS: Global Position System.

GPRS: General Packet Radio Service.

HQTT: Message Queue Telemetry Transport.

HTTP: Hypertext Transfer Protocol.

HTTPs: Hypertext Transfer Protocol secure.

HVAC: Heating, ventilation, and air conditioning.

IoT: Internet of Things.

ISPs: International Ship and Port Facility Security**.**

IPSs: International Prostate Symptom Score.

IRC: Internet Relay Chat.

NATO: North Atlantic Treaty Organization**.**

NIST: National Institute of Standards and Technology.

OASIS: Organization for the Advancement of Structured Information Standards.

QoS: Quality of Services.

RFID: Radio Frequency Identification.

PC: personnel computer.

SMTP: Simple Mail Transfer Protocol.

SSH: Secure Shell.

UFW: The Uncomplicated Firewall.

VTY: Virtual teletype.

WiMax: Worldwide interoperability for Microwave access.

WiFi: Wireless Fidelity.

WSN: Wireless Sensor Networks.

ABSTRACT

IoT devices are poised to become more pervasive in our lives than mobile phones and will have access to the most sensitive personal data such as social security numbers and banking information. As the number of are also exponentially multiplied. While security solutions won't even run on most embedded devices. It must secure the data stored by the device, secure communication and protect the device from cyber-attacks.

The goal of our project is discuss how to secure our IoT device which presented in that research as a Raspberry Pi by using Firewall, IDS, and SSL/TLS. First it will connect to Losant platform (IoT platform) and we will make sure it works perfectly.

Then simply we will take all the work on the Raspberry Pi, which take on three steps: installing Firewall, IDS, and prove the default existing of SSL/TLS protocol.

Finally we tried to scan the traffic all between them to prove that no one will try to hack our Raspberry Pi.

Keywords: IoT, Firewall, IDS, SSL/TLS.

الملخص

تاخذ أجهزة إنترنت الأشياء الوجهة لتكون الأكثر انتشارًا في حياتنا من الهواتف المحمولة ما سيسمح لها من الوصول إلى اكثر البيانات الشخصية حساسية مثل أرقام الضمان الاجتماعي والمعلومات المصرفية. كما سيكون العدد أضعافا مضاعفة. حتى انه لا يتم تشغيل الحلول الأمنية على معظم الأجهزة المدمجة. بينما يجب تأمين البيانات المخزنة بواسطة الجهاز ، وتأمين الاتصالات وحماية الجهاز من الهجمات الإلكترونية .

الهدف من مشروعنا هو مناقشة كيفية تأمين جهاز إنترنت الأشياء الذي قدم في هذا البحث على انه Raspberry Pi باستخدام Firewall و IDS و .SSL / TLS . أولاً ، سيتم الاتصال بالمنصة Losant(منصةIoT)، وسوف نتأكد من أنها تعمل بشكل مثالي. ثم ببساطة سنتخذ جميع الأعمال على Raspberry Pi ، وذلك على ثلاث خطوات: تثبيت جدار الحماية ، IDS، وإثبات بروتوكول SSL / TLS. أخيرًا ، حاولنا فحص حركة المرور جميعها لإثبات أنه لن يحاول أحد اختراق Raspberry Pi لدينا.

كلمات المفتاحية: إنترنت الأشياء ، جدار الحماية, SSL / TLS, IDS.

General Introduction

General Introduction

The Internet of Things (IoT) represents the concept of a massive system where things on the Internet communicate through omnipresent sensors. Since the inception of the Internet of Things, consumers have connected smart devices to the network at an exponential rate, bringing us closer to a future where everyday things all interconnect. It is comprised of a wildly diverse range of device types- from small to large, from simple to complex – from consumer gadgets to sophisticated systems found in DoD (department of defense), utility and industrial systems.

But we have focus on security part in it which is simply keep our information or even our privacy, the domain of security Attacks on embedded device is increasing day by day. We just apply what NIST (National Institute of Standards and Technology) Defines of how system should be secured to be compliant with federal level security. It has federal security guidelines (SP 800-53). Security is perhaps the most complex and undeveloped area of network security.

So when designing a system, it is important to understand the potential threats to that system, and add appropriate defenses accordingly, as the system is designed and architected. It is important to design the product from the start with security in mind because understanding how an attacker might be able to compromise a system helps make sure appropriate mitigations are in place from the beginning.

Motivation

Usually maker ignore the security concerns because they are complicated, painful, and time consuming issues without any WOW effect. However, we are building systems that we may use every day. We should pay a little more attention to the security aspect of our stuffs. The level of security required for an embedded device varies dramatically depending upon the function of the device. Rather than asking if the device is secure and even it is the most important part from all IoT systems and can make all of it in danger.

Objective

IoT is gaining momentum and innovation is fuelling diverse IoT application, however security isn’t advancing as fast as it should be. Insecure IoT network are leveraged to initiate massive DDoS (Distributed denial of service) attacks triggering major financial losses and IP damage. This study intends to use tips to secure IoT device from bad hackers.

The memory research chapters

We split the project on three chapters, **first chapter**: it is an overview of IoT for gives us more information about the basic parts on it, this chapter is attempting to present IoT definition, applications, architecture and technologies.

Then **Chapter II**: in this chapter we dealing with IoT security, some of the most danger attack in it and the propose solution of secure IoT device.

Finally **Chapter III**: it is the practice of our project, it was the very hard part because we did many researches documentations, to handle these softwares and started experience with raspberry pi and Losant platform too.

Chapter I

An overview

of Internet of Things(IoT)

I.1 Introduction

Internet of things (IoT) is not just a term that implies things are depending on the internet for an application, but more widely available and immediately new rule of devices that is going to be “anything that can be connected, will be connected”. Conversations about the IoT are taking place all over technologies, platforms, sensors, new invent and even all the old part from networking industries.

This chapter is attempting to present IoT definition, applications, architecture and technologies from the low level up to the general details that it bases on.

I.2 History

The term Internet of Things is 16 years old but the actual term “Internet of Things” was coined by Kevin Ashton in 1999 during his work at Procter&Gamble. He said "I could be wrong, but I'm fairly sure the phrase "Internet of Things" started life as the title of a presentation I made at Procter & Gamble (P&G). Even though Kevin grabbed the interest of some P&G executives, the term Internet of Things did not get widespread attention for the next 10 years [1][2].

I.3 What is IoT?

The Internet of Things (IoT) has become a common news item and marketing trend. Beyond the hype, it has emerged as an important technology with applications in many fields and it has roots in several earlier technologies: pervasive information systems, sensor networks, and embedded computing. The term IoT system more accurately describes the use of this technology than does Internet of Things ,where the Most IoT devices are connected together to form purpose-specific systems; they are less frequently used as general-access devices on a worldwide network [3].

This new concept involves objects of our daily life, like clothes, cars, smart cards, which will be able to reveal information about themselves, interacting with each other and with the environment. it will therefore add an enormous range of new industrial opportunities to the software and hardware markets [4].

The purpose of IoT is to transform the way we live today by making intelligent devices around us perform daily tasks and chores. Smart homes, smart cities, smart transportation and infrastructure etc. they are the terms which are used in relevance with IoT [5].

I.4 IoT Applications

There are many applications domains of IoT, ranging from personal to enterprise environments [5], for that, The potentialities offered by the IoT make it possible to develop numerous applications based on it, of which only a few applications are currently de-played. In future, there will be intelligent applications for smarter homes and offices, smarter transportation systems, smarter hospitals, smarter enterprises and factories. In the following subsections, some of the important example applications of IoT are briefly discussed [6]. The IoT applications have seen rapid development in recent years due to the technologies of Radio Frequency Identification (RFID) and Wireless Sensor Networks (WSN). The RFID enables the tagging or labeling of every single device, so as to serve as the basic identification mechanism in IoT. Due to WSN, each "thing" i.e. people, devices etc, becomes a wireless identifiable object and can communicate among the physical, cyber, and digital world [7].

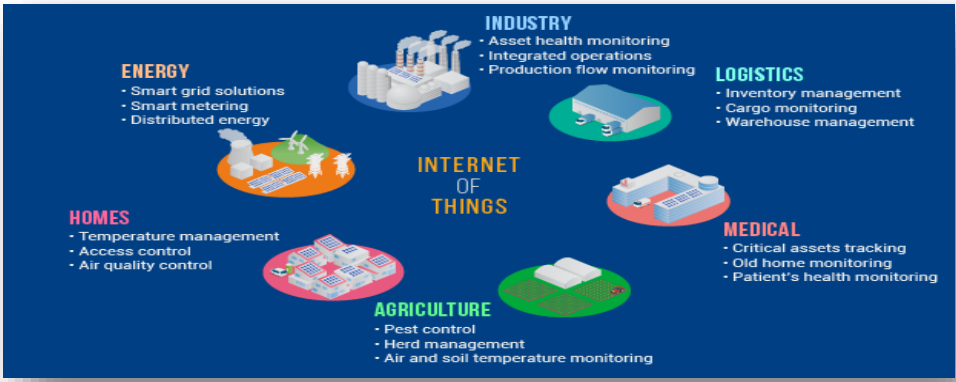


Figure I.1: Different IoT application [8].

I.4.1 Industry application

Industrial systems use sensors to monitor both the industrial processes themselves, the quality of the product, and the state of the equipment. An increasing number of electric motors, for example, include sensors that collect data used to predict impending motor failures [3]. In the future most industries may build their corresponding industry IoT. Cross-regional IoT are countless, while national IoT and global IoT arise. These Unit IoT and Ubiquitous IoT will cover all fields in our life. As an example there is, smart grid which is an important application area, and we cannot consider the development achievements of it all to be the benefits of IoT [9]. Figure I.2 represents a simple example of oil and gas industry with IoT. Oil and gas have been facing challenges, largely attributed to the antiquated and inefficient approach that many companies take to maintain assets and collect data [10].

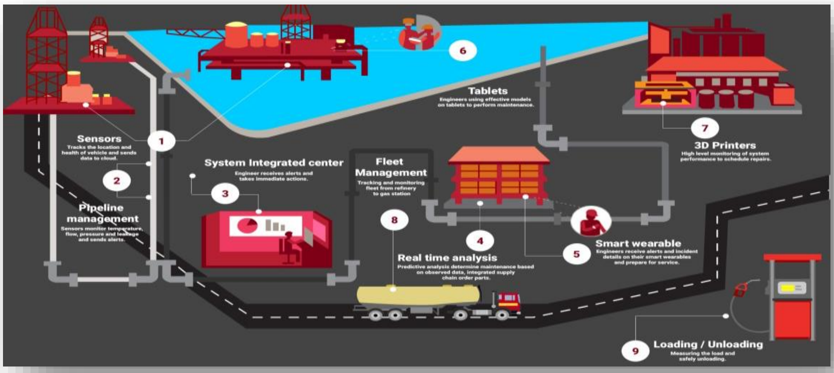


Figure I.2: Industrial IoT application in oil and gas industry [11].

I.4.2 Medical application

Among the panoply of applications enabled by the Internet of Things (IoT), smart and connected health care is a particularly important one. Networked sensors, either worn on the body or embedded in our living environments, make possible the gathering of rich information indicative of our physical and mental health as we see in the Figure I.3. Captured on a continual basis, aggregated, and effectively mined, such information can bring about a positive transformative change in the health care landscape [12]. It connect a wide range of patient monitoring sensors that may be located at the home, in emergency vehicles, the doctor’s office, or the hospital [3]. IoT helps in revolutionizing healthcare and provides pocket-friendly solutions for the patient and healthcare professional [13].

Figure I.3: Medical system as one of IoT application [7].

I.4.3 Smart City

The Smart Cities Council defines the smart city as “one that has digital technology embedded across all city functions [14].

In a Smart City, wireless sensor networks are the major sources of heterogeneous information generation. The information generated by different sensors often overlaps and is partial in nature. Addressing the challenges related to fusion of partial data is a research challenge [15]. The reason why it is so popular is that it tries to remove the discomfort and problems of people who live in cities. IoT solutions offered in the Smart City area solve various city related problems comprising of traffic, reduce air and noise pollution and help make cities safer [13].

Figure I.4: Smart cities infrastructure IoT wide [16].

I.5 IoT Architecture

Therefore IoT needs to develop a process flow for a definite framework over which an IoT solution is built [17]. The IoT Architecture generally classified on different sides on the basis of models, protocols or technologies.

I.5.1 IoT layers Model

In IoT, each layer is defined by its functions and the devices that are used in that layer. There are different opinions regarding the number of layers in IoT.[5] even so, after a look for some investigator [18],[19],[5]. IoT development depends on the technology progress and design of various new applications and business models [20].

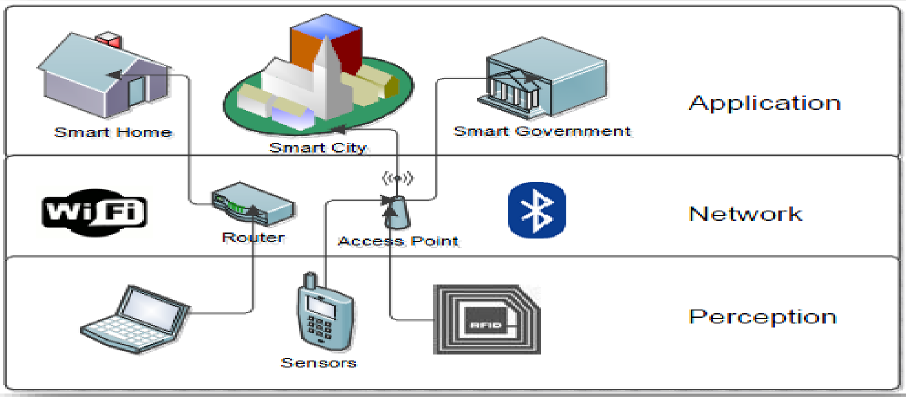


Figure I.5: Three-layer IoT Architecture [5].

The communication choose to focus on those 3 layers designated as Perception, Network, and Application layers. As shown in Figure I.5.

I.5.1.1 Perception layer

The perception layer is also known as the "Sensors" layer in IoT. The purpose of this layer is to acquire the data from the environment with the help of sensors and actuators [5]. It is the information origin and the core layer of IoT. All kinds of information of the physical world used in IoT are perceived and collected in this layer, by the technologies of sensors, Wireless Sensors Network (WSN), tags and reader-writers, RFID system, camera, Global Position System (GPS), intelligent terminals, Electronic Data Interface (EDI), objects, and so like [21].

I.5.1.2 Network Layer

This layer, also called transport layer, including access network and core network, provides transparent data transmission capability. By the existing mobile communication network, radio access network, Wireless Sensor Network (WSN) and other communications equipment, such as global system for mobile communications (GSM), General Packet Radio Service (GPRS), Worldwide interoperability for Microwave access (WiMax), Wireless Fidelity (WiFi), Ethernet, etc.., the information form perception layer can be sent to the upper layer. At the same time, this layer provides an efficient, reliable, trusted network infrastructure platform to upper layer and large scale industry application [21].

Its main purpose is to transmit data between devices and from the devices to receivers [19].

At this layer, cloud computing platforms, Internet gateways, switching, and routing devices etc. The network gateways serve as the mediator between different IoT nodes by aggregating, filtering, and transmitting data to and from different sensors [5].

I.5.1.3 Application Layer

Consists of the various applications and services that the IoT provides. Applications include smart cities, smart home, transportation, utilities and healthcare [19]. Guarantees the authenticity, integrity, and confidentiality of the data. At this layer, the purpose of IoT or the creation of a smart environment is achieved [5].

It is the interface between the end devices and the network and it is implemented through a dedicated application at the device end. Like for a computer, application layer is implemented by the browser. It is the browser which implements application layer protocols like Hypertext Transfer Protocol (HTTP), Hypertext Transfer Protocol secure (HTTPS), Simple Mail Transfer Protocol (SMTP) and File Transfer Protocol (FTP). Same way, there are application layer protocols specified in context to IoT as well.

This layer is responsible for data formatting and presentation. The application layer in the Internet is typically based on HTTP protocol. However, HTTP is not suitable in resource constrained environment because it is extremely heavyweight and thus incurs a large parsing overhead [22].

I.5.2 IOT protocols

The Internet of Things covers a huge range of industries and use cases that scale from a single constrained device up to massive cross platform deployments of embedded technologies and cloud systems connecting in real time. Tying it all together are numerous legacy and emerging communication protocols that allow devices and servers to talk to each other in new, more interconnected ways.

I.5.2.1 Message Queue Telemetry Transport (MQTT)

Cisco is continuing its long standing participation in OASIS (Organization for the Advancement of Structured Information Standards) by participating in the effort to Produce an MQTT standard [23].

MQTT is one of the most commonly used protocols in IoT projects. It stands for Message Queuing Telemetry Transport.

In addition, it is designed as a lightweight messaging protocol that uses publish/subscribe operations to exchange data between clients and the server. Furthermore, its small size, low power usage, minimized data packets and ease of implementation make the protocol ideal of the "machine to machine" or "Internet of Things" world. It is a TCP based publish-subscribe protocol developed by IBM and then open-sourced for messaging applications. In a publish-subscribe format, clients can either "publish" data on a specific topic to the server or "subscribe" to a topic where the server will automatically send new data on the topic to the subscriber once registered. MQTT combines the relatively high overhead and high QoS of TCP with the one to one, one to many, and many to one capabilities of a publish-subscribe format. Additionally, this protocol also allows clients to specify which telemetry topics are of interest and receive only data published through those topics [24].

I.5.2.1.A Quality of Services (QoS) Levels

MQTT supports three levels of QoS, specified by each published message and while subscribers are connecting.

- **QoS 0 (Maximum once):** This is also known as 'fire and forget'; no acknowledgement is sent by the receiver.

- **QoS 1 (At least once):** Each published message will be acknowledged using PUBACK; the sender retransmits a message if no acknowledgement is received within a time out by setting the DUP flag.
- **QoS 2 (Exactly once):** Sender and receiver exchange PUBLISH, PUBREC, PUBREL, PUBCOMP to ensure assured delivery of messages without duplicates [25]. There more the limited capacity and power of the device, and paid attention to preserve and exploit it properly. This is why the Message Queuing Telemetry Transport (MQTT) protocol in used in the process of exchanging commands and information between the IoT device and the user [26].

I.5.2.1.B MQTT Publisher-subscriber pattern (MQTT Broker, MQTT Client)

As described above MQTT is a message based protocol that uses publisher-subscriber pattern. The key component in MQTT is the MQTT broker. The main task of MQTT broker is dispatching messages to the MQTT clients ("subscribers"). In other words, the MQTT broker receives messages from publisher and dispatches these messages to the subscribers. While it dispatches messages, the MQTT broker uses the **topic** to filter the MQTT clients that will receive the message. The topic is a string and it is possible to combine the topics creating topic levels. A topic is a virtual channel that connects a publisher to its subscribers. MQTT broker manages this topic. Through this virtual channel, the publisher is decoupled from the subscribers and the MQTT clients (publishers or subscribers) do not have to know each other to exchange data. This makes this protocol highly scalable without a direct dependency from the message producer ("publisher") and the message consumer("subscriber") [27].

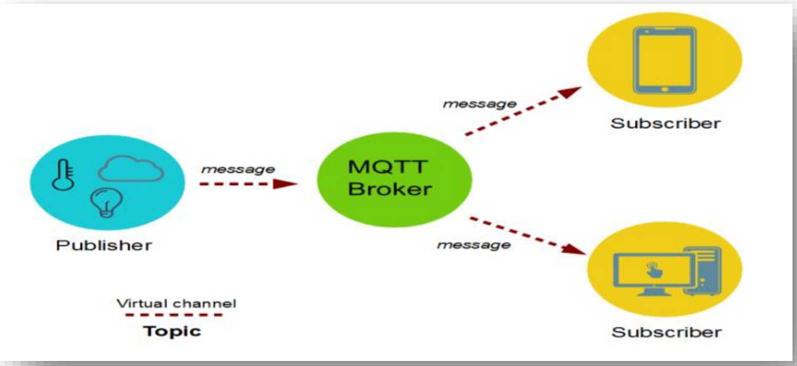


Figure I.6: The schema Describes the MQTT Architecture [28].

MQTT is very good at transferring data/commands form/to remote device over unstable connections. With MQTT, the broker is the center of the network. MQTT is well suited for application with unreliable network or where the nodes are in deep sleep.

I.5.2.2 Constrained Application Protocol (COAP)

CoAP is very well suited for client-server application over stable networks and it is good for client/server concept over stable connections, nodes can also execute “Commands”. MQTT and CoAP are well suited for low volume network and for low power devices, they are heavily used in IoT application and both of them can use secure lines, with CoAP, the preferred cryptographies method is DTLS.

Table I.1: Major differences between MQTT and CoAP.

	MQTT	**CoAP**
Application Layer	Single Layered completely	Single Layered with 2 conceptual sub layers (messages layer and Request Response Layer)
Transport Layer	Runs on TCP	Runs on UDP
Reliability Mechanism	3 Quality of Service levels	Confirmable massage, Non confirmable messages, Acknowledgements and retransmissions
Supported Architecture	Publish-Subscribe	Request-Response, Resource observe/publish-Subscribe

I.5.3 Heterogeneous wireless communication

A heterogeneous network is a network connecting computers and other devices with different operating systems and/or protocols [29].

I.5.3.1 Wireless Fidelity (WIFI)

WiFi is a high power communication technology that is currently the most widely used for indoor positioning. The transmission power of WiFi is typically 20 dBm with smartphones having a low WiFi scan rate of 1 Hz, both of which are considered drawbacks for IPSs [30]. The information used for positioning from WiFi technology is the RSS value sent from APs to mobile devices. Fingerprinting requires that a database of RSS values at different RPs be recorded in an offline stage [31].

I.5.3.2 ZigBee technology

ZigBee/RF4CE has some significant advantages in complex systems offering low-power operation, high security, robustness and high scalability with high node counts and is well positioned to take advantage of wireless control and sensor networks in M2M and IoT applications [32]. And it is a short-range, low-rate wireless network technology, and its physical layer and MAC layer protocol are almost the same as IEEE802.15.4. ZigBee Alliance, and IEEE802.15.4 is a communication protocol for short range wireless network, which was founded in August 2001, has enhanced the IEEE802.15.4; including the definition of the secure network layer and API are standardized so that it can support multiple architectures as well as providing high reliability wireless communication.

ZigBee is widely used in home automation, digital agriculture, industrial controls, and medical monitoring. The characteristic of ZigBee Wireless Sensor Network are shown in table2 [33].

Table I.2: The characteristic of ZigBee Wireless sensor Network [26].

Features	Description
Shorter delay	15ms—30ms
Low rate	1kB/S—250kB/S
Large capacity	Can support up to 255 devices
Multi-band	2.4GHz, 868MHz and 915 MHz
Security	Provides data integrity checking, and a AES-128 encryption algorithm
Low power Consumption	Two ordinary route 5th battery can be used 6 months to 2 years (standby mode)

I.5.3.3 Bluetooth

There are two branches of Bluetooth: traditional Bluetooth and Bluetooth Low Energy (known as BLE) [34].

- **Classic Bluetooth**: which is a derivative of the conventional Bluetooth (called Bluetooth Classic to differentiate from Bluetooth Smart), and new interfaces. In Bluetooth Classic, there are 79 channels, each with a channel bandwidth of 1 MHz and a raw symbol rate of 1 Msymbol/s. The modulation scheme could be Gaussian frequency shift keying (GFSK), quadrature phase shift keying (4PSK), or 8PSK [32].
- **Bluetooth Low Energy (BLE):** BLE (also called Bluetooth Smart) was first introduced in 2010 with the goal of expanding the application of Bluetooth for use in power-constrained devices such as wireless sensors and wireless controls. Not only does application in sensors and controls require low power consumption, but the amount of data transmission is small

and communication happens infrequently [35]. The BLE technology has propagation properties that allow us to determine values of proximity and distances, with a certain error in closed environments [36].

The lower transmission power of BTLE/BLE also contributes to the better performance of localization because it can reduce the multipath effect in some scenarios, BLE technology provides the variables that can be used to predict the position information. One of those variables is the received signal strength indicator (RSSI), which may give an estimate of the flight distance of the signal from the beacon to the receiver [37].

There are several commercially available BLE beacons, e.g., Aruba, Nordic or Estimote beacons.

Finally, the implementation of BLE technologies for positioning is still under study and is completely new [36].

I.5.3.4 Cellular network

Cellular technology is a great fit for applications that need high throughput data and have a power source of IoT application that requires operation over longer distances. It can take the advantage of GSM/3G/4G cellular communication capabilities it can provide reliable high speed connectivity to the internet .However, it needs high power consumption. Therefore, it is not suitable for M2M or local network communication. Cellular communication protocol is also used for many applications especially for applications that involve mobile devices. Cellular topology depends on various based technology [38].

I.6 IOT Technology

Internet of Things (IoT) technology enables organizations to easily connect to IoT sensor devices, collect and analyze data in real time, and gain new business process efficiencies [39].

I.6.1 RFID Technology

RFID enables applications to become “thing aware”, because RFID allows things to be identified by computer systems. As a result, RFID is one of the key technologies that the Internet of Things depends on. It is sensing technology that contains a microchip, an antenna and protective film (which cover the microchip and antenna) [40]. Adding RFID tags to part assembled goods, pallets and stillages, or finished items can speed manufacturing, logistics and service operations. Applications tracking assets can make a wide range of business activities more efficient.

RFID tags can be used to tell applications what things are, where things are, if things have moved, who moved them or used them. Connecting up the things a business works with to its computer applications can revolutionize its operations and, sometimes, even the products it offers [41]. The great appeal of RFID technology is that it allows information to be stored and read without requiring either contact or a line of sight between the tag and the reader. As better fabrication techniques are being developed, and as more advanced algorithms and circuit designs arise, the reliability and the read range of passive RFID continues to improve, and the cost continues to come down [42].

That technology is a major breakthrough in the embedded communication paradigm which enables design of microchips for wireless data communication [43]. There is a microchip and an antenna that surrounds the microchip in the tag. Electromagnetic waves emitted by the reader providing energy that activate the chip and data transfer is made the reader from tag without contact and wirelessly at defined distance [44].

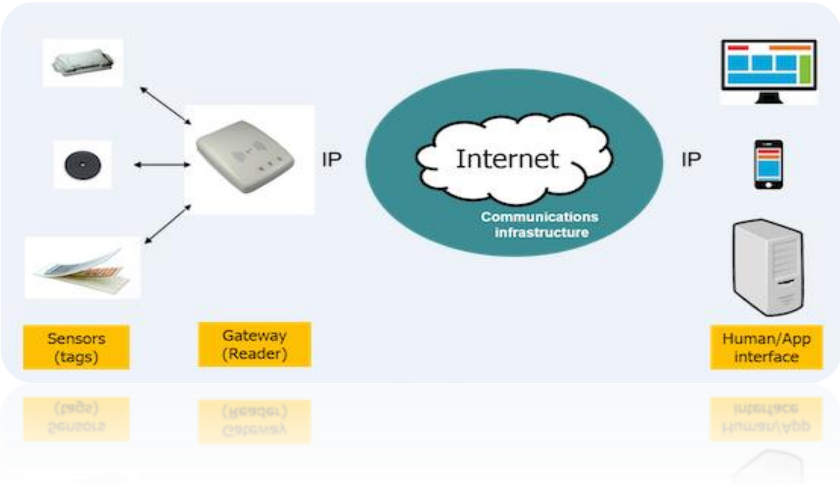


Figure I.7: RFID tags connect to the internet of things via Reader/Gatways.

I.6.2 Wireless Sensor Networks (WSN)

The past few years have witnessed increased interest in the potential use of wireless sensor networks (WSNs) in applications such as disaster management, combat field reconnaissance, border protection and security surveillance. Sensors in these applications are expected to be remotely deployed in large numbers and to operate autonomously in unattended environments. To support scalability, nodes are often grouped into disjoint and mostly non-overlapping clusters [45].

WSNs are characterized by high heterogeneity because they are compliant with different proprietary and non-proprietary solutions. This wide range of solutions is currently delaying a large-scale deployment of these technologies in order to obtain a virtual wide sensor network able to integrate all existing sensor networks. Interoperability among heterogeneous sensing systems and abstraction between low layers (i.e. hardware) and high layers (i.e. user applications) are thus very important challenges [46].

Wireless Sensor Networks (WSNs), namely networks consisting of tiny inexpensive autonomous devices equipped with sensors, can take measurements, locally store, handle sensed data, and can communicate to each other. WSNs are networks of things [47].

I.6.3 Middleware

Middleware is a software layer interposed between software applications to make it easier for software developers to perform communication and input/output. Its feature of hiding the details of different technologies is fundamental to free IoT developers from software services that are not directly relevant to the specific IoT application [48].

I.6.4 Cloud Computing

Cloud computing provides computing, storage, services, and applications over the Internet. In general, to render smartphones energy efficient and computationally capable, major changes to the hardware and software level are required. This entails the cooperation of developers and manufacturers. Mobile cloud computing is defined as an integration of cloud computing technology with mobile devices in order to make the mobile devices resource-full in terms of computational power, memory, storage, energy, and context awareness. The technology of Mobile Cloud computing is the outcome of interdisciplinary approaches combining mobile computing with cloud computing. Thus, this transdisciplinary domain is also referred as mobile cloud computing [42].

I.7 Conclusion

To sum up, this chapter tackled some principal parts from IoT to get the right direction, without forgetting the aforementioned aspects of IoT. In addition, these aspects are not enough to introduce IoT as a revolution of the century. But with the tens of billions of devices that have sensing or actuation capabilities, and are connected to each other via the internet. There are a big challenges to solve, one of the most important parts of that IoT challenges is the security of its devices in the beginning.

Chapter II

IoT security

I.1 Introduction

With increasing deployments of such Internet-connected devices in the house come increasing risks to both privacy and security: an eavesdropper can illegitimately snoop into family activities, even a legitimate entity (such as the device manufacturer) may be gathering data about users that they are not aware of, and of course a malicious entity may remotely take over control of the IoT devices, using it to either harm the household or to use it as a Launchpad for attacking other domains. Indeed in early 2014 it was revealed that there was a large scale attack on IoT devices including TVs and fridges [1], in which hackers were believed to have broken into more than 100,000 everyday consumer gadgets [2]. Anything that connects to the internet, even if it does not contain your medical records, poses a risk. The attacks were made possible by the large number of unsecured internet-connected digital devices, such as home routers and surveillance cameras [3].

However, companies cannot just give up on their IoT initiatives because of the nervousness and anxiety that security threats have stirred. All they need to do is follow certain risk-mitigation steps for enterprise IoT security and then fearlessly drive IoT growth [4]. So what do we do? Do we implement good practices to ensure the systems are not easily compromised, or do we risk lower security for ease of use? The bottom line is you must choose the security solution that best meets the need to protect the data and services without forcing the user to endure onerous practices and without making their lives difficult [5].

II.2 Security Challenges within IoT System

One of the more obvious challenges is keeping up with the production of IoT devices. There are more of these devices in use today than there are people in the world, but this number doesn't even come close to what's expected by 2020 [6]. There were a lot of security challenges coming from outside to be inside our network too and IoT deployments that deal with personal or sensitive data have a lot of it.

Confidentiality and Integrity: End-to-End encryption and unnoticed modification protection, while IoT data is in transit through a wireless multi-hop networks and at rest (stored in an IoT), is hard but necessary.

Availability: Compared with traditional Internet hosts, due to unattended IoT deployments it is easier to compromised IoT devices, and due to low-power wireless connectivity it easier to interfere with or jam IoT networks.

Authenticity: Source authentication is important but challenging because the limited IoT resources may not always permit digital signatures [7].

Data privacy and security continues to be the single largest issues in today's interconnected world. Data is constantly being harnessed, transmitted, stored and processed by large companies using a wide array of IoT devices, such as smart TVs, speakers and lighting systems, connected printers, HVAC systems, and smart thermostats [8].

Compliance: It is very challenging to ensure new EU GDPR compliance when ubiquitous environment-sensing IoT devices sense personal data.

Freshness: Often connection-less data transfer protocols are used in IoT; it is therefore necessary that old packets are not replayed [7].

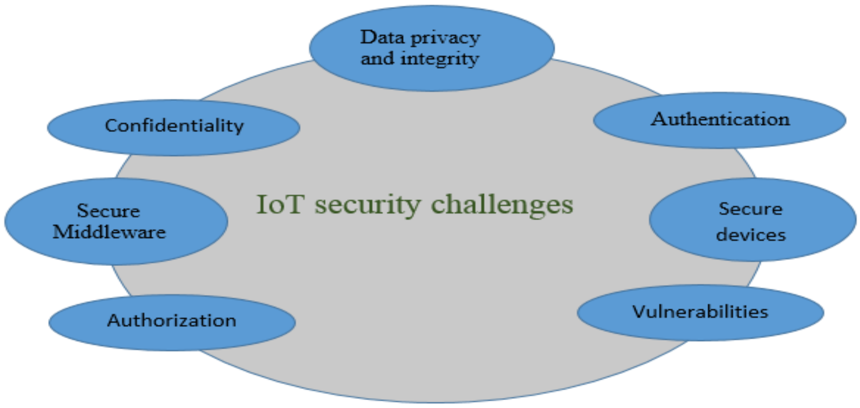


Figure II.1: IoT security challenges.

II.3 the CIA Triad: Confidentiality, Integrity, Availability

The CIA triad is a model that shows the three main goals needed to achieve information security. While a wide variety of factors determine the security situation of information systems and networks, some factors stand out as the most significant.

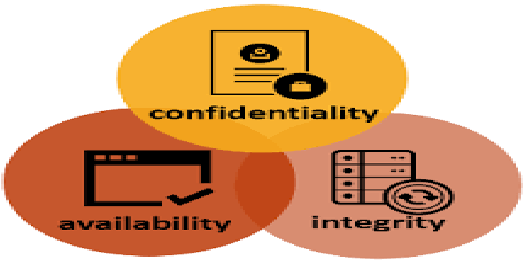


Figure II.2: CIA triad [9].

The assumption is that there are some factors that will always be important in information security. These factors are the goals of the CIA triad, as follows:

Confidentiality: Confidentiality is the protection of information from unauthorized access. This goal of the CIA triad emphasizes the need for information protection.

Integrity: The CIA triad goal of integrity is the condition where information is kept accurate and consistent unless authorized changes are made.

Availability: The CIA triad goal of availability is the situation where information is available when and where it is rightly needed [9].

II.4 IoT attacks

Software defined network separates a network is control plane from data plane, it becomes simple for network management. . But the novel network architecture faces some new serious security concerns [10].

An attack is an information security threat that involves an attempt to obtain, alter, destroy, remove, implant or reveal information without authorized access or permission [11].

II.4.1 Botnet

In terms of IoT risks, earlier this year the Reaper botnet targeted known vulnerabilities in IoT devices and hijacked them, including internet-connected webcams, security cameras, and digital video recorders. Each time a device is infected, the device spreads the malware to other vulnerable devices, expanding its reach [12].

A botnet can send spam, steal data, and allow remote access to devices without the owner’s knowledge. One botnet, Mirai, did so by scanning the internet for open telnet ports a protocol giving access to other devices within the same network [6].

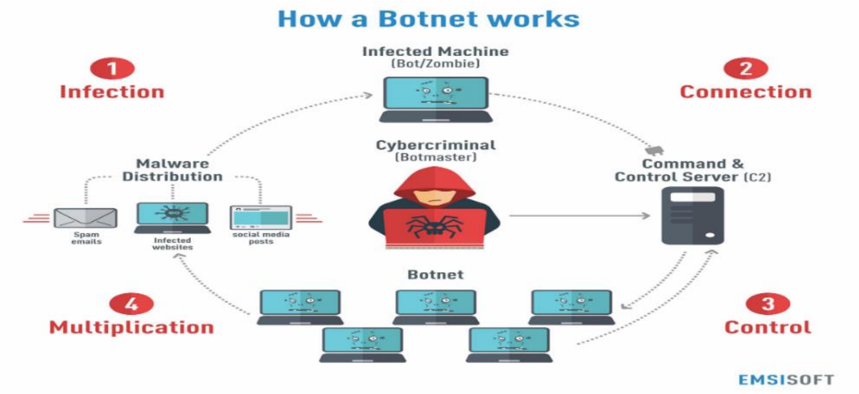


Figure II.3: How a botnet attack works [13].

A botnet is a collection of Internet connected computers whose security defenses have been breached and control ceded to a malicious party. Each such compromised device, known as a "bot," is created when a computer is penetrated by software from a malware distribution, otherwise known as malicious software.

The controller of a botnet is able to direct the activities of these compromised computers through communication channels formed by standards based network protocols such as Internet Relay Chat (IRC) and hypertext transfer protocol (http) [6].

II.4.2 Man-in-the-Middle-Attacks

Man-in-the-middle attacks (Mim- or Mitm attacks for short) behave like a proxy, but on an unintentional base. Some individuals therefore consider transparent proxies of ISPs a Man-in-the-Middle attack [14]. The attacker may get the access of privileged information or he may change the information before it reaches to the server. To prevent from the attack, the encryption mechanism with verification of digital certificates are used [15].

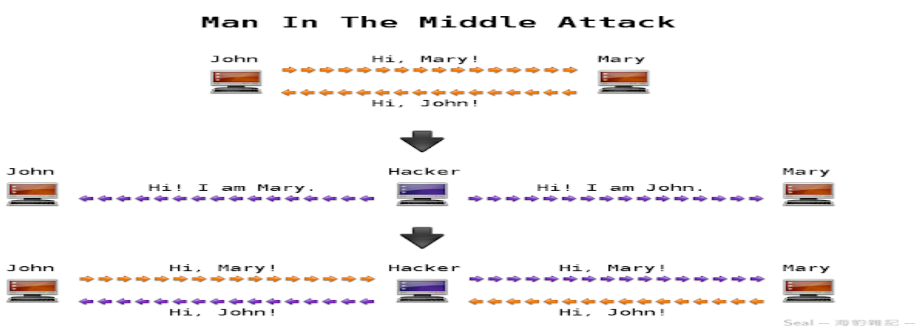


Figure II.4: Where MITM attack strategy [16].

II.4.3 Distributed denial of service (DDoS)

DDoS is a coordinated attack, launched using a large number of compromised hosts. At an initial stage, the attacker identifies the vulnerabilities in one or more networks for installation of malware programs in multiple machines to control them from a remote location. At a later stage, the attacker exploits these compromised hosts to send attack packets to the target machine(s), which is (are) usually outside the original network of infected hosts, without the knowledge of these compromised hosts [17]. It is an attack initiated and continued by some hundreds or even thousands of attackers. At the same time, the traffic disallows legitimate requests from reaching the DC and also depletes the bandwidth of the DC [18].

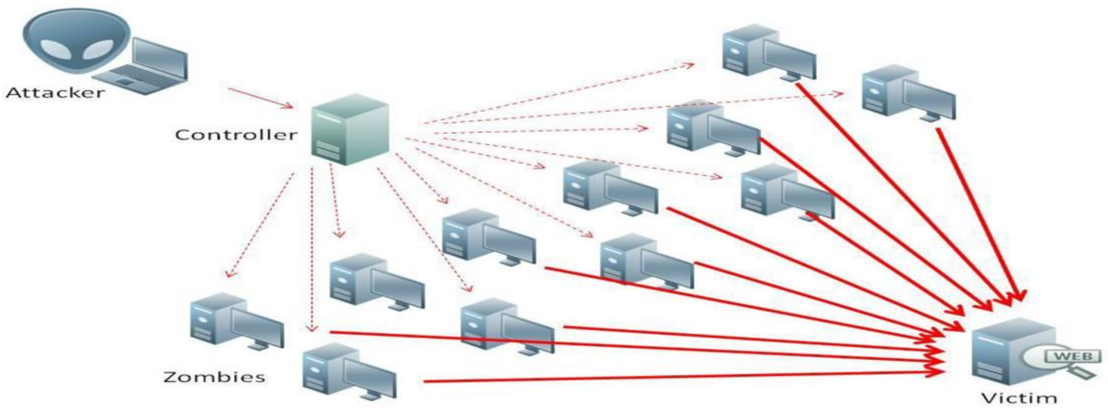


Figure II.5: How DDOS attack works [19].

Creation of DDoS attack in a denial-of-service (DoS) attack, an offender makes an attempt to stop legitimate users from accessing information or services. The foremost common and obvious kind of DoS attack happens once an assailant "floods" a network with data. Once you type a URL for a specific web site into your browser, you're causing asking to it site's PC server to look at the page. DoS/DDoS attacks cause a waste of the sources inside the host or networks, and make services work improperly. There are predominant classes of DoS/DDoS, good judgment and flooding assaults [18]. In DDoS attacks, multiple systems submit as many requests as possible to a single Internet computer or service, overloading it and preventing it from servicing legitimate requests [18].

II.4.4 IP spoof attack

Spoofing is a type of attack in which the attacker pretends to be someone else in order to gain access to restricted resources or steal information. This type of attack can take a variety of different forms; for instance, an attacker can impersonate the IP address of a legitimate user to get into their accounts. IP address spoofing, or IP spoofing, refers to the creation of IP packets with a forged source IP address, called spoofing, with the purpose of concealing the identity of the sender or impersonating another computing system.

IP spoofing is most frequently used in DoS attacks. In such attacks, the goal is to flood the victim with overwhelming amounts of traffic, and the attacker does not care about receiving responses to the attack packets [18].

II.5 IoT security

One of the most significant concerns in every application is always security. In IoT, that aspect is even more critical since a compromised object could perform all sorts of attacks, such as denial of service [20]. Network security is an integration of multiple layers of defenses in the network and at the network. Policies and controls are implemented by each network security layer. Access to networks is gained by authorized users, whereas, malicious actors are indeed blocked from executing threats and exploits [21].

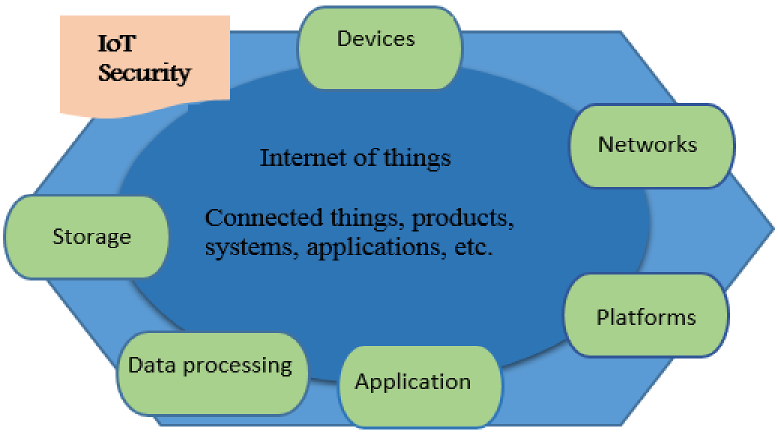


Figure II.6: Illustration of the essential components of IoT security.

II.6 IoT security levels

Figure II.7: The 3 levels of security in IoT.

II.6.1 Cloud security

The cloud layer refers to the software backend of the IoT solution i.e., where data from devices is ingested, analyzed and interpreted at scale to generate insights and perform actions. Security has always been a major topic of discussion when assessing the risk of using cloud versus on-premise solutions. However, for the Internet of Things cloud is viewed as a key enabler to widespread adoption. Cloud providers are expected to deliver secure and efficient cloud services by default, and protecting from major data breaches or solution downtime issues is becoming the norm. Sensitive information stored in the cloud (i.e., data at rest) must be encrypted to avoid being easily exposed to attacks and it is also beneficial to verify the integrity of other cloud platforms or third party applications that are trying to communicate with your cloud services to help protect against malicious activity. Digital certificates can play a key role for identification and authentication needs at the scale required for the IoT [22].

II.6.2 Connection security

While IoT device connect to the cloud, their connection will be risk by the attackers; for that, IoT gave that part of security more attention to make sure the privacy of all his user and servers. So the transport layer is the responsible part here and it use a lot of secure protocol for transport.

II.6.3 IoT devices security

IoT devices have heterogeneous capabilities in terms of processing power, storage, and energy; therefore, the definition of an IoT device varies across different sectors and use cases [7].

II.6.3.1 Firewall

Just like a physical construction that prevents the spread of fire, a firewall acts as a protective barrier between your network and the Internet. It is your first line of defense against viruses, bots and other malicious code constantly attempting to attack your devices. A solid firewall blocks the traffic that you have not specifically requested, while still allowing all legitimate communication with the outside world to run freely.

Think of it as a gatekeeper who won’t let anyone enter or leave your network if they don’t have permission. Firewalls work in a similar way. They scrutinize incoming or outgoing data for any potential threats to the system. Firewalls use a ‘wall of code’ that inspects every single ‘packet’ of information to decide whether it should be allowed or rejected based on a set of predefined security rules [23].

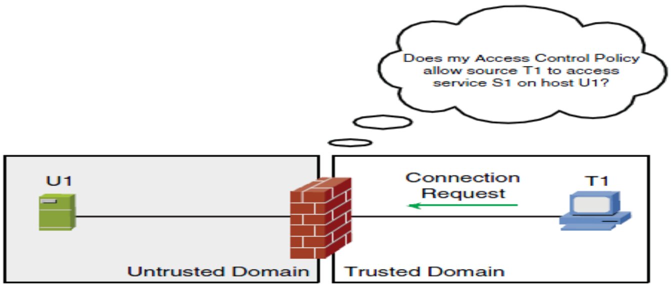


Figure II.8: Firewall and Security Domains [24].

II.6.3.2 Intrusion Detection Systems (IDS)

Intrusion Detection Systems (IDS) are devices or software applications that monitor network or system activities for malicious activities or policy violations and send reports to a management station.

IDS systems are being developed in response to the increasing number of attacks on major sites and networks, including those of the Pentagon, the White House, NATO, and the U.S. Defense Department. The safeguarding of security is becoming increasingly difficult because the possible technologies of attack are becoming ever more sophisticated; at the same time, less technical ability is required for the novice attacker, because proven past methods are easily accessed through the Web [25].

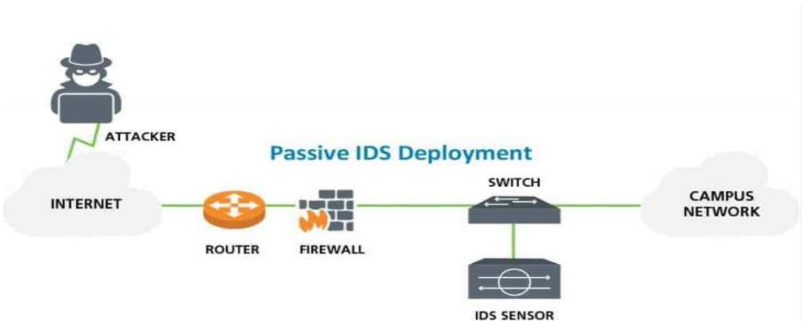


Figure II.9: Illustration of IoT where IDS modules are kept in Network [19].

In the IoT, resource-restrained elements are connected to the untrustworthy and undependable Internet via IPv6 and 6LoWPAN networks [1]. The wireless intrusions from both the Internet and the internal part of 6LoWPAN network are what make these susceptible to both external and internal attacks. The requirement of intrusion detection systems (IDSs) is considered to be necessary due to the probability of occurrence of these attacks.

Presently, due to the limitation of IDS approaches, which are being designed only for wireless sensor networks (WSNs) or traditional Internet, there is no IDS that can live up to the necessities of the IoT connected to IPv6 [26]. No network impact if there is a sensor failure or overload but response action cannot stop trigger.

II.6.3.3 Transport layer encryption (TLS)/ Secure Sockets Layer (SSL)

SSL/TLS is chosen to protect weak protocols. TLS is currently one of the most popular security protocols and is accepted as a secure means for secret communication.

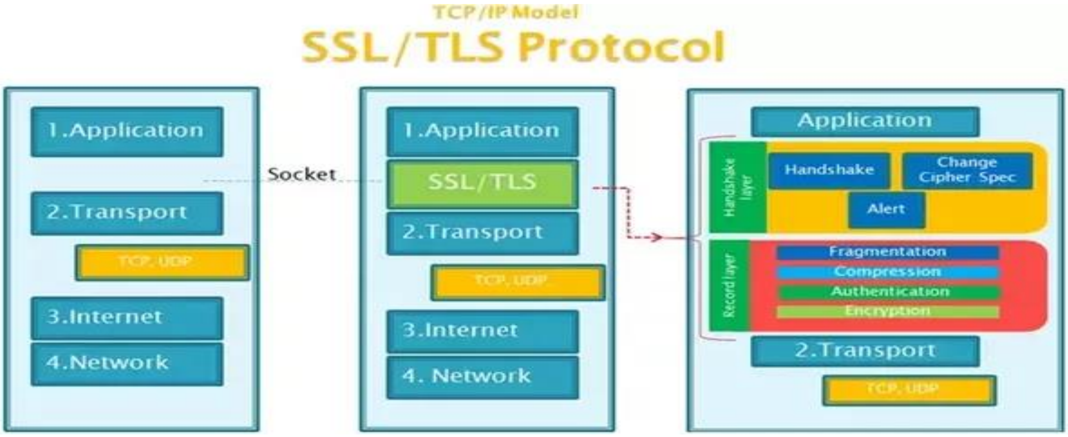


Figure II.10: SSL and TLS architecture [27].

However, TLS can only guarantee the protection of the communication to the respective end point, but not the security of the underlying protocols that e.g. implement the authentication of users with services. The sole purpose of TLS lies in the protection of communication between two trustworthy end points, i.e. both communication partners know and trust each other [28].

Secure Sockets Layer (SSL) was the most widely deployed cryptographic protocol to provide security over internet communications before it was preceded by TLS in 1999. Despite the deprecation of the SSL protocol and the adoption of TLS in its place, most people still refer to this type of technology as ‘SSL’. SSL provides a secure channel between two machines or devices operating over the internet or an internal network. One common example is when SSL is used to secure communication between a web browser and a web server [27].

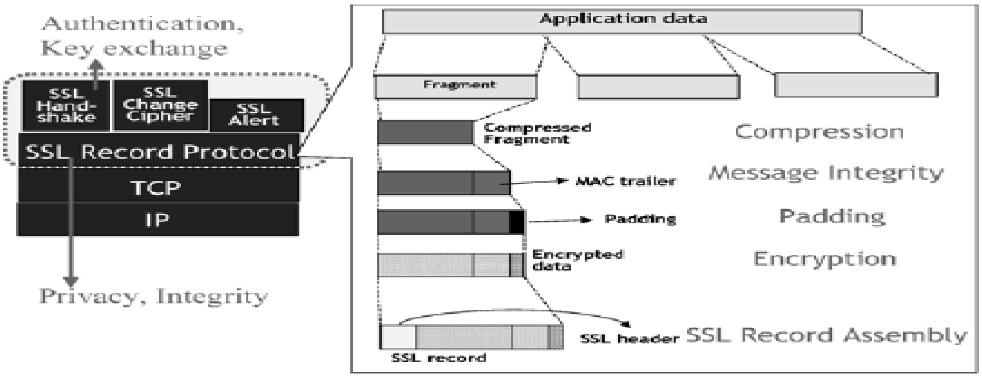


Figure II.11: SSL record protocol and his header [29].

II.7 Conclusion

The fast pace of development and nature of IoT device bring a variety of security and forensics challenges. In this chapter, we briefly presented major security and privacy issues along with potentially promising solutions.it included in this special issue offer state of art view of privacy, security and forensics challenges in IoT along with innovative solutions that paves the way towards secure sound deployment of IoT networks.

Chapter III

Implementation part

III.1 Introduction

Security is paramount for the safe and reliable operation of IoT connected devices, so our project bases on Raspberry Pi3 model B as a principal hardware requirement, which deal with defining software resource requirements and prerequisites that need to be installed on it, there is Losant platform too. We going to focus on both two parts, the various component of hardware and all programs of software requirements in this chapter, without forgetting about the details step by step.

III.2 Hardware requirements

In that part we will try to set up our materials that we used to implement our purpose of study.

III.2.1 Raspberry Pi

We use the Raspberry Pi as an IoT device here, which will connect to IoT platform “Losant”

III.2.1.1 Raspberry Pi Definition

The Raspberry pi is a low cost, credit sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in language like scratch and Python [1].

Raspberry Pi is a 900MHz single board computer running its own version of the Linux operating system.

You can attach a keyboard, mouse and monitor to a Raspberry Pi and use it as a regular computer if you like. What sets a Raspberry Pi apart from a PC is that it:

- is small—tiny in fact
- Low cost—
- Low power—uses around 2W
- Has GPIO (general purpose IO pins) these are used to connect external electronics

The Raspberry Pi's Ethernet port is main gateway for communication with other devices. It is auto-sensing which means that it may be connected to a router or directly to another computer (without the need for a crossover cable) [2].

III.2.1.2 The Raspberry Pi Components

CPU: Raspberry Pi 3 uses Broadcom BCM2837 SOC 64-bit quad-core ARM Cortex A53 (ARMv8 CPU) with 512KB shared L2 cache.

Memory: Provided with 1 GB of RAM.

Wi-Fi Support: 802.11n Wireless LAN.

Bluetooth: Supports Bluetooth 4.1 Bluetooth Low Energy (BLE) [3].

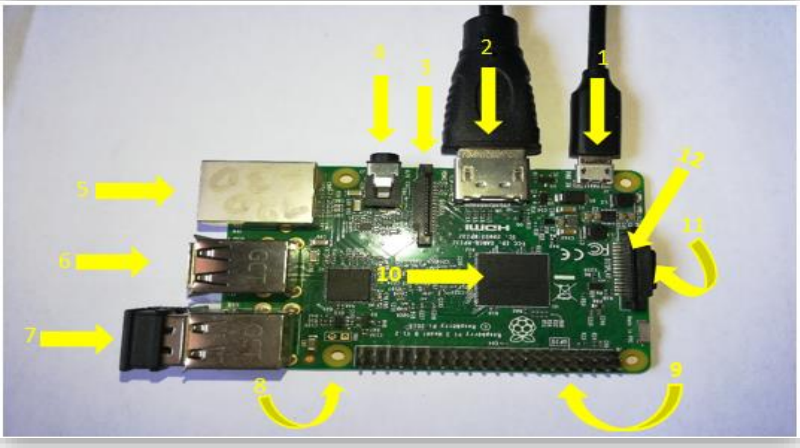


Figure III.1: Raspberry pi 3 elements.

- **1 :** Upgraded switched Micro USB power source up to 2.5A
- **2 :** Full size HDMI port
- **3 :** CSI Camera connector
- **4 :** Audio/video Jack: The 3.5mm audio jack on the board can be used with any standard pair of headphones.
- **5:** Ethernet port
- **6:** 2 USB 2.0 port
- **7:** 2 USB 2.0 port
- **8:** Reset-pins
- **9:** 40 pin GPIO header
- **10:** System-on-chip (SoC)
- **11:** Micro sd carte slot of 32 GB, The Micro SD Card will hold the operating system which will boot while we power on Raspberry Pi 3. In next tutorial, we will learn how to setup and prepare SD card with Raspbian OS.

Figure III.2: Micro SD carte.

- **12:** DSI Display Connector

III.2.1.3 Raspberry Pi advantages

- The Raspberry Pi is a small independent computer that runs on the various distribution of Linux operating system and can be programmed as needed.
- It has a very large working memory (RAM memory).
- It has expandable memory to store the data (up to 64GB).
- It works on multi operating processor (supports a set of instructions).
- It operates at speeds from 700 MHz to 1000 MHz.
- It has support for USB 2.0 which allows its expansion with a large number of peripherals.
- Depending of the needs it is possible to expand the Raspberry Pi with WiFi and Bluetooth adapters (power and range can be changed by changing the adapter).
- Expansion and communication with network devices over a LAN adapter are possible.
- It can be expanded with various prototype shields (Pi-Face, GSM/GPRS & GPS, GPIO expansion board, GertBoard).
- It is possible to form an expandable system with various electronic components (sensors and electronic circuits) using digital inputs and outputs, I 2C or SPI protocols.
- C, Python or object oriented languages such as C++ and Java can be used for programming of Raspberry Pi.
- It can be powered form battery or sollar cell.
- It can be run in server mode.
- A various web server can be installed and running on Raspberry Pi.

III.2.1.4 The main disadvantages of Raspberry Pi are:

- It does not have a real-time clock (RTC) with a backup battery but it can easily work around the missing clock using a network time server, and most operating systems do this automatically.
- It does not have a Basic Input Output System (BIOS) so it always boots from an SD card.
- It does not support Bluetooth or WiFi out of the box but these supports can be added by USB dongles.
- Unfortunately, most Linux distributions are still a bit picky about their hardware, so it should be first checked whether flavor of Linux supports particular device.
- It doesn’t have built-in an Analog to Digital converter. External component must be used for AD conversion.

III.2.2 The actuators

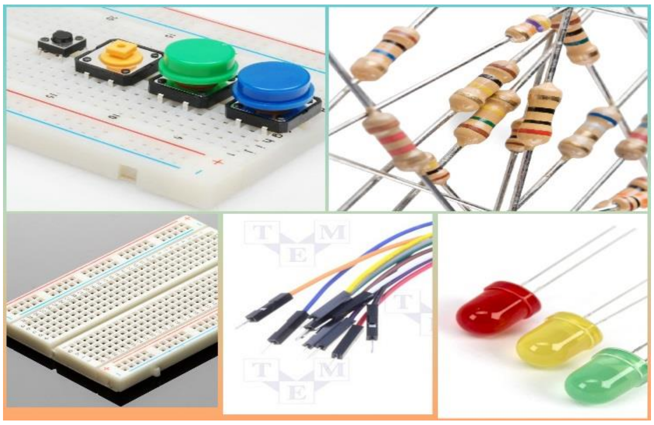


Figure III.3: Actuators components.

1. LEDs.
2. Jumper Wires (Male/Female, Male/Male).
3. Button (switch).
4. Resistors.
5. Breadboard.

III.3 Software requirements

This part is dedicated to the all softwares must be installed on the Raspberry Pi 3 and the softwares that will connect to it.

III.3.1 Losant platform

Losant is an easy-to-use and powerful enterprise IoT platform designed to help you quickly and securely build complex connected solutions. Losant uses open communication standards like REST and MQTT to provide connectivity from one to millions of devices. Losant provides powerful data collection, aggregation, and visualization features to help understand and quantify vast amounts of sensor data. Losant's drag-and-drop workflow editor allows you to trigger actions, notifications, and machine-to-machine communication without programming [4].

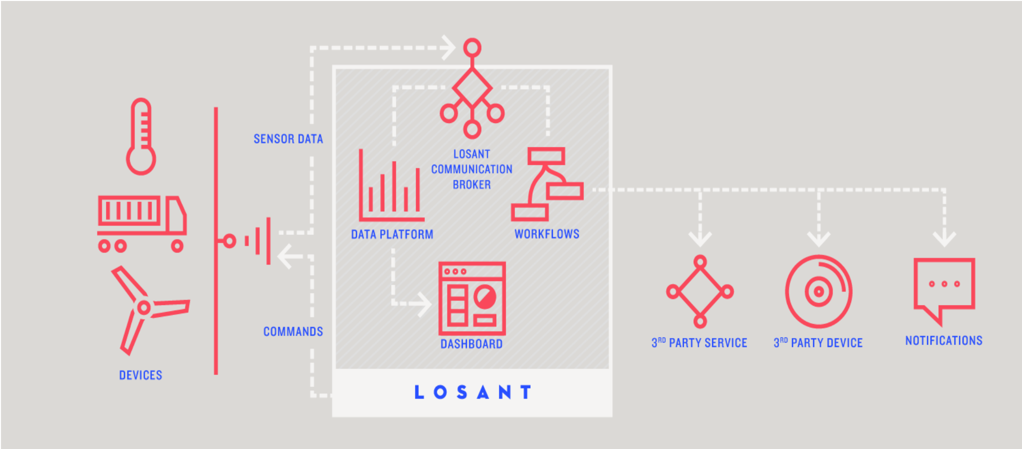


Figure III.4: Losant platform functions [4].

III.3.2 Postman

Postman's API Documentation feature lets you view private API documentation or share public API documentation in a beautifully formatted web page.

Postman generates and hosts browser-based API documentation for your collections automatically in real-time. Each collection has a private and public documentation view that Postman generates from synced data in the servers.

III.3.3 Wireshark

Wireshark is an open-source packet analyzer used for packet capture, analysis, and network troubleshooting. In this research, Wireshark is used to capture all live network traffic which is then 73 stored in .PCAP files for later analysis [5].

III.3.4 Putty

Putty is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. The name "PuTTY" has no official meaning.

PuTTY was originally written for Microsoft Windows, but it has been ported to various other operating systems. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and macOS, and unofficial ports have been contributed to platforms such as Symbian, Windows Mobile and Windows Phone [6].

III.3.5 Raspberry pi software

There for a few softwares that allow us to secure our raspberry pi.

III.3.5.1 Docker

Docker is a collection of interoperating software as a service and platform as a service offerings that employ operating system level virtualization to cultivate development and delivery of software inside standardized software packages called containers. The software that hosts the containers is called Docker Engine. It was first started in 2013 and is developed by Docker, Inc. The service has both free and premium tiers [7].

III.3.5.2 The Uncomplicated Firewall (ufw)

The Uncomplicated Firewall (ufw) is a frontend for iptables and is particularly well suited for host based firewalls. ufw provides a framework for managing netfilter, as well as a command line interface for manipulating the firewall. ufw aims to provide an easy to use interface for people unfamiliar with firewall concepts, while at the same time simplifies complicated iptables commands to help an adminstrator who knows what he or she is doing. ufw is an upstream for other distributions and graphical frontends [8].

III.3.5.3 Snort

Snort's open source network based intrusion detection/prevention system (IDS/IPS) has the ability to perform real time traffic analysis and packet logging on Internet Protocol (IP) networks. Snort performs protocol analysis, content searching and matching.

The program can also be used to detect probes or attacks, including, but not limited to, operating system fingerprinting attempts, semantic URL attacks, buffer overflows, server message block probes, and stealth port scans [9].

III.4 Implementation

This project based on Raspberry Pi3 Board, in this part we going to focus on two sides one it's the Hardware, the second it's the Software. The Hardware is all various components and elements that have been used in this project, The Software part is all programs and libraries.

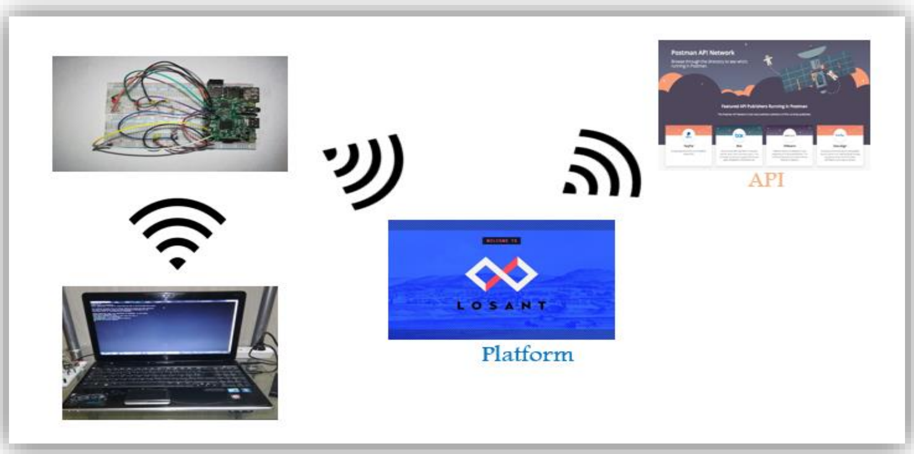


Figure III.5: The parts of connection and communication.

III.4.1 Setup our Raspberry Pi

First thing that we have to do is preparing the raspberry pi, by Setting Up it, we can use a Raspberry Pi without a keyboard, mouse and monitor, but to be able to do that we first need them to set up the Raspberry Pi so that can be accessed from a second computer over WiFi. We do not have to use a computer monitor, the Raspberry Pi will also connect to any TV that has an HDMI lead or VGA, by using HDMI-VGA converter.

Then installing the operating system, the Raspberry Pi was designed for Linux operating system, and many Linux distributions now have a version optimized for the Raspberry Pi. Two of most popular option is Raspbian.

Figure III.6: The components of set up the Raspberry Pi.

Then, we connected to a Raspberry Pi running in a standalone setup using a SSH connection. My client computer is a HP Pavilion Entertainment PC, which we use for all our manuscript production. It also allowed us to easily capture screenshots of the terminal window controlling the Raspberry Pi.

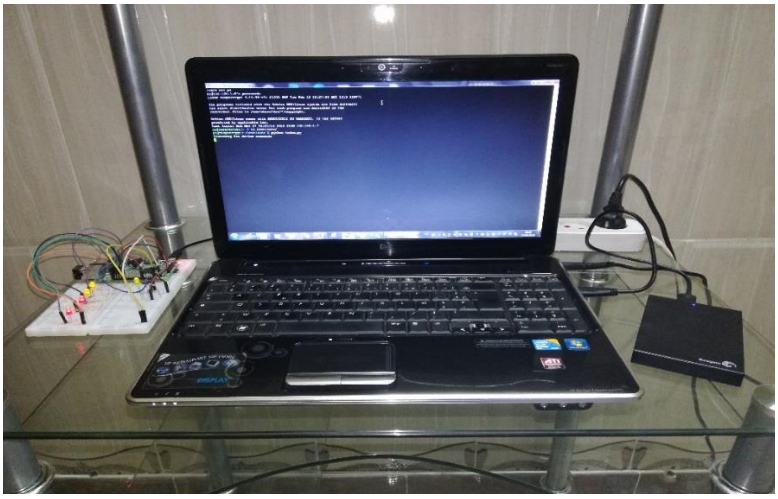


Figure III.7: The PC for SSH connection.

We find this connection type to be very efficient; it allows us full access to the Raspberry Pi. Everything displayed in the terminal window can be duplicated in a monitor connected directly to the Raspberry Pi, if this is the way we choose to run our system, for that we use PuTTY which is a very small program that we do not even have to install. We Download the executable and start it, and we find the configuration screen shown here:

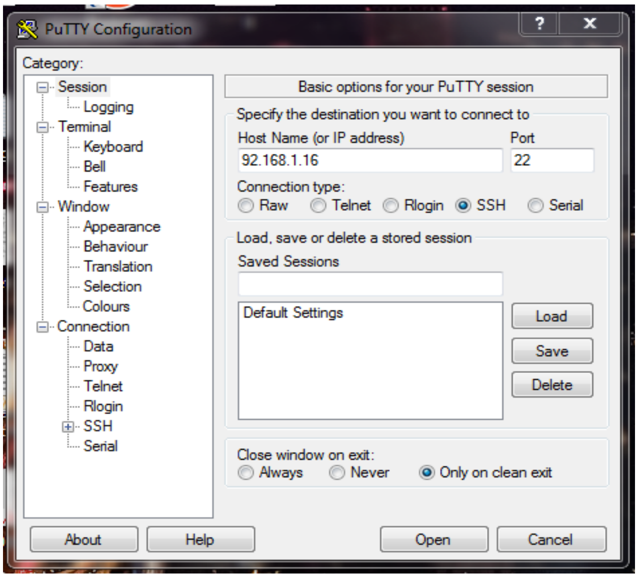


Figure III.8: Connecting to Raspberry Pi with PuTTY.

PuTTY allows us to configure a lot of things, and it also lets us store our configuration for each type of session we need. To log into the Pi, we only have to enter its IP address and click the Open button. Then we'll see the Pi's regular login prompt.

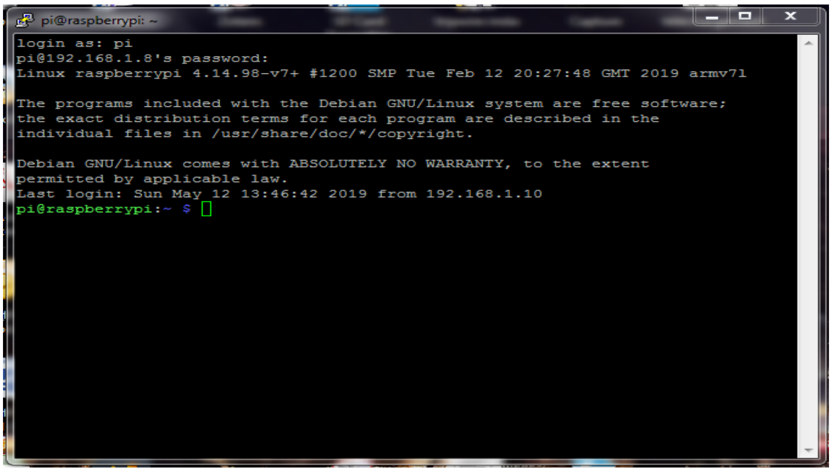


Figure III.9: Accessing the Pi from Windows is easy.

Another way to connect to the Pi over a network connection is to run a Virtual Network Computing Server (VNC server) on the Pi and connect to it using a VNC client. The benefit of this is that we can run a complete working graphical desktop environment in a window on our laptop or desktop. This is a great solution for a portable development environment.

Every time, the Raspberry Pi connects and work with WiFi, it takes a different IP address and it be changed too, so I used an application in my phone, it calls Cayenne (android application), it allows us to know the IP address of my Pi after a few minute of search as in the figure below:

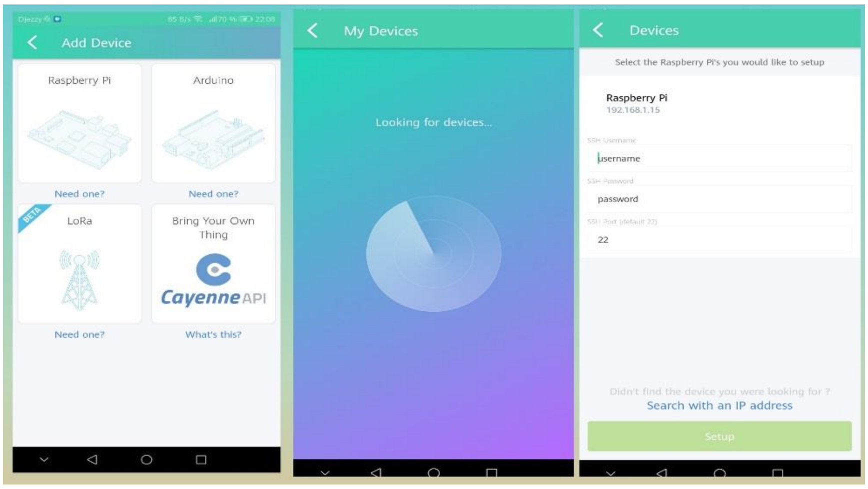


Figure III.10: Phone application Cayenne.

III.4.2 Connect our Raspberry pi to the losant platform

III.4.2.1 Create edge compute Device

In Losant, a device could be Raspberry Pi, Arduino, smart bulb, or any custom hardware.

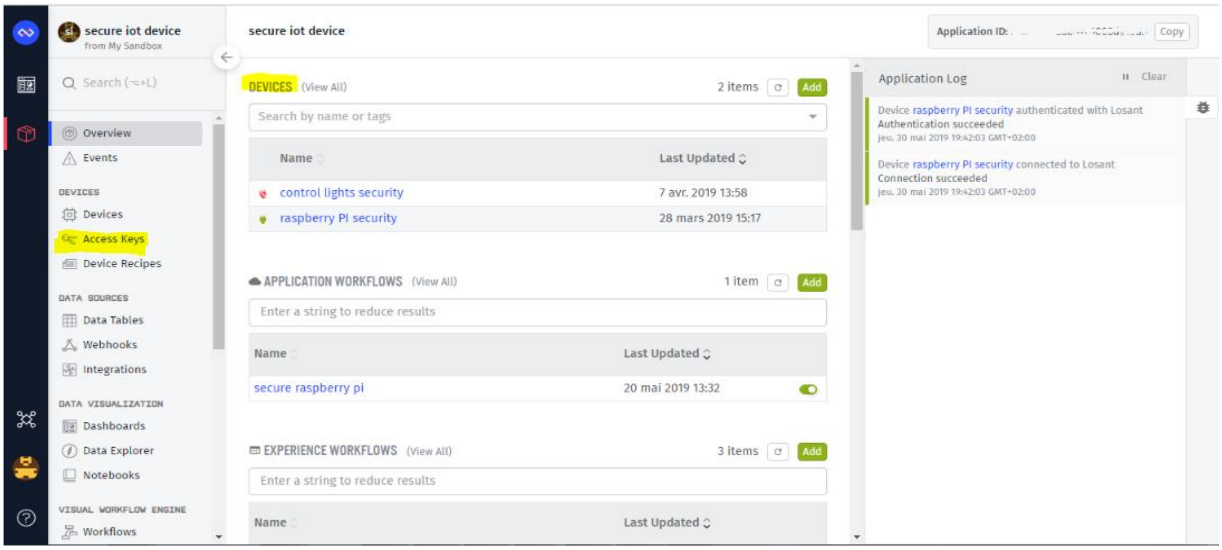


Figure III.11: Create a device in Losant platform.

III.4.2.2 Create access key/secret

To connect our devices to the Losant Platform, we must use a set of security credentials called access keys. Access keys consist of a generated key and secret pair.

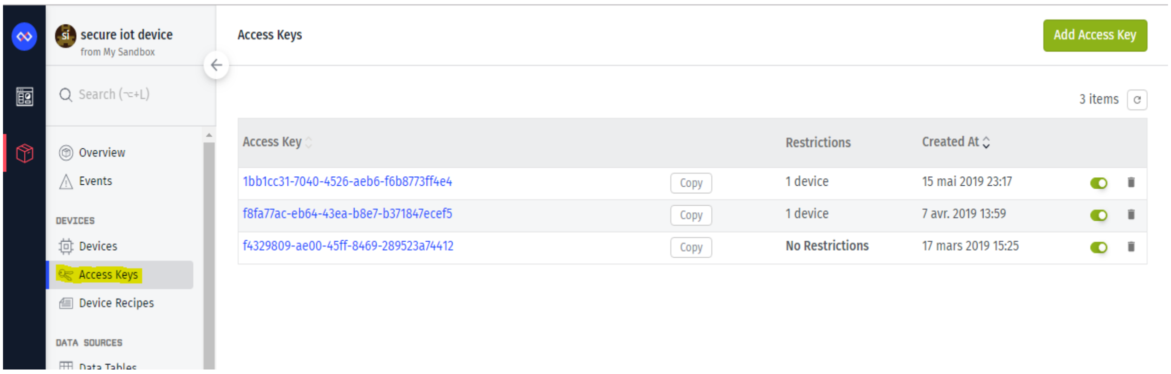


Figure III.12: Create the access keys of the application device.

III.4.2.3 Install Losant edge agent

The Losant Edge Agent ("Agent") is a command line utility exposed through Docker as a container we can run on our Edge Compute device. The below focuses on installing the Agent on a device running Ubuntu, but the Agent can be run on any device that can run Docker. While the expectation is that it will be running on an IoT style device.

First, we will need to make sure our packages are up to date:

```
pi@raspberrypi:~ $ sudo apt-get update
Atteint:1 http://archive.raspberrypi.org/debian stretch InRelease
Réception de:2 http://raspbian.raspberrypi.org/raspbian stretch InRelease [15,0 kB]
Atteint:3 https://download.docker.com/linux/raspbian stretch InRelease
Atteint:4 https://repo.mosquitto.org/debian stretch InRelease
15,0 ko réceptionnés en 10s (1 387 o/s)
Lecture des listes de paquets... Fait
```

Before installing Docker on a new host machine, it is recommended to install the Docker repository. Afterward, we can install and update Docker from the repository.

```
pi@raspberrypi:~ $ sudo apt-get install \
>     apt-transport-https \
>     ca-certificates \
>     curl \
>     software-properties-common
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
apt-transport-https is already the newest version (1.4.9).
ca-certificates is already the newest version (20161130+nmu1+deb9u1).
curl is already the newest version (7.52.1-5+deb9u9).
software-properties-common is already the newest version (0.96.20.2-1).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  realpath vlc-plugin-notify vlc-plugin-samba vlc-plugin-video-splitter vlc-plugin-visualization
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 124 non mis à jour.
```

Then, we add Docker's official GPG key and we verify that we now have it with the fingerprint. Next, we set up the stable repository

- AMD

```
pi@raspberrypi:~ $ sudo add-apt-repository \
>    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
>    $(lsb_release -cs) \
>    stable"
```

- ARM

```
pi@raspberrypi:~ $ echo "deb [arch=armhf] https://download.docker.com/linux/$(. /etc/os-release; echo "$ID") \
>     $(lsb_release -cs) stable" | \
>     sudo tee /etc/apt/sources.list.d/docker.list
```

Finally, we are ready to install the latest version of Docker:

```
pi@raspberrypi:~ $ sudo apt-get install docker-ce
```

We can verify that we have successfully installed Docker by the command below:

```
pi@raspberrypi:~ $ sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Now we have to Getting the Losant Edge Agent. At this point, we have a working Docker Daemon running in the background on our device and are ready to "pull" the **Losant/edge-agent** from Docker Hub. Again, the Agent runs as a Docker Container and therefore comes with everything it needs to run on our system. We don't need to install anything else on our device to run the Agent.

```
pi@raspberrypi:~ $ sudo docker pull losant/edge-agent
Using default tag: latest
```

All what we have to do for using that losant edge agent with our raspberry pi, will be simply with two steps, First we will create storage area to have access to the persistent data created by the agent on our host machine and can reuse it if we have to destroy an Agent container and rebuild (upgrading the agent, for instance).

So, all what we need to do is create a folder locally to house the data. This folder can be anywhere on my file system, so long as permissions are set so "docker" can write to it. For instance, we might choose to put this folder at "~/losant-edge-agent" for easy access. For this example, we're going to create a folder at "/var/lib/losant-edge-agent/"and update the permissions to allow "docker" to write to it.

```
pi@raspberrypi:~ $ sudo mkdir -p /var/lib/losant-edge-agent/data
pi@raspberrypi:~ $ sudo chmod -R a+rwx /var/lib/losant-edge-agent
```

Second, we must be set to rain the container the three required environment variables, we must provide:

The "Device ID" of our edge agent Device, as well as the "Access Key" and "Access secret" that associated with our application.

```
pi@raspberrypi:~ $ docker run -d --restart always --name docs-agent \
  -e 'DEVICE_ID=<5c8e58c3c3bfa3000b183d52>' \
  -e 'ACCESS_KEY=<f43[illegible]-ae00-45ff-8469-[illegible]3a74412>' \
>   -e 'DEVICE_ID=<5c8e58c3c[illegible]183d52>' \
>   -e 'ACCESS_KEY=<f4329[illegible]9-ae00-45ff-8469-289523a74412>' \
>   -e 'ACCESS_SECRET=<f28de6ab3fbbcca851707d7341dbd93e67e326c99939ca62d4b203c4083e6107>' \
>   -v /var/lib/losant-edge-agent/data:/data \
>   losant/edge-agent
```

III.4.3 Test the connection between the pi and losant Platform

First, we setup our Raspberry Pi and we made sure our device is updated and the proper libraries are installed.

Then, we choose two Experiment will connect to Losant by our pi, one of them will be controlled by API services which allows us to read/control GPIO remotely with code. And the other can be controlling by Losant platform directly.

III.4.3.1 Creating an API service

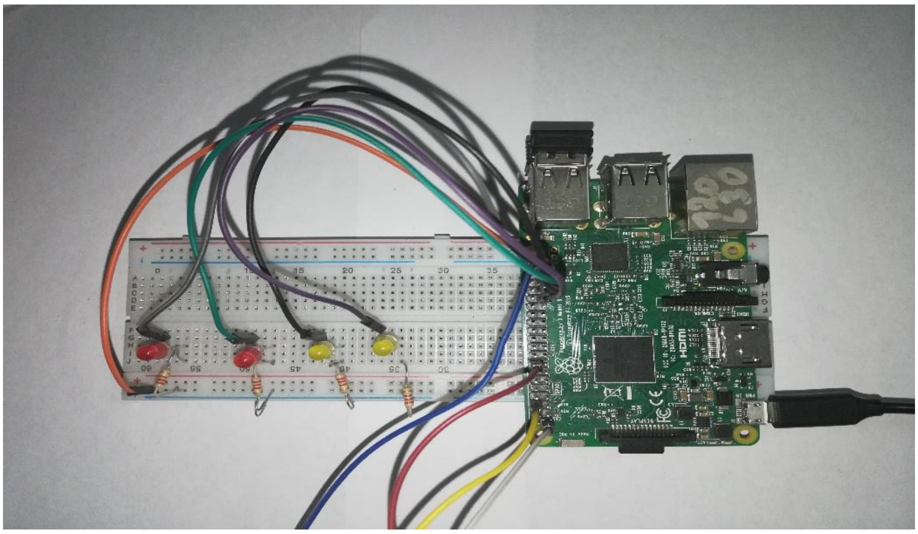


Figure III.13: The components of switch a led using Losant platform.

We will walk through creating an API for the Raspberry Pi to control four LEDs, each connected to its own GPIO. Then we will create an application in Losant, a device and also the Access Key and Secret.

To trigger an API request, we use an application called Postman and all what we have to do is Enter our URL and send the request but first we need to create more a few things in losant platform.

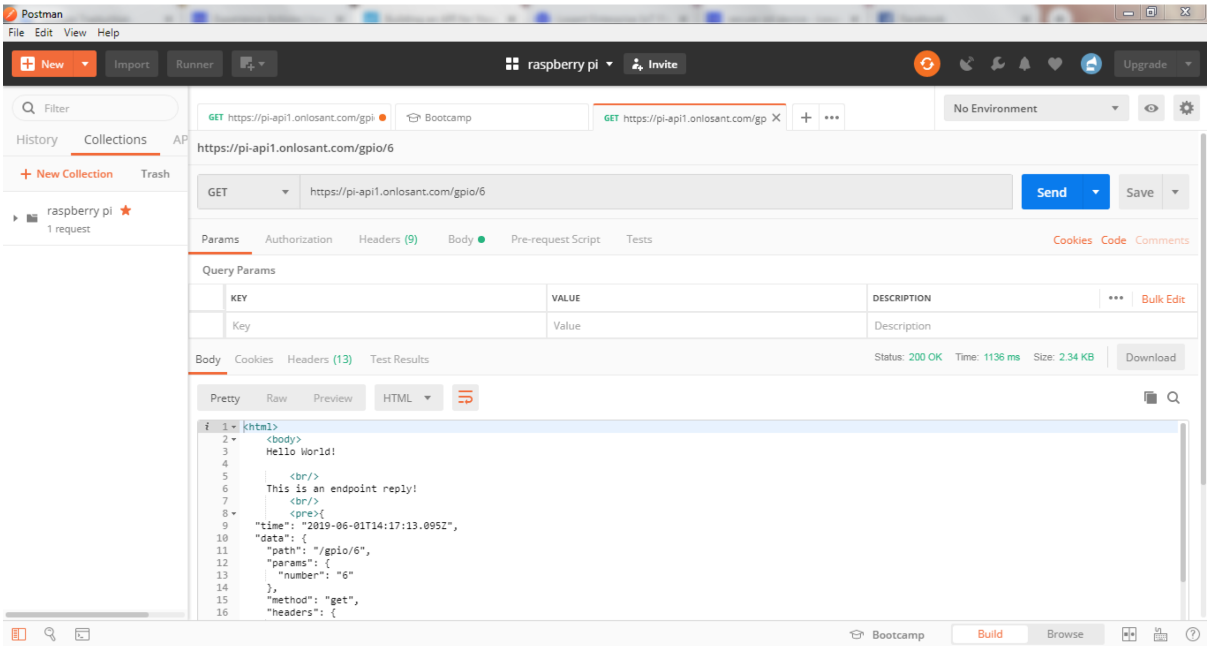


Figure III.14: Postman software as API service.

III.4.3.1.A Create a Losant Experience

An Experience Endpoint is a combination of an HTTP method and a route that, when invoked by an HTTP request, can fire a workflow or directly respond with an Experience Page. Fired workflows can also generate and issue a response to the request.

Writing an API service, implementing all of the user authentication, and hosting the result somewhere is a lot of work. An Experience in Losant brings all of this functionality directly inside our Losant application.

Experiences are the key to building a fully functional API allowing users to interact with our device.

III.4.3.1.B Create an Endpoint

An Endpoint is a combination of an HTTP method and a route that, when invoked by an HTTP request, can fire a workflow. That workflow does some work, like control an LED, and responds to the request.

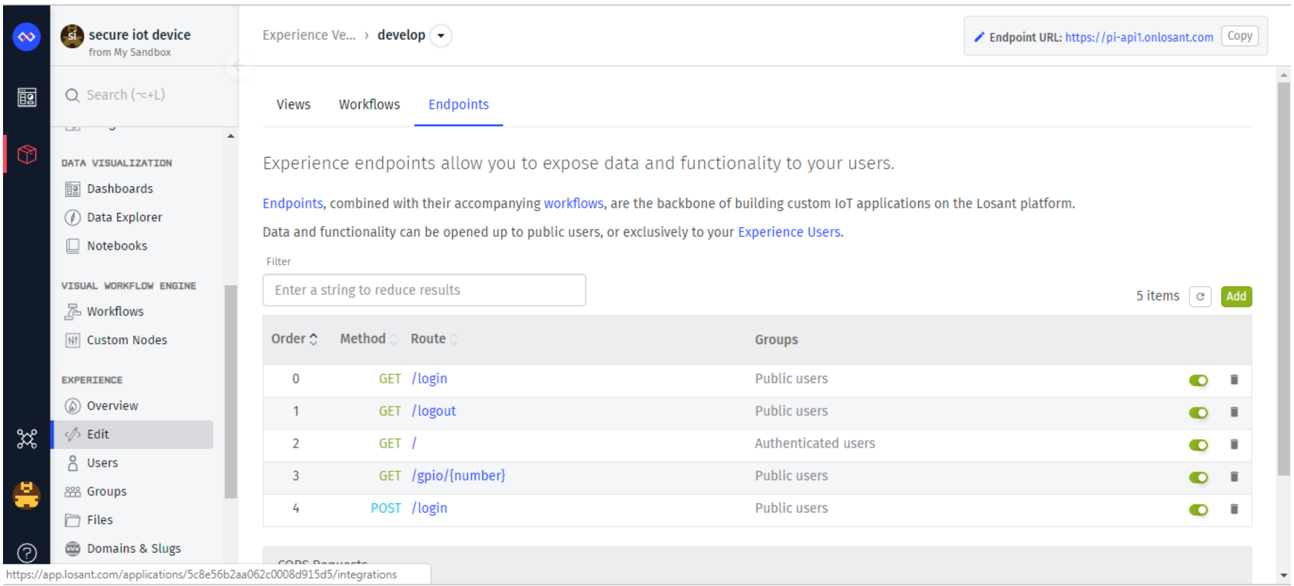


Figure III.15: Create an endpoint experience.

For this example, we're going to create a GET request to toggle a GPIO pin. If the URL for this route would look something like this:

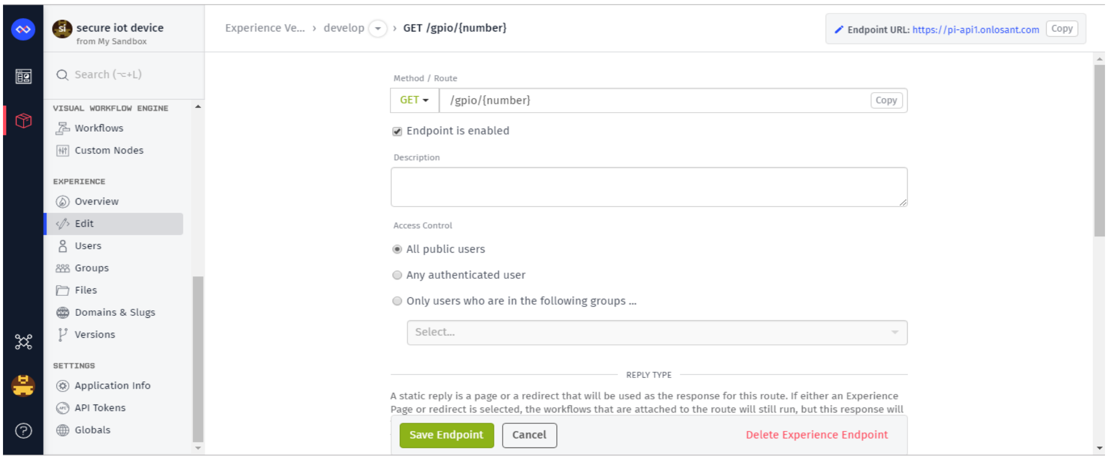


Figure III.16: Create a GET request to toggle a GPIO pin.

"Number" in the route is a path parameter. We will replace this with the GPIO we want to toggle.

Every endpoint has access control. To make things easy, our routes will be public. However, we are able to add authentication to endpoints.

III.4.3.1.C Create a Workflow for the new endpoint

We had a basic workflow:

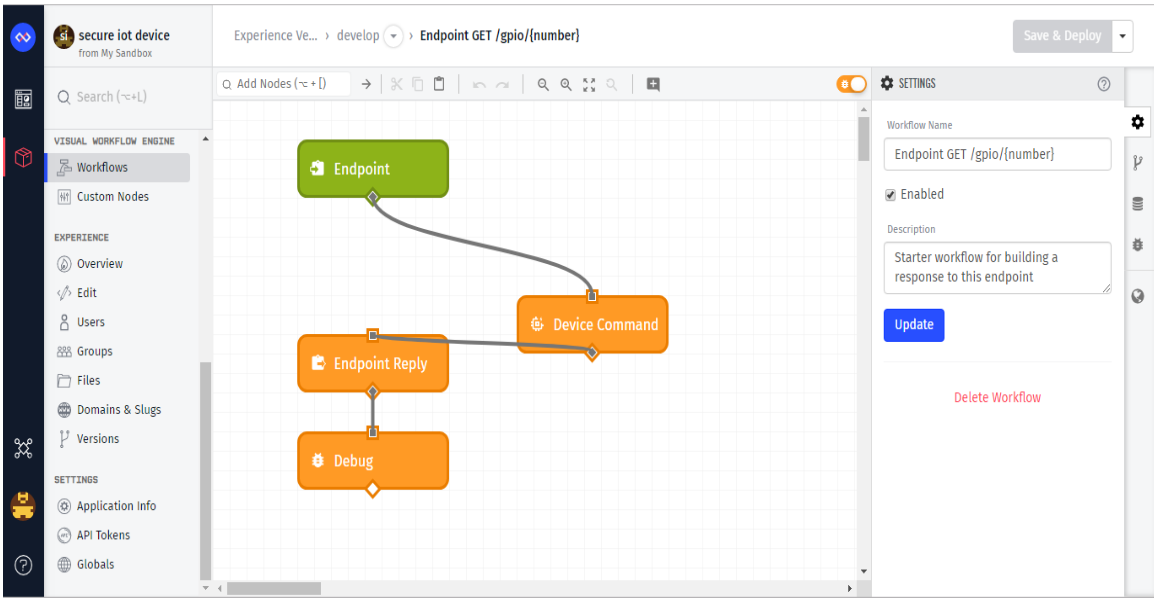


Figure III.17: a workflow for the endpoint GET.

Every endpoint workflow will start with an Endpoint Trigger and end with an Endpoint Reply. The Endpoint Trigger is fired when we make an HTTP request. The Endpoint Reply responds to the HTTP request. So, this workflow is not doing anything but responding immediately.

In our python code, there is an "on command" function that's attached to an event. We can trigger this function by sending a Device Command to our Pi. Drag and Drop the Device Command node.

In the code, we are also listening for a command called "gpioControl" and looking for a payload variable called "gpio".

And we select the Device Command node to configure it.

- Command Name Template: gpioControl
- Command Payload Type: JSON Template
- Command Payload JSON Template

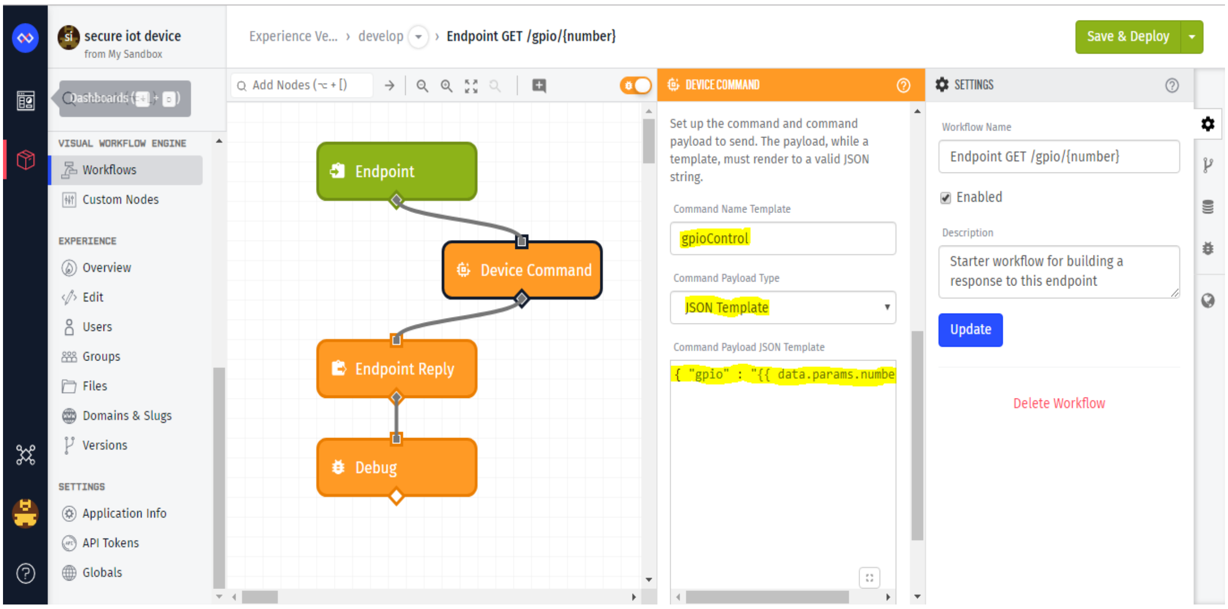


Figure III.18: Device command node functions.

Now we are ready to test, once we fire the request, we use 6, 13, 19, or 26 as the parameter and we watch that LED light up.

III.4.3.2 Controlling a LED by losant platform

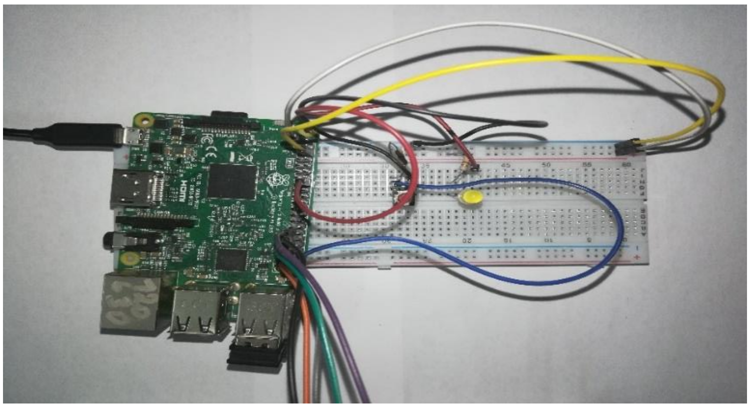


Figure III.19: Controlling a LED connecting to raspberry pi.

III.4.3.2.A Creating a Workflow/Toggle LED

We add a new device for our Raspberry Pi, in this example we will receive a state when the button pressed, Then we create an access key, which we will use to authenticate the device's connection.

Once the states are being sent, Losant workflows allow us to trigger any number of useful actions. The workflow will send an SMS message whenever the button is pressed. But there was a problem with the area of our country in that strp.

Losant also supports commands, which allow you instruct the device to take an action. For this example, the device will watch for a “toggle” command, which will cause it to toggle the LED.

A typical way to send commands is by using a Losant workflow:

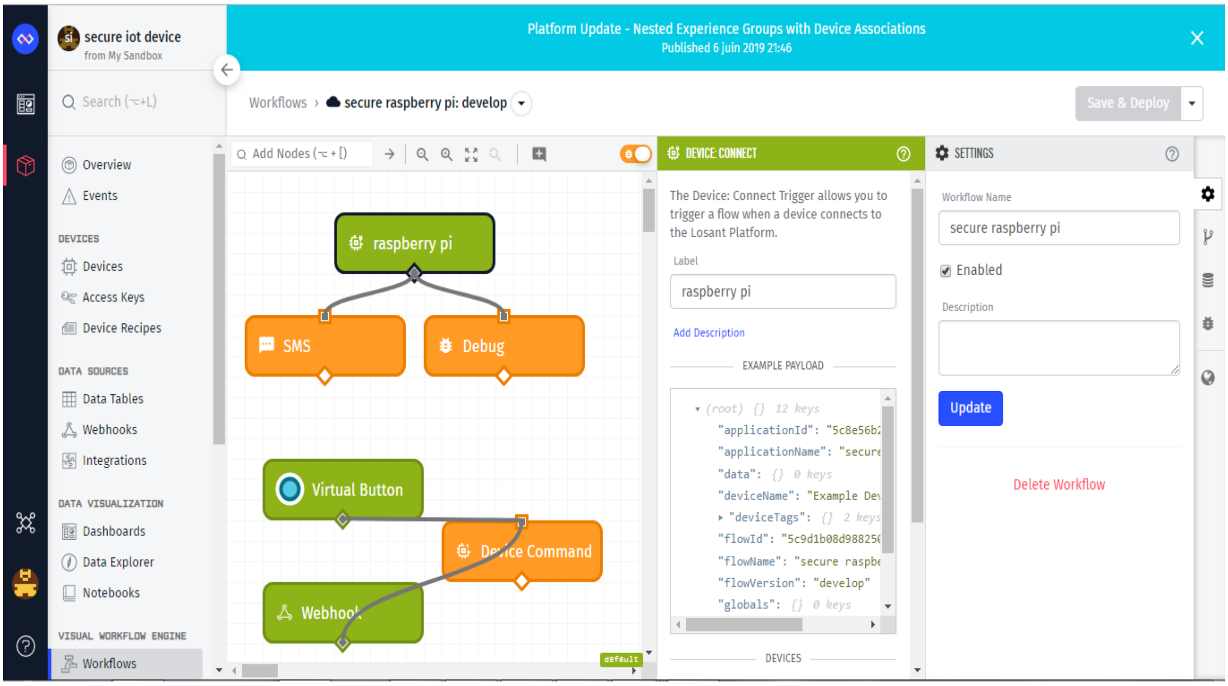


Figure III.20: Losant workflow to toggle the LED.

Whenever the button on the above workflow is clicked, it will send a command with the name “toggle” to the connected Raspberry Pi. As below:

```
pi@raspberrypi: ~/Downloads
pi@raspberrypi:~/Downloads $ python3 index.py
Listening for device commands
Sending Device State
toggle command received.
```

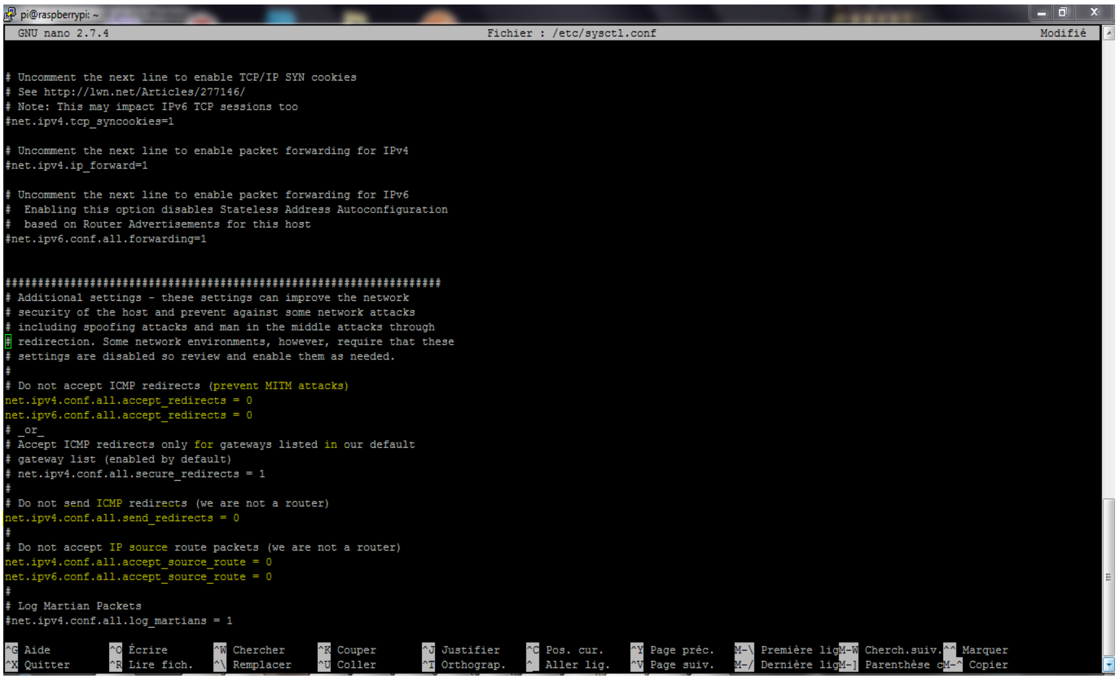
III.4.4 Protection of our network

In order to prevent the many networking attacks, we edit the sysctl file as described below.

```
pi@raspberrypi:~ $ sudo nano /etc/sysctl.conf
```

And we try to modify this file to enable the protection against MITM, Spoofing, and others. The code below illustrates those modifications:

```
GNU nano 2.7.4                    Fichier : /etc/sysctl.conf                    Modifié

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1


###################################################################
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv6.conf.all.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
#net.ipv4.conf.all.send_redirects = 0
#
# Do not accept IP source route packets (we are not a router)
#net.ipv4.conf.all.accept_source_route = 0
#net.ipv6.conf.all.accept_source_route = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1

^G Aide      ^O Écrire     ^W Chercher   ^K Couper  ^J Justifier   ^C Pos. cur.  ^Y Page préc.  M-\ Première lig M-W Cherch.suiv. ^^ Marquer
^X Quitter   ^R Lire fich. ^\ Remplacer  ^U Coller  ^T Orthograp.  ^_ Aller lig. ^V Page suiv.  M-/ Dernière lig M-] Parenthèse   M-^ Copier
```



```
GNU nano 2.7.4                    Fichier : /etc/sysctl.conf                    Modifié

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1


###################################################################
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
net.ipv4.conf.all.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
net.ipv4.conf.all.send_redirects = 0
#
# Do not accept IP source route packets (we are not a router)
net.ipv4.conf.all.accept_source_route = 0
net.ipv6.conf.all.accept_source_route = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1

^G Aide      ^O Écrire     ^W Chercher   ^K Couper  ^J Justifier   ^C Pos. cur.  ^Y Page préc.  M-\ Première lig M-W Cherch.suiv. ^^ Marquer
^X Quitter   ^R Lire fich. ^\ Remplacer  ^U Coller  ^T Orthograp.  ^_ Aller lig. ^V Page suiv.  M-/ Dernière lig M-] Parenthèse   M-^ Copier
```

III.4.5 Setup a Firewall

The Linux kernel in Ubuntu provides a packet filtering system called netfilter, and the traditional interface for manipulating netfilter are the iptables suite of commands. Iptables provide a complete firewall solution that is both highly configurable and highly flexible.

Becoming proficient in iptables takes time, and getting started with netfilter firewalling using only iptables can be a daunting task. As a result, many frontends for iptables have been created over the years, each trying to achieve a different result and targeting a different audience [8].

So we install the Uncomplicated Firewall (UFW) in our Raspberry Pi as described below:

```
pi@raspberrypi:~ $ sudo apt-get install ufw
```

Then we must configure the firewall to our needs. However, deny any incoming connection by default as described below:

```
pi@raspberrypi:~ $ sudo ufw default deny incoming
```

```
pi@raspberrypi:~ $ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
```

We may for example allow SSH access only from our local network. The command below illustrate that:

```
pi@raspberrypi:~ $ sudo ufw allow from 192.168.1.0/24 to any port 22 proto tcp
Rules updated
```

To allow access to our web server using HTTP and HTTPS, we have to open the corresponding ports. Below two configuration, the first we open the port to our local network, the second we open them for internet:

III.4.5.1 Open the ports locally

```
pi@raspberrypi:~ $ sudo ufw allow from 192.168.1.0/24 to any port 80 proto tcp
Rules updated
```

```
pi@raspberrypi:~ $ sudo ufw allow from 192.168.1.0/24 to any port 443 proto tcp
Rules updated
```

III.4.5.2 Open the ports to internet

```
pi@raspberrypi:~ $ sudo ufw allow from any to any port 80 proto tcp
```

```
pi@raspberrypi:~ $ sudo ufw allow from any to any port 80 proto tcp
Skipping adding existing rule
Skipping adding existing rule (v6)
```

```
pi@raspberrypi:~ $ sudo ufw allow from 192.168.1.0/24 to any port 443 proto tcp
Rules updated
```

```
pi@raspberrypi:~ $ sudo ufw allow from any to any port 443 proto tcp
Skipping adding existing rule
Skipping adding existing rule (v6)
```

By default, UFW is disabled so we should enable it by command below:

```
pi@raspberrypi:~ $ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
pi@raspberrypi:~ $
```

If UFW is active, the output will say that it's active, and it will list any rules that are set. For example, if the firewall is set to allow SSH (port 22) connections just from my network, the output might look something like this:

```
pi@raspberrypi:~ $ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       192.168.1.0/24
80/tcp                     ALLOW       192.168.1.0/24
443/tcp                    ALLOW       192.168.1.0/24
Anywhere                   DENY        192.168.43.0/24
80/tcp (v6)                ALLOW       Anywhere (v6)
443/tcp (v6)               ALLOW       Anywhere (v6)
```

III.4.5.3 Delete UFW rule

The easiest way, but perhaps not the most efficient way to remove UFW rules, is to list all rules in numbered format:

```
pi@raspberrypi:~ $ sudo ufw status numbered
Status: active

     To                         Action      From
     --                         ------      ----
[ 1] 22/tcp                     ALLOW IN    192.168.1.0/24
[ 2] 80/tcp                     ALLOW IN    192.168.1.0/24
[ 3] 443/tcp                    ALLOW IN    192.168.1.0/24
[ 4] Anywhere                   DENY IN     192.168.43.0/24
[ 5] 80/tcp (v6)                ALLOW IN    Anywhere (v6)
[ 6] 443/tcp (v6)               ALLOW IN    Anywhere (v6)
```

We note, the line numbers for each rule. To remove rule eg. [1] execute:

```
pi@raspberrypi:~ $ sudo ufw delete  1
```

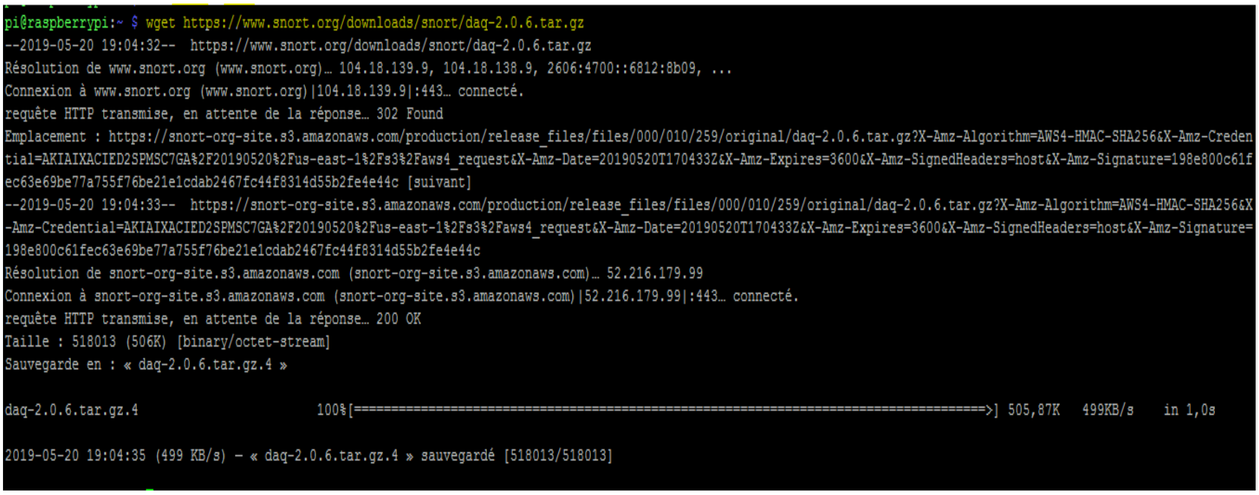
III.4.6 Setup an Intrusion Detection System (IDS)

Firewall blocks the traffic from/to unauthorized ports. However, it may be some malicious traffic coming over the ports authorized by the firewall (22, 80, and 443 in occurrence). IDS is a program that looks deeper into packets payloads allowing it to detect malicious traffic.

Suricata is Complicated in operation and requires more system resources for full-fledged functioning in List of Open Source IDS Tools, so we used snort which is one of the best open source IDS. There is not yet a supported version for Raspbian Jessie. We have to build it from the sources. Note that the build in our R-PI may take more than an hour. Below the commands to build it:

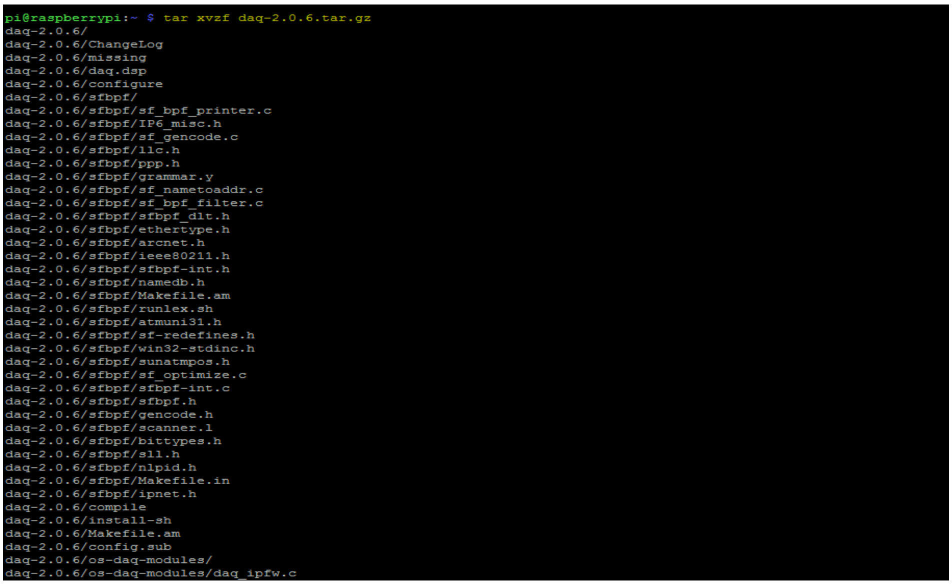
III.4.6.1 Install Daq

Snort itself uses something called Data Acquisition library (DAQ) to make abstract calls to packet capture libraries. Download the latest DAQ source package from the Snort website with the wgetcommand underneath. We can replace the version number in the command if a newer source available.



```
pi@raspberrypi:~ $ wget https://www.snort.org/downloads/snort/daq-2.0.6.tar.gz
--2019-05-20 19:04:32--  https://www.snort.org/downloads/snort/daq-2.0.6.tar.gz
Résolution de www.snort.org (www.snort.org)… 104.18.139.9, 104.18.138.9, 2606:4700::6812:8b09, ...
Connexion à www.snort.org (www.snort.org)|104.18.139.9|:443… connecté.
requête HTTP transmise, en attente de la réponse… 302 Found
Emplacement : https://snort-org-site.s3.amazonaws.com/production/release_files/files/000/010/259/original/daq-2.0.6.tar.gz?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIXACIED2SPMSC7GA%2F20190520%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20190520T170433Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=198e800c61fec63e69be77a755f76be21e1cdab2467fc44f8314d55b2fe4e44c [suivant]
--2019-05-20 19:04:33--  https://snort-org-site.s3.amazonaws.com/production/release_files/files/000/010/259/original/daq-2.0.6.tar.gz?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIXACIED2SPMSC7GA%2F20190520%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20190520T170433Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=198e800c61fec63e69be77a755f76be21e1cdab2467fc44f8314d55b2fe4e44c
Résolution de snort-org-site.s3.amazonaws.com (snort-org-site.s3.amazonaws.com)… 52.216.179.99
Connexion à snort-org-site.s3.amazonaws.com (snort-org-site.s3.amazonaws.com)|52.216.179.99|:443… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 518013 (506K) [binary/octet-stream]
Sauvegarde en : « daq-2.0.6.tar.gz.4 »

daq-2.0.6.tar.gz.4                    100%[==========================================================================>] 505,87K   499KB/s    in 1,0s

2019-05-20 19:04:35 (499 KB/s) — « daq-2.0.6.tar.gz.4 » sauvegardé [518013/518013]
```

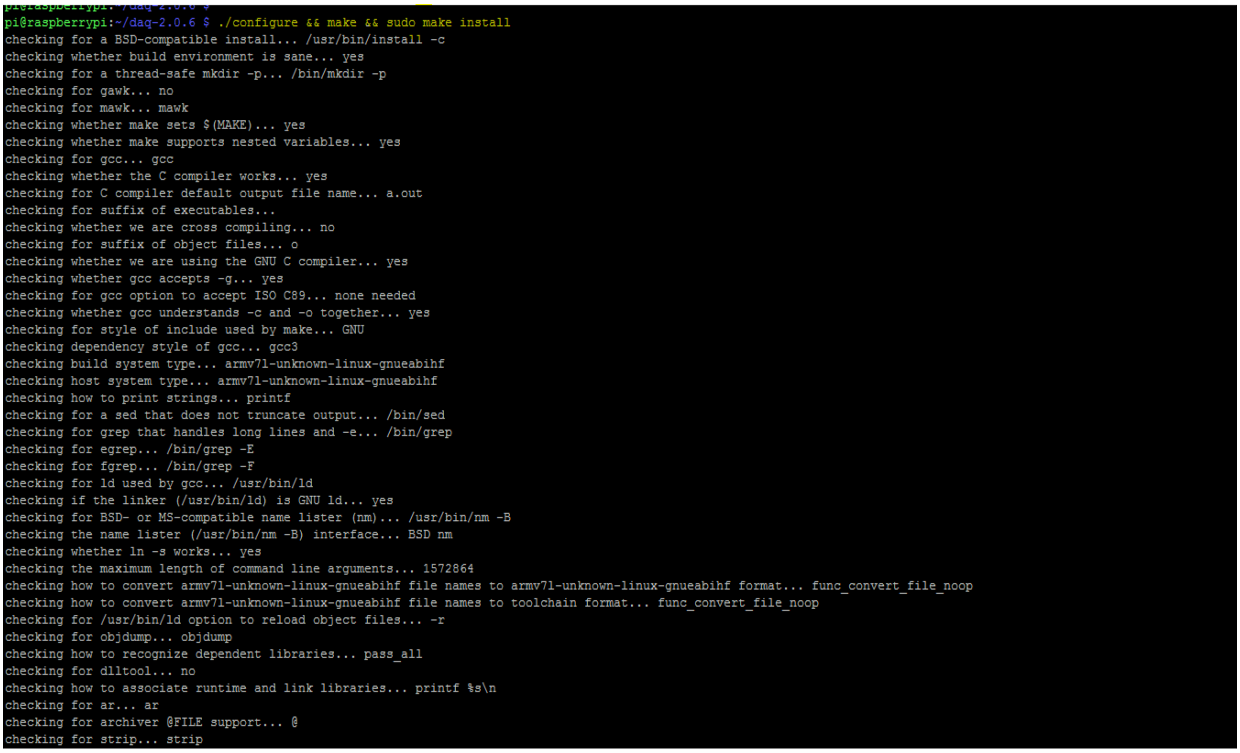
The appropriate package for my operating system and the installation: The download will only take a few seconds. When complete, extract the source code and jump into the new directory with the following commands.



```
pi@raspberrypi:~ $ tar xvzf daq-2.0.6.tar.gz
daq-2.0.6/
daq-2.0.6/ChangeLog
daq-2.0.6/missing
daq-2.0.6/daq.dsp
daq-2.0.6/configure
daq-2.0.6/sfbpf/
daq-2.0.6/sfbpf/sf_bpf_printer.c
daq-2.0.6/sfbpf/IP6_misc.h
daq-2.0.6/sfbpf/sf_gencode.c
daq-2.0.6/sfbpf/llc.h
daq-2.0.6/sfbpf/ppp.h
daq-2.0.6/sfbpf/grammar.y
daq-2.0.6/sfbpf/sf_nametoaddr.c
daq-2.0.6/sfbpf/sf_bpf_filter.c
daq-2.0.6/sfbpf/sfbpf_dlt.h
daq-2.0.6/sfbpf/ethertype.h
daq-2.0.6/sfbpf/arcnet.h
daq-2.0.6/sfbpf/ieee80211.h
daq-2.0.6/sfbpf/sfbpf-int.h
daq-2.0.6/sfbpf/namedb.h
daq-2.0.6/sfbpf/Makefile.am
daq-2.0.6/sfbpf/runlex.sh
daq-2.0.6/sfbpf/atmuni31.h
daq-2.0.6/sfbpf/sf-redefines.h
daq-2.0.6/sfbpf/win32-stdinc.h
daq-2.0.6/sfbpf/sunatmpos.h
daq-2.0.6/sfbpf/sf_optimize.c
daq-2.0.6/sfbpf/sfbpf-int.c
daq-2.0.6/sfbpf/sfbpf.h
daq-2.0.6/sfbpf/gencode.h
daq-2.0.6/sfbpf/scanner.l
daq-2.0.6/sfbpf/bittypes.h
daq-2.0.6/sfbpf/sll.h
daq-2.0.6/sfbpf/nlpid.h
daq-2.0.6/sfbpf/Makefile.in
daq-2.0.6/sfbpf/ipnet.h
daq-2.0.6/compile
daq-2.0.6/install-sh
daq-2.0.6/Makefile.am
daq-2.0.6/config.sub
daq-2.0.6/os-daq-modules/
daq-2.0.6/os-daq-modules/daq_ipfw.c
```

```
pi@raspberrypi:~ $ cd daq-2.0.6
pi@raspberrypi:~/daq-2.0.6 $
```

We run the configuration script using its default values, then we compile the program with make and finally install DAQ.



```
pi@raspberrypi:~/daq-2.0.6 $ ./configure && make && sudo make install
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... no
checking for mawk... mawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
checking build system type... armv7l-unknown-linux-gnueabihf
checking host system type... armv7l-unknown-linux-gnueabihf
checking how to print strings... printf
checking for a sed that does not truncate output... /bin/sed
checking for grep that handles long lines and -e... /bin/grep
checking for egrep... /bin/grep -E
checking for fgrep... /bin/grep -F
checking for ld used by gcc... /usr/bin/ld
checking if the linker (/usr/bin/ld) is GNU ld... yes
checking for BSD- or MS-compatible name lister (nm)... /usr/bin/nm -B
checking the name lister (/usr/bin/nm -B) interface... BSD nm
checking whether ln -s works... yes
checking the maximum length of command line arguments... 1572864
checking how to convert armv7l-unknown-linux-gnueabihf file names to armv7l-unknown-linux-gnueabihf format... func_convert_file_noop
checking how to convert armv7l-unknown-linux-gnueabihf file names to toolchain format... func_convert_file_noop
checking for /usr/bin/ld option to reload object files... -r
checking for objdump... objdump
checking how to recognize dependent libraries... pass_all
checking for dlltool... no
checking how to associate runtime and link libraries... printf %s\n
checking for ar... ar
checking for archiver @FILE support... @
checking for strip... strip
```

./configure

Tells us whether are quite ready to build the application. It will check if we have everything needed to build the application, and, if it sees any critical errors it will inform you.

Make

Builds (compiles) the source code. Compiler compiles the code, but, most of the times, the code cannot stand alone, it requires external libraries (usually provided by ubuntu packages) to be installed. After this step the executable(s) of this specific application you are trying to install will be created.

Sudo make install

Moves all the needed for the application files to the appropriate system directories. This has to be done after "**make**" because the executable of the application have been created and can be moved to the appropriate system directory (e.g. /usr/bin/) for later use.

Libraries are necessary, because they allow a programmer to use code made by other people to achieve certain things. i.e. if I wanted to do some disk formatting in my program, I could use the libs someone already wrote to do the formatting, and I just have to make my program call those libraries. If that person finds an issue in their library, they can fix it, and it will fix it in my program too. This is how open-source software can be written so fast and be so stable.

```
checking for capable lex... insufficient
configure: error: Your operating system's lex is insufficient to compile
        libsfbpf. You should install both bison and flex.
        flex is a lex replacement that has many advantages,
        including being able to compile libsfbpf.  For more
        information, see http://www.gnu.org/software/flex/flex.html .
pi@raspberrypi:~/daq-2.0.6 $
```

So we have to install both bison and flex in our raspberry pi. The command below to intall flex library:

```
pi@raspberrypi:~/daq-2.0.6 $ sudo apt-get install flex
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
flex is already the newest version (2.6.1-1.3).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  realpath vlc-plugin-notify vlc-plugin-samba vlc-plugin-video-splitter vlc-plugin-visualization
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 113 non mis à jour.
```

Then we install bison library by the command in the screen below:

```
pi@raspberrypi:~/daq-2.0.6 $ sudo apt-get install bison
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  realpath vlc-plugin-notify vlc-plugin-samba vlc-plugin-video-splitter vlc-plugin-visualization
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
The following additional packages will be installed:
  libbison-dev
Paquets suggérés :
  bison-doc
Les NOUVEAUX paquets suivants seront installés :
  bison libbison-dev
0 mis à jour, 2 nouvellement installés, 0 à enlever et 113 non mis à jour.
Il est nécessaire de prendre 1 177 ko dans les archives.
Après cette opération, 2 479 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] o
Réception de:1 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian stretch/main armhf libbison-dev armhf 2:3.0.4.dfsg-1 [433 kB]
Réception de:2 http://ftp.igh.cnrs.fr/pub/os/linux/raspbian/raspbian stretch/main armhf bison armhf 2:3.0.4.dfsg-1 [744 kB]
1 177 ko réceptionnés en 32s (36,3 ko/s)
Sélection du paquet libbison-dev:armhf précédemment désélectionné.
(Lecture de la base de données... 143578 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libbison-dev_2%3a3.0.4.dfsg-1_armhf.deb ...
Dépaquetage de libbison-dev:armhf (2:3.0.4.dfsg-1) ...
Sélection du paquet bison précédemment désélectionné.
Préparation du dépaquetage de .../bison_2%3a3.0.4.dfsg-1_armhf.deb ...
Dépaquetage de bison (2:3.0.4.dfsg-1) ...
Paramétrage de libbison-dev:armhf (2:3.0.4.dfsg-1) ...
Traitement des actions différées (« triggers ») pour man-db (2.7.6.1-2) ...
Paramétrage de bison (2:3.0.4.dfsg-1) ...
update-alternatives: utilisation de « /usr/bin/bison.yacc » pour fournir « /usr/bin/yacc » (yacc) en mode automatique
```

After we make sure that flex and bison are well install there was another library missed too which is libpcap-dev.

```
checking for linux/if_packet.h... yes
checking whether TPACKET2_HDRLEN is declared... yes
checking whether PACKET_TX_RING is declared... yes
checking pcap.h usability... no
checking pcap.h presence... no
checking for pcap.h... no
checking for pcap_lib_version in -lpcap... no
checking netinet/in.h usability... yes
checking netinet/in.h presence... yes
checking for netinet/in.h... yes
checking libipq.h usability... no
checking libipq.h presence... no
checking for libipq.h... no
checking for linux/netfilter.h... yes
checking for netinet/in.h... (cached) yes
checking libnetfilter_queue/libnetfilter_queue.h usability... no
checking libnetfilter_queue/libnetfilter_queue.h presence... no
checking for libnetfilter_queue/libnetfilter_queue.h... no
checking for linux/netfilter.h... (cached) yes
checking for pcap.h... (cached) no
checking for pcap_lib_version... checking for pcap_lib_version in -lpcap... (cached) no

    ERROR!  Libpcap library version >= 1.0.0 not found.
    Get it from http://www.tcpdump.org

pi@raspberrypi:~/daq-2.0.6 $ sudo apt-get install libpcap-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  realpath vlc-plugin-notify vlc-plugin-samba vlc-plugin-video-splitter vlc-plugin-visualization
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
The following additional packages will be installed:
  libpcap0.8-dev
Les NOUVEAUX paquets suivants seront installés :
  libpcap-dev libpcap0.8-dev
0 mis à jour, 2 nouvellement installés, 0 à enlever et 113 non mis à jour.
Il est nécessaire de prendre 245 ko dans les archives.
Après cette opération, 625 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] o
Réception de:1 http://raspbian.mirror.garr.it/mirrors/raspbian/raspbian stretch/main armhf libpcap0.8-dev armhf 1.8.1-3 [220 kB]
Réception de:2 http://raspbian.mirror.garr.it/mirrors/raspbian/raspbian stretch/main armhf libpcap-dev all 1.8.1-3 [25,4 kB]
```

III.4.6.2 Install Snort

We are now ready to download snort source code:

```
pi@raspberrypi:~ $ wget https://www.snort.org/downloads/snort/snort-2.9.13.tar.gz
--2019-05-20 17:04:04--  https://www.snort.org/downloads/snort/snort-2.9.13.tar.gz
Résolution de www.snort.org (www.snort.org)… 104.18.139.9, 104.18.138.9, 2606:4700::6812:8a09, ...
Connexion à www.snort.org (www.snort.org)|104.18.139.9|:443… connecté.
requête HTTP transmise, en attente de la réponse… 302 Found
Emplacement : https://snort-org-site.s3.amazonaws.com/production/release_files/files/000/010/269/original/snort-2.9.13.tar.gz?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Cre
dential=AKIAIXACIED2SPMSC7GA%2F20190520%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20190520T150405Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=606e7d90
93adc2afe53dff0fb981097368fd1d92fee6d55d4e7128cf501df266 [suivant]
--2019-05-20 17:04:05--  https://snort-org-site.s3.amazonaws.com/production/release_files/files/000/010/269/original/snort-2.9.13.tar.gz?X-Amz-Algorithm=AWS4-HMAC-SHA25
6&X-Amz-Credential=AKIAIXACIED2SPMSC7GA%2F20190520%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20190520T150405Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signatu
re=606e7d9093adc2afe53dff0fb981097368fd1d92fee6d55d4e7128cf501df266
Résolution de snort-org-site.s3.amazonaws.com (snort-org-site.s3.amazonaws.com)… 52.216.186.67
Connexion à snort-org-site.s3.amazonaws.com (snort-org-site.s3.amazonaws.com)|52.216.186.67|:443… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 6553425 (6,2M) [binary/octet-stream]
Sauvegarde en : « snort-2.9.13.tar.gz.1 »

snort-2.9.13.tar.gz.1            100%[===================================================================================>]   6,25M   241KB/s   in 26s

2019-05-20 17:04:32 (249 KB/s) — « snort-2.9.13.tar.gz.1 » sauvegardé [6553425/6553425]

pi@raspberrypi:~ $
```

```
pi@raspberrypi: ~
pi@raspberrypi:~ $
pi@raspberrypi:~ $ tar xvzf snort-2.9.13.tar.gz
snort-2.9.13/
snort-2.9.13/Makefile.am
snort-2.9.13/configure
snort-2.9.13/configure.in
snort-2.9.13/aclocal.m4
snort-2.9.13/Makefile.in
snort-2.9.13/config.h.in
snort-2.9.13/snort.pc.in
snort-2.9.13/COPYING
snort-2.9.13/ChangeLog
snort-2.9.13/compile
snort-2.9.13/config.guess
snort-2.9.13/config.sub
snort-2.9.13/install-sh
snort-2.9.13/ltmain.sh
snort-2.9.13/missing
snort-2.9.13/snort.8
snort-2.9.13/LICENSE
snort-2.9.13/verstuff.pl
snort-2.9.13/RELEASE.NOTES
snort-2.9.13/src/
snort-2.9.13/src/Makefile.am
snort-2.9.13/src/Makefile.in
snort-2.9.13/src/cdefs.h
snort-2.9.13/src/event.h
snort-2.9.13/src/generators.h
snort-2.9.13/src/sf_protocols.h
snort-2.9.13/src/plugin_enum.h
snort-2.9.13/src/rules.h
snort-2.9.13/src/treenodes.h
snort-2.9.13/src/checksum.h
snort-2.9.13/src/debug.c
snort-2.9.13/src/snort_debug.h
snort-2.9.13/src/decode.c
snort-2.9.13/src/decode.h
snort-2.9.13/src/encode.c
snort-2.9.13/src/encode.h
snort-2.9.13/src/active.c
snort-2.9.13/src/active.h
snort-2.9.13/src/log.c
snort-2.9.13/src/log.h
snort-2.9.13/src/mstring.c
```

```
pi@raspberrypi:~ $ cd snort-2.9.13
pi@raspberrypi:~/snort-2.9.13 $
```

```
pi@raspberrypi: ~/snort-2.9.13
pi@raspberrypi:~ $ cd snort-2.9.13
pi@raspberrypi:~/snort-2.9.13 $ ./configure --enable-sourcefire && make && sudo make install
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... no
checking for mawk... mawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether make supports the include directive... yes (GNU style)
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking dependency style of gcc... gcc3
checking for gcc option to accept ISO C99... none needed
checking for gcc option to accept ISO Standard C... (cached) none needed
checking for gcc... (cached) gcc
checking whether we are using the GNU C compiler... (cached) yes
checking whether gcc accepts -g... (cached) yes
checking for gcc option to accept ISO C89... (cached) none needed
checking whether gcc understands -c and -o together... (cached) yes
checking dependency style of gcc... (cached) gcc3
checking build system type... armv7l-unknown-linux-gnueabihf
checking host system type... armv7l-unknown-linux-gnueabihf
checking how to print strings... printf
checking for a sed that does not truncate output... /bin/sed
checking for grep that handles long lines and -e... /bin/grep
checking for egrep... /bin/grep -E
checking for fgrep... /bin/grep -F
checking for ld used by gcc... /usr/bin/ld
checking if the linker (/usr/bin/ld) is GNU ld... yes
checking for BSD- or MS-compatible name lister (nm)... /usr/bin/nm -B
checking the name lister (/usr/bin/nm -B) interface... BSD nm
checking whether ln -s works... yes
checking the maximum length of command line arguments... 1572864
checking how to convert armv7l-unknown-linux-gnueabihf file names to armv7l-unknown-linux-gnueabihf format... func_convert_file_noop
checking how to convert armv7l-unknown-linux-gnueabihf file names to toolchain format... func_convert_file_noop
```

```
pi@raspberrypi:~/snort-2.9.13 $ sudo apt-get install libdumbnet-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  realpath vlc-plugin-notify vlc-plugin-samba vlc-plugin-video-splitter vlc-plugin-visualization
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
The following additional packages will be installed:
  libdumbnet1
Les NOUVEAUX paquets suivants seront installés :
  libdumbnet-dev libdumbnet1
0 mis à jour, 2 nouvellement installés, 0 à enlever et 113 non mis à jour.
Il est nécessaire de prendre 76,4 ko dans les archives.
Après cette opération, 265 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] o
Réception de:1 http://raspbian.mirror.garr.it/mirrors/raspbian/raspbian stretch/main armhf libdumbnet1 armhf 1.12-7 [22,8 kB]
Réception de:2 http://raspbian.mirror.garr.it/mirrors/raspbian/raspbian stretch/main armhf libdumbnet-dev armhf 1.12-7 [53,6 kB]
76,4 ko réceptionnés en 2s (35,9 ko/s)
Sélection du paquet libdumbnet1:armhf précédemment désélectionné.
(Lecture de la base de données... 143812 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libdumbnet1_1.12-7_armhf.deb ...
Dépaquetage de libdumbnet1:armhf (1.12-7) ...
Sélection du paquet libdumbnet-dev précédemment désélectionné.
Préparation du dépaquetage de .../libdumbnet-dev_1.12-7_armhf.deb ...
Dépaquetage de libdumbnet-dev (1.12-7) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.24-11+deb9u4) ...
Paramétrage de libdumbnet1:armhf (1.12-7) ...
Traitement des actions différées (« triggers ») pour man-db (2.7.6.1-2) ...
Paramétrage de libdumbnet-dev (1.12-7) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.24-11+deb9u4) ...
pi@raspberrypi:~/snort-2.9.13 $
```

```
pi@raspberrypi:~/snort-2.9.13 $ sudo apt-get install luajit
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
luajit is already the newest version (2.0.4+dfsg-1).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  realpath vlc-plugin-notify vlc-plugin-samba vlc-plugin-video-splitter vlc-plugin-visualization
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 113 non mis à jour.
```

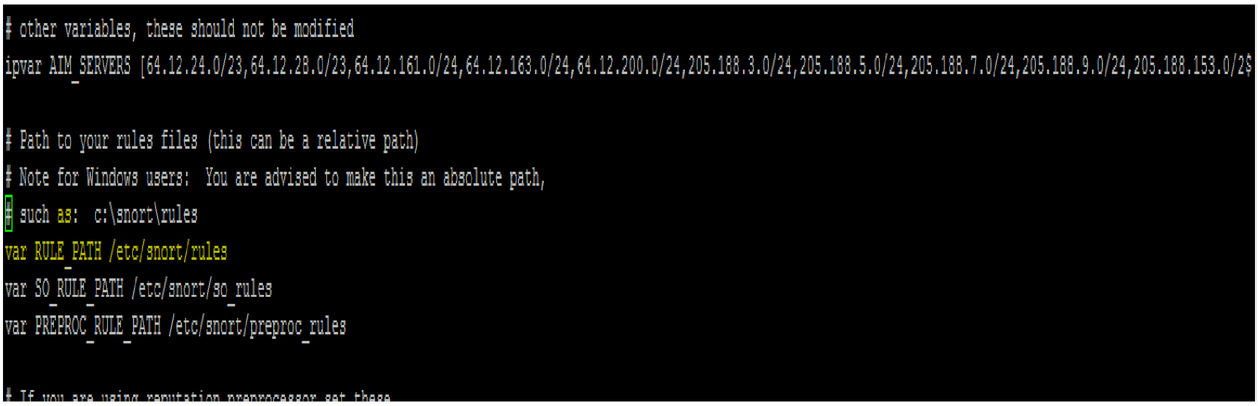
In order to configure Snort, we edit the configuration file as described below:

```
pi@raspberrypi:~/snort-2.9.13 $
pi@raspberrypi:~/snort-2.9.13 $ sudo nano /etc/snort/snort.conf
```

We configure our internal/external networks and the rules file path as described below:

```
pi@raspberrypi: ~/snort-2.9.13
  GNU nano 2.7.4                    Fichier : /etc/snort/snort.conf                    Modifié

# Step #1: Set the network variables.  For more information, see README.variables
###################################################

# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overriden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET any

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

^G Aide       ^O Écrire      ^W Chercher   ^K Couper     ^J Justifier   ^C Pos. cur.   ^Y Page préc.  M-\ Première lig M-W Cherch.suiv. ^^ Marquer
^X Quitter    ^R Lire fich.  ^\ Remplacer  ^U Coller     ^T Orthograp.  ^_ Aller lig.  ^V Page suiv.  M-/ Dernière lig M-] Parenthèse c M-^ Copier
```

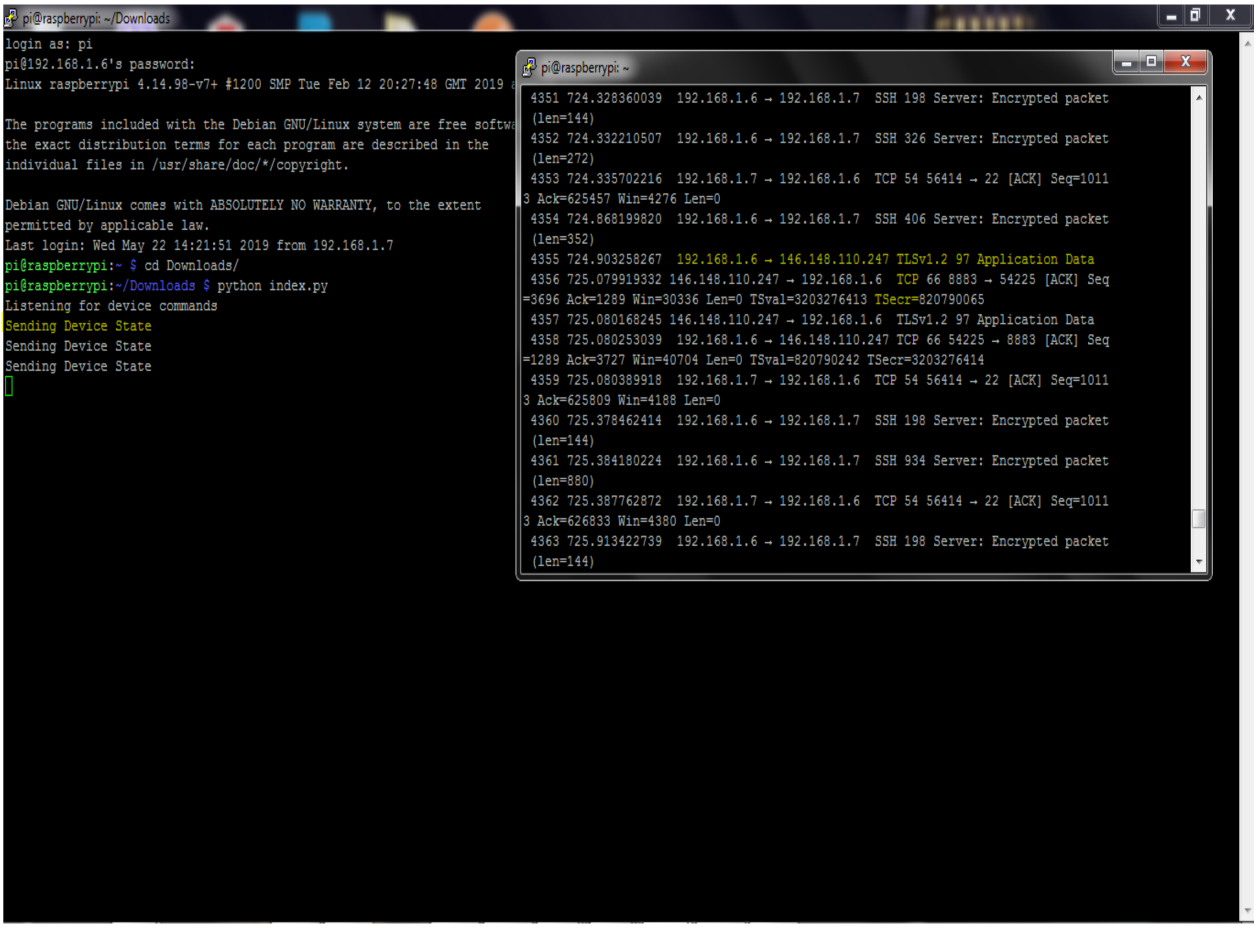
```
# IP addresses:
ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
```



III.4.7 Check the secure transport layer

To make sure that our communication will be secure we install Tshark which is the command line version of Wireshark that has been popular for several years, in our Raspberry Pi to check the packets from our raspberry pi to Losant platform connection.

We started the capture process by using **sudo Tshark** command and the analysis will started in both of the two experiments as we see in the figures below:



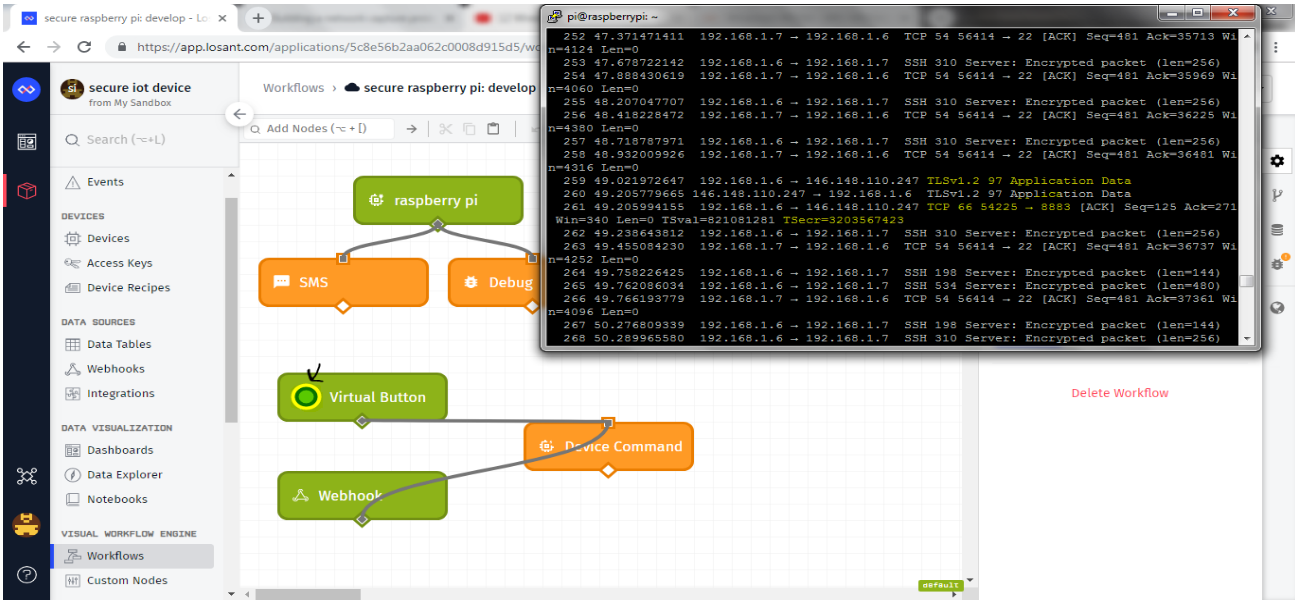


Figure III.21: Analysis packet tshark after a click on the virtual button.

When we click on the virtual button there, we find that the TLS protocol work on as we see in the figure III.21 because when we send the command it use the TLS as transport layer protocol to connect our Raspberry Pi, simply that analysis proves for us that our information is secure from our Losant platform.

The same thing going on with our second experiment which show us the same transport layer protocol TLS as we can realize in figure III.22 below.

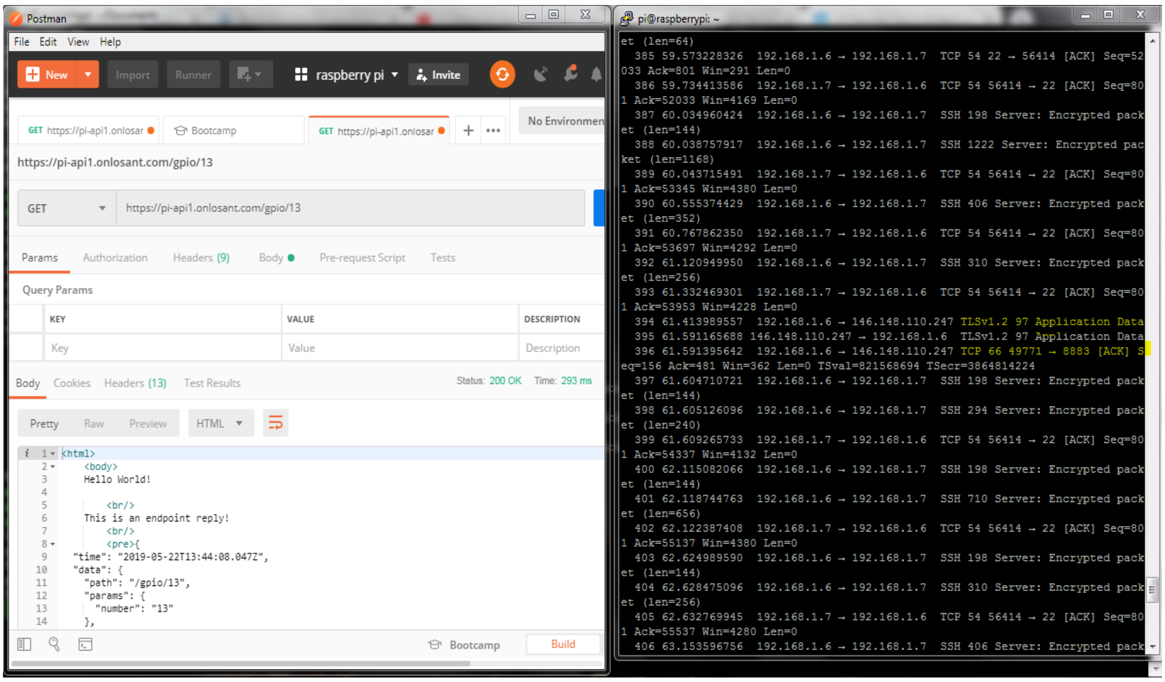


Figure III.22: Tshark analysis of the paquet between API and the raspberry pi.

SLL protocol is already exist in the py files and it work with it too as we find in the figure III.23 with all details.

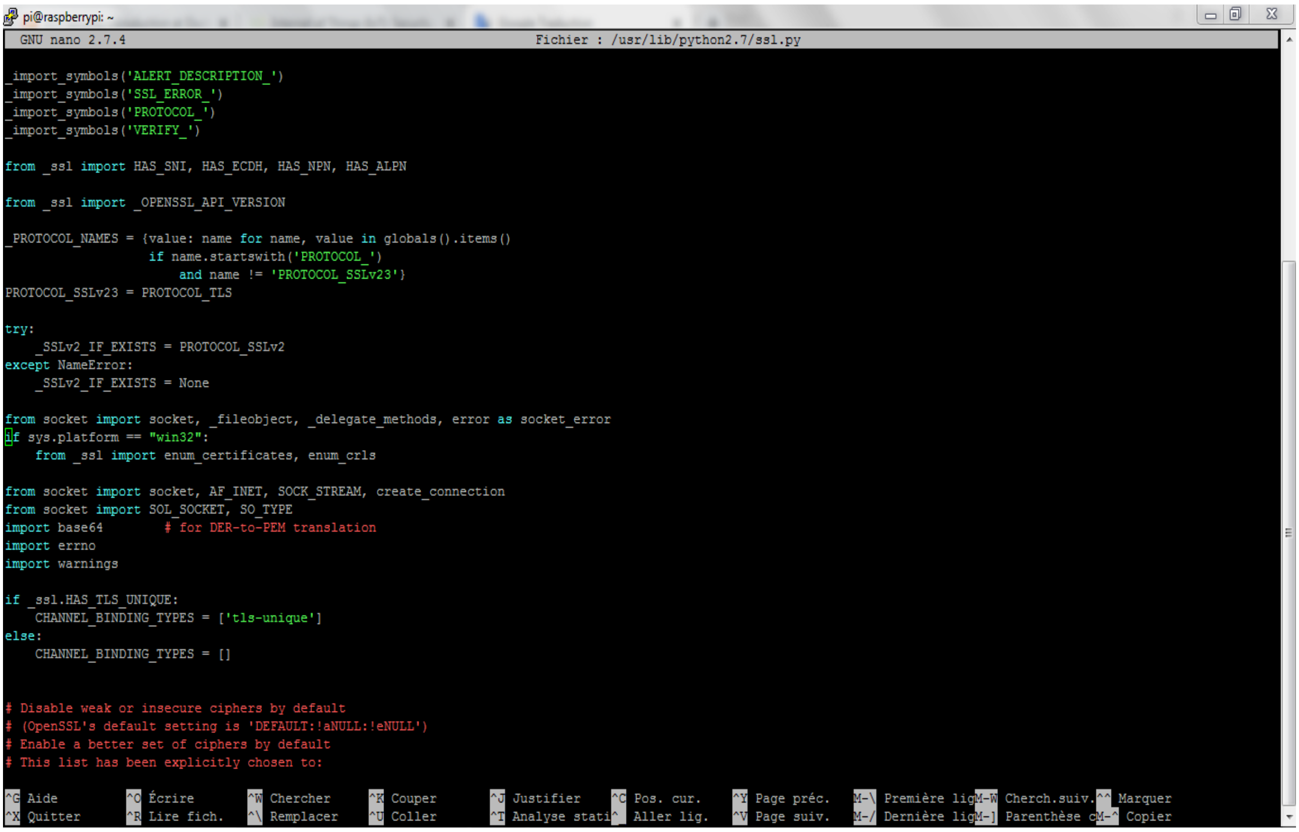


Figure III.23: Screen of SSL.py file in the Rasbperry Pi.

III.5 Conclusion

Security in complicated a painful. However, it protects us and our devices against threats. The main solutions presented in this chapter are installed once per device (ex; IDS and firewall). And we find that firewall allows us to control the port connection, Firewalls applications describe what ports and protocols they need open. It allowed us to block access to those applications if we want.

One of the main advantages of using Tshark here is that we could use the same capture filters as the GUI version aka Wireshark.

Finally, as NIST define us how to secure IoT device and it worked very good with Raspberry Pi.

General Conclusion

General Conclusion

Security research in IoT did an evaluation and found that 70% of what we called IoT devices are venerable of attack. So that motivate us to find more information and look for options that can solve the problem.

The aim of this project was to secure a Raspberry Pi on three steps, first step we started by protect our network in order to prevent the many networking attacks. Second we Setup a Firewall In order to protect our R-PI against threats originating from the internet, a host based firewall is the solution to control any incoming connections. The idea is to close all the ports by default and only open up the ports we need. There also Setup an Intrusion Detection System (IDS), it looks deeper into packets payloads allowing it to detect malicious traffic. Last part we secure our Communications by Transport Layer Security (TLS) formally Secure Sockets Layer (SSL) session which is used to enable privacy and security for the communications over internet. We note that all the IoT platforms provides more or less similar features.

The most important points that we must know it:

We use the raspberry pi for apply a security steps and we used a SD carte of 32 GB , because when we use less GB number we needed to start the work again on another SD carte had more memory size.

There is a new IoT platform which work always to develop all functions of their platform, so we must keep track of all updates or we will find a problem and take a lot of time to solve it.

For IoT device the security part is a thing which we can't ever forget about it or give it less important from the other parts.

There are a lot of software which are open Source IDS Tools that we can replaced with snort in our example here and find the different between them.

We can add security of sensors that enable the Internet of Things (IoT) by collecting the data for smarter decisions. And we advise to add more about energy and the force of the small devices in IoT.

I wish that the students coming after us can gain time and experience in field like IoT device security and add more steps or study a different sides that effect on security like:

IoT device connectivity where devices identify and authenticate each other directly and exchange information without the involvement of a broker.

Compatibility and longevity, this will cause difficulties and require the deployment of extra hardware and software when connecting devices.

Finally, Making stuffs is cool, making secured stuffs is better.

Bibliography

Bibliography

Chapter 1

[1] 20/08/2018, "Internet of Things (IoT) History," Postscapes. [Online]. Available: https://www.postscapes.com/internet-of-things-history/.

[2] "Why it is called Internet of Things: Definition, history, disambiguation".

[3] D. Serpanos and M. Wolf, Internet-of-Things (IoT) Systems. Cham: Springer International Publishing, 2018.

[4] S. Babar, A. Stango, N. Prasad, J. Sen, and R. Prasad, "Proposed embedded security framework for Internet of Things (IoT)," in 2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), Chennai, India, 2011, pp. 1–5.

[5] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of things (IoT) security: Current status, challenges and prospective measures," in 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST), London, United Kingdom, 2015, pp. 336–341.

[6] D. Bandyopadhyay and J. Sen, "Internet of Things: Applications and Challenges in Technology and Standardization," arXiv:1105.1693 [cs], May 2011.

[7] M. Abomhara and G. M. Koien, "Security and privacy in the Internet of Things: Current status and open issues," in 2014 International Conference on Privacy and Security in Mobile Systems (PRISMS), Aalborg, Denmark, 2014, pp. 1–8.

[8] "Internet of thing application at DuckDuckGo." [Online]. Available: https://duckduckgo.com/?q=internet+of+thing+application&t=raspberrypi&iax=images&ia=images.

[9] H. Ning and S. Hu, "Technology classification, industry, and education for Future Internet of Things: TECHNOLOGY CLASSIFICATION, INDUSTRY AND EDUCATION FOR FUTURE IOT," International Journal of Communication Systems, vol. 25, no. 9, pp. 1230–1241, Sep. 2012.

[10] Machfu, "5 Ways IIoT Will Revolutionize the Oil and Gas Industry," IoT For All, 30-Jul-2018.

[11] "Industrial IoT applications in oil and gas industry at DuckDuckGo." [Online]. Available: https://duckduckgo.com/?q=industrial+iot+applications+in+oil+and+gas+industry&t=raspberrypi&iax=images&ia=images.

[12] M. Hassanalieragh et al., "Health Monitoring and Management Using Internet-of-Things (IoT) Sensing with Cloud-Based Processing: Opportunities and Challenges," in 2015 IEEE International Conference on Services Computing, New York City, NY, USA, 2015, pp. 285–292.

[13] D. Team, DataFlair, 04-Jun-2018, "IoT Applications | Top 10 Uses of Internet of Things," [Online]. Available: https://data-flair.training/blogs/iot-applications/.

[14] M. Bielas, "What is a Smart City?" IoT For All, 02-Oct-2018.

[15] A. Gaur, B. Scotney, G. Parr, and S. McClean, "Smart City Architecture and its Applications Based on IoT," Procedia Computer Science, vol. 52, pp. 1089–1094, 2015.

[16] "Smart cities infrastructure iot wide at DuckDuckGo." [Online]. Available: https://duckduckgo.com/?q=smart+cities+infrastructure+iot+wide&t=raspberrypi&iax=images&ia=images.

[17] "What is IoT (Internet of Things)? | IoT Architecture & Application," Edureka, 21-May-2018. .

[18] X. Jia, Q. Feng, T. Fan, and Q. Lei, "RFID technology and its applications in Internet of Things (IoT)," in 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), Yichang, China, 2012, pp. 1282–1285.

[19] S. A. Kumar, T. Vealey, and H. Srivastava, "Security in Internet of Things: Challenges, Solutions and Future Directions," in 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa, HI, USA, 2016, pp. 5772–5781.

[20] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges," in 2012 10th International Conference on Frontiers of Information Technology, Islamabad, Pakistan, 2012, pp. 257–260.

[21] HIMANSHU, "Application Layer Protocols for IOT : IOT Part 11," 23-Oct-2018. [Online]. Available: https://www.engineersgarage.com/Articles/IoT-Application-Layer-Protocols.

[22] "Beyond MQTT: A Cisco View on IoT Protocols." [Online]. Available: https://blogs.cisco.com/digital/beyond-mqtt-a-cisco-view-on-iot-protocols].

[23] Y. Chen and T. Kunz, "Performance evaluation of IoT protocols under a constrained wireless access network," in 2016 International Conference on Selected Topics in Mobile & Wireless Networking (MoWNeT), Cairo, Egypt, 2016, pp. 1–7.

[24] "MQTT: Get started with IoT protocols - Open Source For You." [Online]. Available: https://opensourceforu.com/2016/11/mqtt-get-started-iot-protocols/.

[25] M. Zouai, O. Kazar, G. O. Bellot, B. Haba, N. Kabachi, and M. Krishnamurhty, "Ambiance Intelligence Approach Using IoT and Multi-Agent System:," International Journal of Distributed Systems and Technologies, vol. 10, no. 1, pp. 37–55, Jan. 2019.

[26] F. Azzola, “MQTT Protocol Tutorial: Step by step guide, Mosquitto and MQTT Security,” SwA, 25-Oct-2016. .

[27] “The schema describes the MQTT architecture at DuckDuckGo.” [Online]. Available: https://duckduckgo.com/?q=The+schema+describes+the+MQTT+architecture&t=raspberrypi&iax=images&ia=images.

[28] “Heterogeneous network,” Wikipedia. 12-Jan-2017.

[29] X. Zhao, Z. Xiao, A. Markham, N. Trigoni, and Y. Ren, “Does BTLE measure up against WiFi? A comparison of indoor location performance,” p. 6, 2014.

[30] R. Ijaz, M. A. Pasha, N. U. Hassan, and C. Yuen, “A novel fusion methodology for indoor positioning in IoT-based mobile applications,” in 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, 2018, pp. 742–747.

[31] “11 Internet of Things (IoT) Protocols You Need to Know About.” [Online]. Available: https://www.rs-online.com/designspark/eleven-internet-of-things-iot-protocols-you-need-to-know-about.

[32] X.-Y. Chen and Z.-G. Jin, “Research on Key Technology and Applications for Internet of Things,” Physics Procedia, vol. 33, pp. 561–566, 2012.

[33] “A Bluetooth & ZigBee Comparison For IoT Applications.” [Online]. Available: https://www.link-labs.com/blog/bluetooth-zigbee-comparison. [Accessed: 12-Apr-2019].

[34] J. Porcello, “Industry Perspectives,” p. 2.

[35] M. Teran, J. Aranda, H. Carrillo, D. Mendez, and C. Parra, “IoT-based system for indoor location using bluetooth low energy,” in 2017 IEEE Colombian Conference on Communications and Computing (COLCOM), Cartagena, Colombia, 2017, pp. 1–6.

[36] J.-H. Huh and K. Seo, “An Indoor Location-Based Control System Using Bluetooth Beacons for IoT Systems,” Sensors, vol. 17, no. 12, p. 2917, Dec. 2017.

[37] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, “Internet of Things (IoT) communication protocols: Review,” in 2017 8th International Conference on Information Technology (ICIT), Amman, Jordan, 2017, pp. 685–690.

[38] “IoT - Internet of Things | Oracle.” [Online]. Available: https://www.oracle.com/solutions/internet-of-things/.

[39] Z. Bozdogan and R. Kara, “Layered model architecture for internet of things,” vol. 4, p. 5, 2015.

[40] “The Internet of Things - RFID is a key technology.,” CoreRFID. .

[41] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, “Secure integration of IoT and Cloud Computing,” Future Generation Computer Systems, vol. 78, pp. 964–975, Jan. 2018.

[42] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” Future Generation Computer Systems, vol. 29, no. 7, pp. 1645–1660, Sep. 2013.

[43] A. A. Abbasi and M. Younis, “A survey on clustering algorithms for wireless sensor networks,” Computer Communications, vol. 30, no. 14–15, pp. 2826–2841, Oct. 2007.

[44] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, “From today’s INTRAnet of things to a future INTERnet of things: a wireless- and mobility-related view,” IEEE Wireless Communications, vol. 17, no. 6, pp. 44–51, Dec. 2010.

[45] P. Bellavista, G. Cardone, A. Corradi, and L. Foschini, “Convergence of MANET and WSN in IoT Urban Scenarios,” IEEE Sensors Journal, vol. 13, no. 10, pp. 3558–3567, Oct. 2013.

[46] I. Lee and K. Lee, “The Internet of Things (IoT): Applications, investments, and challenges for enterprises,” Business Horizons, vol. 58, no. 4, pp. 431–440, Jul. 2015.

Chapter 2

[1] Julie Bort, 16 Jan 2014, 18:36, "For the First Time, Hackers Have Used A Refrigerator To Attack Businesses." [Online]. Available: https://www.businessinsider.fr/us/hackers-use-a-refridgerator-to-attack-businesses-2014-1.

[2] V. Sivaraman, H. H. Gharakheili, A. Vishwanath, R. Boreli, and O. Mehani, "Network-level security and privacy control for smart-home IoT devices," in 2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Abu Dhabi, United Arab Emirates, 2015, pp. 163–167.

[3] 19 Dec 2016 - 02:00PM, "IoT attacks: 10 things you need to know | WeLiveSecurity." [Online]. Available: https://www.welivesecurity.com/2016/12/19/iot-attacks-10-things-you-need-to-know/.

[4] Naveen Joshi, 25/01/2019, "3 Steps to Follow to Ensure IoT Security." [Online]. Available: https://www.bbntimes.com/en/technology/3-steps-to-follow-to-ensure-iot-security.

[5] C. Bell, "MySQL for the Internet of Things." Berkeley, CA: Apress, 2016.

[6] Devin Pickell, 22 october2018, "IoT Security Challenges to Expect by 2020." [Online]. Available: https://learn.g2.com/iot-security.

[7] Giancarlo Fortino and al, "Interoperability, Safety and Security in IoT," Third International Conference, INTERIOT 2017, and Fourth International Conference, SASEIOT 2017, Valencia, Spain, 6-7 November 2017, Proceedings. New York, NY: Springer Berlin Heidelberg, 2018.

[8] A. Harper, "10 Biggest security challenges for IoT," Peerbits, 21-Sep-2018.

[9] A. Henderson, "The CIA Triad: Confidentiality, Integrity, Availability," Panmore Institute, 05-Jul-2015.

[10] D. Li, C. Yu, Q. Zhou, and J. Yu, "Using SVM to Detect DDoS Attack in SDN Network," IOP Conference Series: Materials Science and Engineering, vol. 466, p. 012003, Dec. 2018.

[11] "What is an Attack? - Definition from Techopedia." [Online]. Available: https://www.techopedia.com/definition/6060/attack.

[12] H. R. in S. on August 9, 2017, and 11:35 Am Pst, "Report: IoT attacks exploded by 280% in the first half of 2017," TechRepublic. [Online]. Available:

https://www.techrepublic.com/article/report-iot-attacks-exploded-by-280-in-the-first-half-of-2017/.

[13] "Botnet at DuckDuckGo." [Online]. Available:

https://duckduckgo.com/?q=botnet&t=raspberrypi&iar=images&iax=images&ia=images&iai=http%3A%2F%2Fblog.emsisoft.com%2Fwpcontent%2Fuploads%2F2017%2F05%2Fbotnets_infographic-1.png.

[14] Paul Duffy, 30 April2013, "Beyond MQTT: A Cisco View on IoT Protocols." [Online]. Available: https://blogs.cisco.com/digital/beyond-mqtt-a-cisco-view-on-iot-protocols].

[15] B. Prema Sindhuri and M. Kameswara Rao, "IoT security through web application firewall," International Journal of Engineering & Technology, vol. 7, no. 2.7, p. 58, Mar. 2018.

[16] "Man-in-the-Middle-Attacks at DuckDuckGo." [Online]. Available:

https://duckduckgo.com/?q=Man-in-the-Middle-Attacks&t=raspberrypi&iax=images&ia=images&iai=http%3A%2F%2F3.bp.blogspot.com%2F_GbbGkXcpg4%2FSOxQp_oN21I%2FAAAAAAAAAIc%2F3RoQgZay9t0%2Fs640%2Fmitm.png.

[17] D. K. Bhattacharyya, "DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance", 1 ed. Chapman and Hall/CRC, 2016.

[18] F. Hu, "Security and Privacy in Internet of Things (IoTs): Models, Algorithms, and Implementations," 2016, p. 586.

[19] "IDS at DuckDuckGo." [Online]. Available:

https://duckduckgo.com/?q=IDS+&t=raspberrypi&iar=images&iax=images&ia=images&iai=https%3A%2F%2Fwww.digitalplanet.ie%2Fwp-content%2Fuploads%2F2017%2F08%2Fips-ids-diagram3-1024x875.jpg.

[20] K.Wang, X. Qi, L. Shu, D. Deng, and J. J. P. C. Rodrigues, "Toward trustworthy crowdsourcing in the social internet of things," IEEE Wireless Communications, vol. 23, no. 5, pp. 30–36, Oct. 2016.

[21] "What Is Network Security? | Meaning and Types of Network Security," Enterprise Security News, Endpoint Protection Solutions Blog ,Comodo, 11-Jan-2019.

[22] Padraig Scully, 19 January 2017, "Understanding IoT Security - Part 2 - IoT Cyber Security for Cloud and Lifecycle Management." [Online]. Available: https://iot-analytics.com/understanding-iot-cyber-security-part-2/

[23] Erik Walenza, 9 February 2018, "IoT Security: Why Your Toaster Needs a Firewall," Skelia, 09-Feb-2018. [Online]. Available: https://skelia.com/articles/iot-security-why-your-toaster-needs-a-firewall/.

[24] A. M. da S. P. de Moraes, "Cisco firewalls: concepts, design and deployment for Cisco stateful firewall solutions," 1. print. Indianapolis, Ind: Cisco Press, 2011.

[25] "Intrusion Detection Systems (IDS)," IoT ONE. [Online]. Available: https://www.iotone.com/usecase/intrusion-detection-systems-ids/u22.

[26] HIMANSHU, 23-Oct-2018, "Application Layer Protocols for IoT : IoT Part 11," [Online]. Available: https://www.engineersgarage.com/Articles/IoT-Application-Layer-Protocols.

[27] "What is SSL?" [Online]. Available: https://www.globalsign.com/en/ssl-information-center/what-is-ssl/.

[28] P. Kieseberg, P. Fruhwirt, S. Schrittwieser, and E. Weippl, "Security tests for mobile applications — Why using TLS/SSL is not enough," in 2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Graz, Austria, 2015, pp. 1–2.

Chapter 3

[1] "What is a Raspberry Pi?" [Online]. Available: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/.

[2] "Maksimović et al, "Raspberry Pi as Internet of Things hardware Performances and constrains", July 2015.

[3] "Introduction of Raspberry Pi 3 Model B," BINARYUPDATES, 27-Apr-2017. .

[4] "What is Losant? | Losant Documentation." [Online]. Available:
https://docs.losant.com/getting-started/what-is-losant/.

[5] David Nathan Freet, "A Security Visualization Analysis Methodology for Improving Network Intrusion Detection Efficiency", Indea, May 2017.

[6] 06-Jun-2019, "PuTTY," Wikipedia.

[7] 07-Jun-2019, "Docker (software)," Wikipedia.

[8] "UncomplicatedFirewall - Ubuntu Wiki." [Online]. Available:
https://wiki.ubuntu.com/UncomplicatedFirewall.

[9] Wikipedia. 11-Apr-2019, "Snort (software)".