INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Student research project paper

Evaluation of the security of connected objects

Yacine ANSER, Loïc DJEBAR, Eva SOUSSI, Alexandre TANEM, Evan TISSOT
4IR-SC

Tutors : Alata E. Auriol G. Caire R. Migliore V. Nicomette V.

1st February 2019 — 28th May 2019

Acknowledgments

This work has been done as part of the fourth year student research project at INSA Toulouse.

During these past 8 months, we have learned a lot about research and security. We are grateful to all our tutors, Vincent Nicomette, Eric Alata, Vincent Migliore, Romain Cayre and Guillaume Auriol for letting us the chance to make this experiment.
We would also like to thank all other students who worked with us, the ones designing the detection box, and also the ones creating another vulnerable object with who we have written the state of the art.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Student research project paper

Evaluation of the security of connected objects

Yacine ANSER, Loïc DJEBAR, Eva SOUSSI, Alexandre TANEM, Evan TISSOT
4IR-SC

Tutors : Alata E. Caire R. Migliore V. Nicomette V.

1st February 2019 — 28th May 2019

Contents

I. Introduction

Nowadays, connected objects are part of our everyday lives. They are used at work, as well as at home. The development of this IoT (Internet of Things) technology is really fast, and we now find various types of connected objects, from connected watches to connected fridges.

Fig. 1. A common IoT object : a connected watch

However, this growth has been done regardless of their security. This aspect is often ignored or not considered. On the one hand, this is due to the lack of training of professionals in security. On the other one, it can be explained by the lack of budget allocated to securing connected objects. The threats against them are real, and can seriously harm the core system of the object, sometimes until making it unable to work again. Attackers can also gather sensitive information from the vulnerable object. That's why it is important that this thinking about security issues, is done during the design part. Testing the vulnerability of the object before marketing should also be an essential step.

Our fourth year student research project focuses on this idea. The main purpose of this project is to develop a non-intrusive detection device able to detect any object's anomalous behaviour. The device is based on heuristics and on machine learning algorithms. It is physically represented by a box, gathering data such as temperature or brightness, but also network data, analysing the packets exchanged involving the object. In parallel, a vulnerable object has been created, along with normal behavior scripts, and harmful behaviour scripts, so that we can test the detection of abnormal behaviours on the object.

Four groups have worked on this project. Two of them have created the detection box and have handled the heuristics and the machine learning algorithms. Two others, including our group, have developed two different connected objects, one each, and the scripts associated. The object we have developed is a smart connected thermostat.

This research article, dealing with the security evaluation of connected objects, first presents a state of the art of the main attacks, and security breaches that we can find in the IoT field. Next, a detailed presentation of our connected thermostat is given. Then, it presents our project organisation. Finally, it gives an overview of all the functionalities implemented, how it has been done, and the scripts that have been developed.

II. Security of Internet of Things devices : State of the art

Abstract*— The fast, large-scale deployment of the Internet of Things (IoT) is raising major security concerns. This document aims at presenting an end-to-end state-of-the-art of security in this field, by describing the various security attacks that can be performed on a generic IoT system. In this state-of-the-art we link each security exploit with the IoT functionality it targets. To understand the specificities of IoT, we first describe the basic functionalities found in such systems (as defined in [1]). In a second phase, we give an overview of the different types of security attacks in computer systems. Finally, we detail which attack can be used to compromise each IoT functionality and how. This document is part of a joined work of two groups of students : the other part can be found under this reference *Internet of Things Security : A state of the art.

Keywords : IoT, Internet, Security, Devices, Attacks, Networks

A. Introduction

The Internet of Things (IoT) can be defined as the remote interconnection of numerous sensors and objects, through large-scale communication protocols and with few to no human intervention. IoT is providing us with a booming network of smart devices with life-changing applications in sectors such as health-care, life sciences, smart home, municipal infrastructure, agriculture, education [1] ... However, the fast and heterogeneous deployment of such systems raises major security concerns, as smart things manufacturers in such a booming market are likely to quickly develop small low-energy devices, at the expense of allocating the necessary time and processing power to address security issues.

In the following we will consider that an IoT system is made of three basic components [1] : a thing, the controller, and the cloud. The thing is connected to the Internet (it could be a smart home system, a smart watch...) while the controller is a program on a device such as a mobile phone or PC. The controller has two ways of communicating with the thing : either through a router if both the controller and the thing are on the same local network, or through the cloud if the controller is not on the local network (see picture below). In this case, the thing builds a permanent connection to the cloud while the controller periodically sends or requests information.

In such a system we can consider ten distinct functionalities.

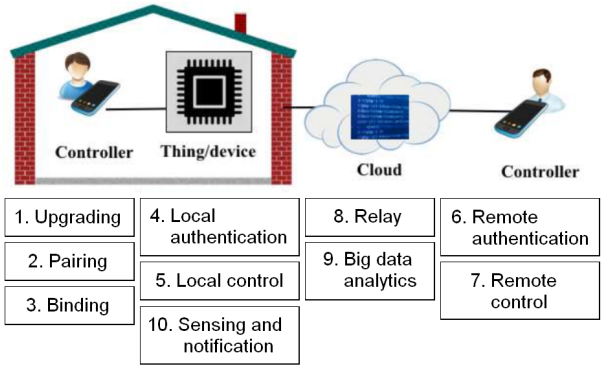

Fig. 2. Functionalities of an IoT system [1]

- Upgrading : The firmware (either an embedded Linux system or a micro-controller) allows for code updating.
- Pairing : Pairing is the act of connecting to the thing in order to setup the initial configuration. When first powered on, a smart thing usually behaves as a wireless router which allows the controller to connect. This can be done using protocols such as WiFi, Bluetooth, ZigBee, barcode/scanner, NFC.
- Binding : The process of configuring the thing's communications through the controller once pairing is done. For instance, the process of connecting the thing to the Internet (by giving it WiFi credentials for example) is binding. Connecting the thing to other devices such as client smartphones or sensors can also be a required bindings.
- Local authentication : Within a local network, the controller may be connected to an open port on the thing which is dedicated to a protocol that requires authentication : SSH (Secure Shell), RFID (Radio Frequency Identification) for example.
- Local control : Once authenticated the controller can send commands through the local network to control the thing.
- Remote authentication : If the controller is not in the local network, and has to go through the cloud, it will have to be authenticated on an application server in the cloud which will forward the commands to the thing.
- Remote control : The thing needs to register to a cloud server in order to allow for remote control.The remote application server can send commands to the thing.
- Relay : Some application servers and some routers are needed to relay authentication and control messages between all components.
- Big Data analysis : Big Data analytic capabilities are deployed on the cloud(s) which can collect and analyse data from its users.
- Sensing and notification : Smart things getting information from the environment (such as temperature) and notifying the user about those parameters or about some behaviors such as repeated login attempts.

B. Types of attacks

1) Security concepts: The security of information technology often refers to three parameters known as the CIA triad : Confidentiality, Integrity and Availability. It is a common model used for designing security policies within a system. Confidentiality ensures that information is not made available or disclosed to unauthorized people. Integrity is the certainty that the information is not tempered : trustworthy and accurate. Availability is a guarantee of reliable access to the information by users.

A security attack is an attempt to gain unauthorized access to information resources or services, or to cause harm or damage to information systems. Attacks can be passive, if they only involve someone listening to the network traffic or recording computer activity, or they can be active : someone uses gathered information to actively attack a system. Active attacks may be detectable, while passive attacks are almost undetectable. Security attacks can be performed at different level : physical, network, software, encryption.

2) Physical attack: Node jamming Jamming can be viewed as a form of Denial-of-Service attack, whose goal is to prevent users from receiving timely and adequate information.The jamming attacks are mainly classified into active and passive attack. In active jamming, the jamming node is continuously generating some malicious packets. When the victim receives these packets, a "malicious" code will start running on their machine. The passive jamming is activated only when jamming node detects some event on channel, this attack consists of occupying the communication channel by sending interference at a precise frequency, for example : jam the 2.4 GHz frequency in a way that drops the signal to a level where the wireless network can no longer function [2].

Sleep deprivation attack A sensor network is composed of nodes interacting with each other. They are organised in clusters and one of them is the head of this cluster. The main problem in sensor network remain their autonomy. Indeed, these nodes have limited batteries supplies. One of the major challenge in this field is to reduce the node's consumption. In order to save energy, sensors have different modes, including a deep sleep mode. It is a period of time during which the device does not communicate with the network, and switches off the most consuming part of its system (like transceivers for example). Sensors spend most of their time in deep sleep mode [3]. The sleep deprivation attack consists in preventing sensors from entering that mode. How to keep sensors in active mode is not complicated : sensors just have to receive what they think is a normal message. To make it look real, attackers have to be a part of the cluster and be the head of it. It is actually possible because clustering algorithms are based on reliability : each node is trusted. Algorithms can hence be easily manipulated [3]. Once attackers have compromised a node, they just have to enter the cluster and take the head of it. Malicious cluster heads pretend sending normal messages while it actually sends just enough so that the receiving device does not get to enter their low power sleep mode. As a result, the target will not get the opportunity to save energy. Its consumption is going to raise significantly [4] and hence, its life-time is going to be shortened. Sleep deprivation can be classified as an active attack.

Physical damage When the device is accessible, the attacker can damage components of the system [5]. It can lead to the loss of some features, and sometimes even make the device unavailable, like in a Denial of Service attack. The attacker can also plug an external device into the system. He can access it and do whatever he wants with it.

Social Engineering There is another type of physical attack : when the attacker contacts and manipulates users who own an IoT system. What he can typically do is obtaining information to guess more easily the user's password for example. The user doesn't always give intentionally these indications. However, we tend to create passwords linked with our lives so that it is easier to remember, and this can be exploited by attackers. Attackers can also pretend they are suppliers, and ask for private information that people wouldn't usually give, but they feel confident to give to their suppliers. Not only passwords are affected. The attackers can also get sensitive information on the system that can be used for wrong purposes.

3) Network attacks: Traffic analysis attacks Traffic analysis is the process of intercepting and examining messages in order to deduce information from patterns in communication. It can be performed even when the messages are encrypted and cannot be decrypted. It leads to a "degradation of network performance, it increases packet collision, increases contention and creates traffic distortion" [6].

Spoofing Spoofing is a malicious practice in which communication is sent from an unknown source disguised as a known source to the receiver. Spoofing can take on many forms : IP, ARP, DNS, Email [7].

- IP Spoofing : The attacker sends packets to the victim using the address of a trusted machine as source address. He has to neutralize the trusted machine so that this machine is unable to send packets to the victim [7]. This can be done by SYN flooding. Then, the attacker have to guess the correct sequence number, and try to communicate with the victim. The only major complication in this attack remain in the fact that, if it works, the victim will send its answers to the source address used, so to the trusted machine and not to the attacker. This is a blind attack[7].
- ARP Spoofing : The Address Resolution Protocol enables to find the MAC address corresponding to a given IP address. A partial mapping called the ARP cache, is kept on every router and machine. The main purpose of ARP spoofing is to corrupt this ARP cache, by associating the attacker's MAC address to the IP address of the machine he wants to spy on. Hence, by this process, all the packets destined to the victim will be received by the

attacker. But how to corrupt the cache ? This is simple. When a router gets a packet with IP X, and doesn't have it in his cache, ARP broadcasts on the network a request to know the MAC address of the machine with IP X. Only the machine who has IP X answer to the router with its MAC address. This is where attackers can interfere. They send forged ARP replies to the router indicating they have IP X with their own MAC address. The router will update his table with the wrong association.

- DNS Spoofing : When users want to visit a website, a DNS request has to be made to convert the domain name they enter into the IP address of the website. Attackers can hack DNS servers to modify the IP address of a domain so that when users think they are connecting to their bank website for example, they are connecting to a fake site. So that users believe in it, the graphical interface has to be similar. Attackers can then get sensible information.
- Email spoofing : It is when the email seems to come from a legitimate source while it actually is from an impostor[7]. It makes people more likely to open emails that can be meant for spreading viruses.

Man in the middle In most cases this attack requires three actors at least. There is the victim, the entity with which the victim is trying to communicate, and the "man in the middle," who is intercepting the victim's communications. The attacker can access the communication media between the two end points[8], and optionally modify their messages.

Man in the middle attacks can occur on different layers of the OSI Model, from the Data Link layer to the Application one, and even in cellular networks[8]. It is based on different approaches depending on the layer : IP spoofing for example for the network and transport layer, interception of the encryption keys through the analysis of the network communications[8].

Man in the middle attacks compromise all three parameters of the CIA triad : confidentiality in the way that the attacker accesses the communication, integrity by being able to modify messages, and availability by having the opportunity to delete part of the exchange.

DoS (Denial of Service) DoS attacks can be carried out by network insiders and outsiders. Their goal is to make the network unavailable to authenticate users by flooding and jamming with likely catastrophic results. By flooding the control channel with high volumes of artificially generated messages, the network's nodes, onboard units and roadside units cannot sufficiently process the surplus data. Hence, devices become unavailable. DoS can affect physical layer, link layer, network layer, transport layer and application layer.

DDoS (Distributed Denial of Service), happens when attackers compromise a lot of vulnerable devices to lead a synchronised attack on a victim [6].

4) Software attacks: Malicious software (virus, worm, spyware, trojan horse) These malware have a behavior which allow the attacker to endanger the three points of the CIA triad. He may get privileges on a system to modify the way it works, to access, alter or destroy information. There are many types of malware : virus is a self-replicating software that infect a computer by inserting its own code in legitimate software. A worm is also a self-replicating malicious software but it does not need host software. A trojan horse is a malicious code that hides it-self within seemingly harmless programs. It does not replicate it-self but it can collect information or create backdoors.

Code injection The aim of this attack is to inject a code in a vulnerable application in order to change the course of its execution [9]. There are different types of code injection attacks :

- SQL injection : it is used to attack data driven application thanks to an injection of a SQL statement in an entry field which allow the attacker to steal, modify or destroy data.
- Script injection : it consists in injecting malicious code. One of the most known script injection attack is XSS (cross site scripting). It allows the attacker to inject client-side scripts into web pages viewed by other users. For instance, the script can steal cookies or redirect the user to another web page.
- Command injection (or shell injection) : it allows an attacker to execute an arbitrary command on the host operating system via a vulnerable application.

Buffer overflow A buffer overflow is software coding error that an attacker could exploit to access into the system. It consists in introducing into a buffer of fixed length (reserved for the software), more data than the buffer can handle. This error can be exploited by an attacker to insert malicious data into the program memory, like its own piece of code to change the program behavior.

5) Encryption attack: Cryptanalysis attacks Cryptanalysis is the science of breaking or decoding cipher text into its corresponding plaintext without prior knowledge of the root key or without knowing the real way to decrypt the ciphertext. It is a technique used for searching about the drawbacks in cryptosystems design. The main methods of cryptanalysis are :

- Known-ciphertext attack : In this attack , the attacker has access to a given set of ciphertexts , he does not have access to corresponding cleartext, however, this methode is successful when correspdonding plaintext can be determined from a given set of ciphertext. Occasionally, the key used to encrypt the ciphertext can be determined from this attack .
- Known-plaintext attack : The known-plaintext attack is a cryptanalysis model assuming that the attacker can access a set of plaintexts and the corresponding ciphertexts encrypted by the same secret key.
- Chosen-plaintext attack : In this attack, we assume that the cryptanalysis has access to a device that

provides the capability to choose arbitrary plaintext to be encrypted and obtain the corresponding ciphertext. With such capabilities, he will encrypt selected plaintext, and then analyze the resulting ciphertext to gain further information that reduces the security of the encryption scheme.

— Adaptive chosen-plaintext attack : In this attack, the analyst is able to make a sequence or series of interactive queries. Subsequent plaintexts are able to be selected based on the results from the previous instances of encryption.

Weak encryption algorithms To encrypt communications, connected devices use encryption algorithm. They may be proprietary or well-known by everyone. Most encryption algorithms can be broken and attackers can reveal information, especially if they have enough time, will, and resources. Encrypting data avoid sensible information like passwords or encryption keys to be revealed. Therefore, avoid using algorithms that are flawed or weak (DES, 3DES, MD5, Sha1, RC4) or non-standard (home-grown) algorithms.

Side Channel Attacks Side-channel attack is an attack performed based on gained information such as timing information, power consumption, electromagnetic leaks or even sound. Thanks to these information, the attacker can deduce how the circuit works and what data it is processing. An example of this vulnerability is the electromagnetic emanations of keyboards. Just analysing it, it is possible to determine what keystroke was originally pressed[10]. We almost always use our keyboard to enter passwords and this technique may allow attackers to get sensible information, standing in another room with affordable equipment .

The attacker can also extract secret cryptographic keys from the analysis of external parameters.

C. IoT vulnerabilities

Since we are presenting this document as a joined work with a second team, we will only go through these functionalities : binding, remote authentication, remote control, relay, sensing and notification. You can find the other functionalities detailed in their paper.

1) Binding: Binding connected objects is the process of establishing a link between the object and other elements essential to its functioning. These elements may be the Internet, the user through its mobile phone, deported parts of the thing, other connected objects etc...

For example, a typical scenario could be a user who wants to install his wireless IP camera : once the camera is powered up, it will get into AP mode and offer a WiFi network. The user may then connect to it with his smartphone. Then comes the biding : on his smartphone, the user has access to an interface for providing the local WiFi password, allowing the camera to connect to the Internet.

This biding can also be done automatically by the thing using "Service Discovery" protocols : SDP[11]. SDPs exist within multiple technologies : Bluetooth SDP, MQTT Discovery (MQTT is a IoT-dedicated protocol) ...

Bluetooth Bluetooth can be used to connect the thing to other devices like sensors, actuators, smartphones. The development of BLE (Bluetooth Low Energy) is very used in IoT since connected objects are very concerned about energy issues.

On September 12, 2017 researchers from Armis reported several security vulnerabilities concerning Bluetooth on many devices[12]. One of these vulnerability affects the Linux kernel, used by a vast majority of connected objects, «IoT devices such as TVs, watches, cars, and even medical appliances.»according to Armis. We will present here exploits for two vulnerabilities.

The first one [13], present in the Linux kernel from 2.6.32 (12/14/2009) to 4.13.1 (09/10/2017) so potentially on a huge number of non updated objects, is based on a stack overflow allowing the attacker to do remote code execution (RCE). It is caused by the processing of packets in the L2CAP layer. The attacker is able thank to this overflow to execute code on the victim object and then take control of it without the devices being associated together. L2CAP is the Bluetooth's equivalent of TCP, it is responsible for managing connections between devices. For establishing a new L2CAP connection, the two devices have to exchange in both ways "L2CAP configuration Request" and "L2CAP configuration Response". The vulnerability can be triggered by sending a first configuration request with a given flag to the attacked device which sets it into "Pending" state. The next configuration response sent will make the buffer overflow in the memory stack : the attack is realized.

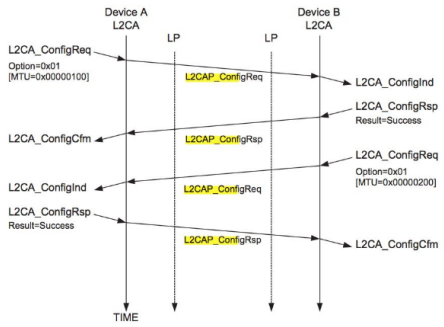

Fig. 3. L2CAP Basic MTU Exchange [12]

The second one, CVE-2017-1000250 is an attack causing Information disclosure. By sending particularly crafted SDP Bluetooth packets to the victim, the attacker can obtain memory bits in response from the Bluetooth process. These bits may be very sensitive because Bluetooth encryption keys are contained into the process memory. Obtaining these keys would allow the attacker to decrypt and insert Bluetooth packets in the communication. An attacker could also obtain information about the software version used by the device and then achieve the first attack in an easier way to take control of the victim's device.

WiFi During the binding, the thing can also connect to a Wireless AP, this connection may be vulnerable to Evil

twin attack. Indeed, without any properly implemented protection [14], an attacker could impersonate the legitimate AP to which the thing will connect by using the same SSID / BSSID. So then, the entire traffic will go through the attacker AP allowing him to have access to information and even modify it if packets are not encrypted and, respectively not signed.

2) Remote authentication: Remote authentication protocols are used when the controller is not in the local network, and has to go through the cloud to access the IoT device. To do so, it will have to be authenticated on an application server in the cloud which will forward the commands to the thing. When using a remote IoT device, authentication is the first phase and one of the most important on a security level because if an intruder manage to get past this phase, he can gain total control over the device. For this reason, this step is generally very well secured because developers have started to understand that security will take a very large part in IoT in the future. The OWASP Internet of Things (IoT) Project [15] reflects this awareness. The main defaults in remote authentication may occur in those areas [16] :

- Insecure Web/Cloud/Mobile interface.
- Insufficient Credential Authentication/Authorisation.
- Lack of Transport Encryption.

The Web/Cloud/Mobile Interface The Web Interface makes the connection between the user and the remote connected object. It allows the user to choose and send the orders to the IOT device. Insecure web interfaces are considered the Achilles's heel of IoT security defences [17]. Here is a review of the main vulnerabilities existing in this field :

- Cross-site scripting (XSS) : This attack consist in injecting a script into a web application that accepts inputs but does not properly separate data from executable code before sending these inputs back to a user's browser.
- SQL injection : This vulnerability is already explained in part II.
- Flaws in the Session management : To allow the interface to be accessible to several users simultaneously there must be session. They allow the process of securing multiple requests to a service from the same user or entity. Once authenticated, the web applications often use cookies to authenticate the user on the different parts of the website. If the cookies are not secured enough, the cookies can be stolen and used to gather unauthorized data.

Insufficient Credential Authentication/Authorisation This part will be focused on the vulnerabilities that can occur because of the flaws of the authentication mechanism used to check the validity of the credentials.

- Account enumeration : This vulnerability allows an intruder to use the response to a failed authentication attempt that indicates whether the authentication failed due to an incorrect account identifier or an incorrect password to find all valid accounts given sufficient time [18]. This can be considered a vulnerability because it facilities the task of password cracking by allowing attackers to discern the valid set of account identifiers.
- Weak default credentials : This one is easy to understand, the simpler the password is, the simpler it will be for the intruder to find it.
- Lack of two factor authentication : Two-step verification is a method by which a user can access a computer resource after submitting two separate pieces of identification to an authentication mechanism. So if only one verification is required, it will weaken the all authentication mechanism.

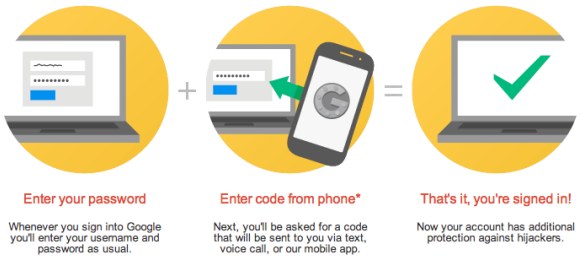

Fig. 4. Example of a two-Factor authentication [19]

- Weak account lockout settings : The account lockout threshold security setting determines the number of failed login attempts that causes a user account to be locked out. A locked-out account cannot be used until it is reset by an administrator or until the lockout duration for the account has expired. If this protection is not implemented, an intruder may attempt to brute-force the logins until a valid one is found. This mechanism can for example protect from an account enumeration attack .

Lack or Flaws of Transport Encryption

- Unencrypted services via the internet : Nowadays all serious website are using the HTTPS [20] protocol to secure there communication rather than the old HTTP. However if a web application that permits to authenticate a user in order to use an IOT device was using HTTP to carry the credentials, it could be a huge security breach.
- Poorly implemented or misconfigured SSL/TLS : TLS is a cryptographic protocol that allows end-to-end devices to communicate safely over networks and is mostly used for internet communications and online transactions [21]. TLS is SSL's successor. There is a one-way TLS and a two-way TLS. One-way TLS enables the TLS client to verify the identity of the TLS server while in two-way TLS, the client verifies the identity of the server followed by the server verifying the identity of the client [22]. To have a good protection, certificates need to be delivered by a trusted authority and the encryption process has to be strong.

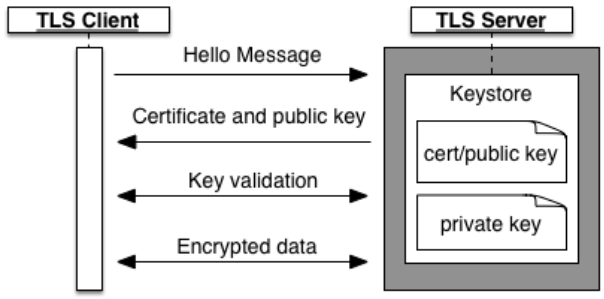

Fig. 5. One Way TLS/SSL [22]

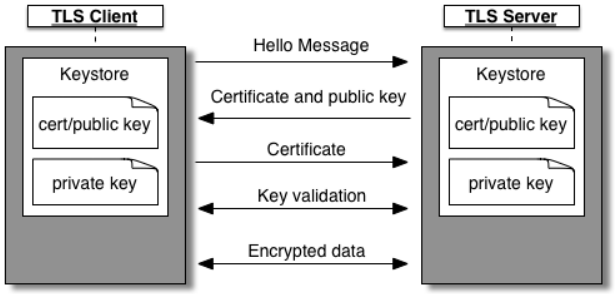

Fig. 6. Two Way TLS/SSL [22]

— TLS flaws : Over the years, a few vulnerabilities were discovered [21] :
  — BEAST (2011) : The Browser Exploit Against SSL/TLS is an exploit that took advantage of a weakness in the cipher blocking chain (CBC) to extract the unencrypted information in an encrypted session.
  — CRIME and BREACH (2012 and 2013) : The creators of BEAST authored the security exploit Compression Ratio Info-link Made Easy, which allows a hacker to harvest the content of Web cookies, even when compression and TLS are used. One huge security breach could occur if the intruder manages to recover the authentication cookies and hijacks authenticated web sessions. Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext, or BREACH, is built on CRIME and extracts login tokens, e-mail addresses and other information.
  — Heartbleed (2014) : Heartbleed enables hackers to steal private keys from what should be secure servers. Infected servers were left wide open to let anyone on the Internet read the memory in systems being protected by a vulnerable version of OpenSSL. The breach let threat actors steal data from servers or listen in on conversations or even spoof services and other users.

  This shows the importance of updates that can be a solution to counter those flaws.

3) Remote control: In this part, we will focus on the vulnerabilities concerning the remote control. Remote control allows the user to control the thing without being connected to the local network. Thus he can use the object from anywhere in the world.

There are two ways of controlling remotely the thing :
- Via a server relay (cloud)
- Via a direct connection

Connection to a cloud

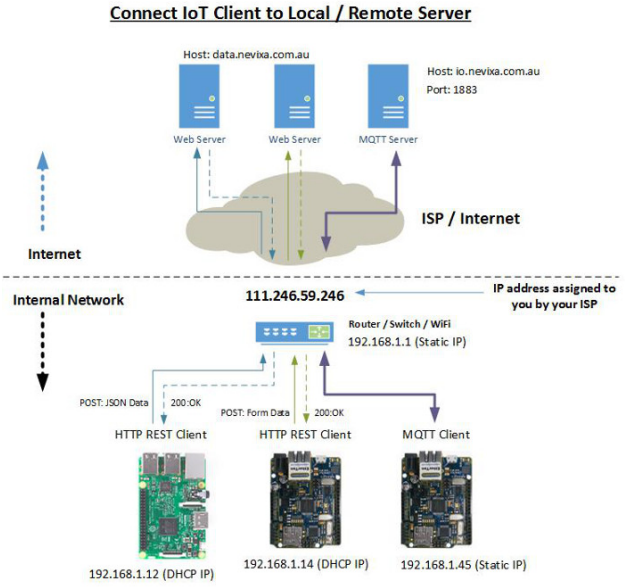

Fig. 7. Architecture of a connection with a local/remote server [23]

With this architecture (Fig 7.), the thing initiates a continuous connection to the server. When the user wants to control the thing, he connects to the server and transmits his request. The server relays it to the thing. In the same way, the thing responds through the server. Vulnerability during the phase of control has been discovered by researcher on this architecture and can be exploit using a 'phantom device'.

The purpose of this attack is to get control of a legitimate device thanks to a fake device. To perform the phantom device attack, a deep analysis phase is required. The attacker has to know how the thing, the controller and the cloud work and interact together. After identifying design flaws and exploiting them, the attacker can remotely hijack or substitute the target's device.

Below is presented an attack on a Alink device (Fig 8.) which leads to a remote hijacking and an attack on a TP-Link (Fig 9.) device which leads to a remote substituting.

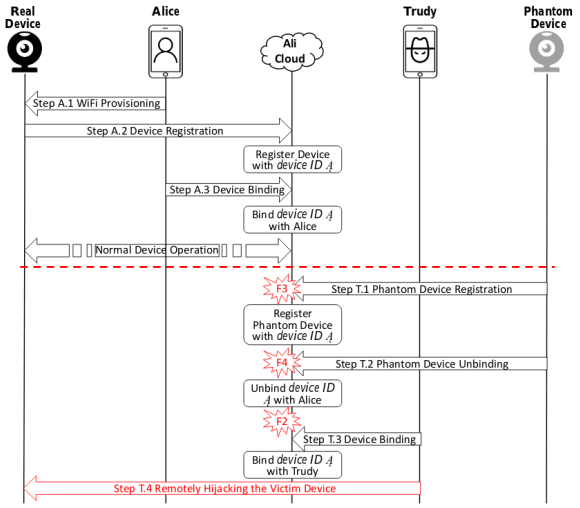

Fig. 8. Workflow of an attack on a Alink device [24]

On the top of the Fig 8. is represented the legitimate scenario for Alice. The attack starts at the step T.1 when Trudy connects to the cloud with credential obtained on the Internet and guessed device identity information (it is the same as the step A.2). The cloud accepts this request and registers both real and phantom device with the same device ID, but still keeps device ID A bound with Alice. At this moment, Alice actually binds the phantom device and real device at the same time. Now Trudy logs in the phantom device and sends the device-side unbinding request to the cloud. Finally, Trudy binds the device with her own user account and completely hijacks the real device. At the end, Trudy can control Alicia's device.

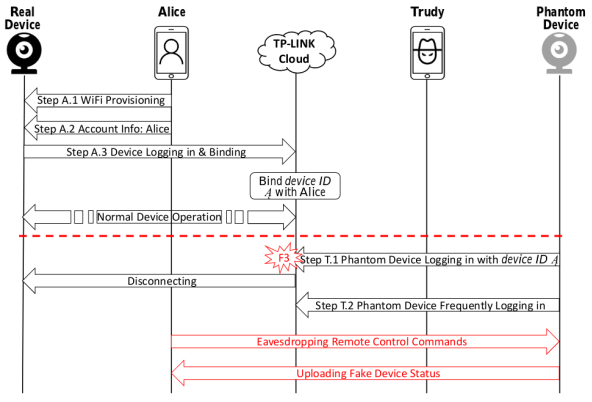

Fig. 9. Workflow of an attack on a TP-LINK device [24]

The top of the Fig 9. shows the original workflow. At the bottom, Trudy initiates a connection between the phantom device (with the device ID of Alice) and the cloud. Consequently, it disconnects the original connection with the real device and establishes a new connection with the phantom device. At the end, Trudy can send false information to Alice and eavesdrops remote control commands.

Direct connection

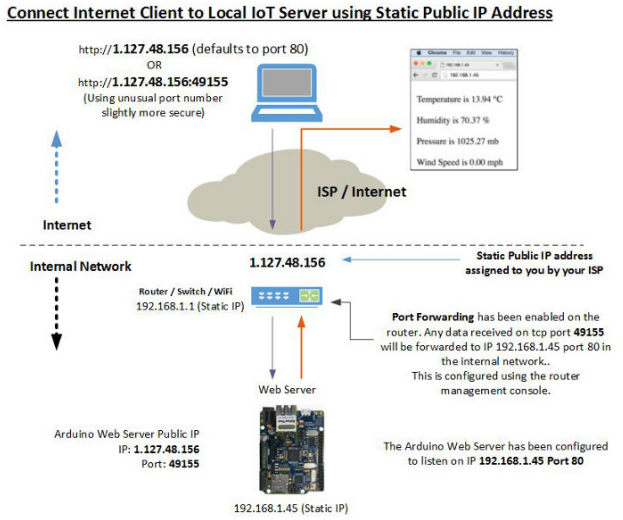

Fig. 10. Architecture of a direct connection [23]

With this architecture (Fig 10.), the user connects directly to the thing. The thing can be assigned a public IP address or it can be behind a router and therefore port forwarding is mandatory. In both case, the thing is accessible from anyone on Internet.

The main vulnerability is that the default credentials are used. The exploit of this vulnerability leaded to a major attack in 2016 named Mirai. The attackers infected many devices which were used to perform DDoS attacks.

UPnP can also be a vulnerability for IoT devices. UPnP is a protocol which is used by a significant number of IoT devices. This protocol allows easy communication between devices and is most of the time enabled by default. But it suffers from serious security issues which can expose private networks to attacks from the internet. [25] Basically, UPnP is composed of two services :

- Service Simple Discovery Protocol (SSDP) which is responsible for advertising available services and responding to discovery request
- HTTP service which hosts the Simple Object Access Protocol (SOAP) interface. SOAP allows others systems to command the device.

In the document [26], it is estimated that 81 million unique IPs were found to expose the SSDP service to the public internet and 17 million unique IPs expose the SOAP service to the public internet. These vulnerabilities come from misconfiguration or usage of outdated UPnP implementations. Attackers can exploit them to steal data or to use the device as a bot to perform attacks.

4) Relay: What is the relay ?

The IoT environment is mainly composed of an authentication server (embedded in the object), a controller, that can be on a PC or an app on the smartphone, and the IoT device. Today most builders add the ability to authenticate and control the device remotely through the

cloud who is attached to an authentication server. The object establishes a permanent connection to the cloud. The controller checks or requests information from the object across the cloud[1] [27].

We call RELAY the phase that relay the remote controller and the connected object through the cloud, this phase begins with mutual authentication of the server cloud and the remote controller [28]. If authentication succeeds, the controller will be able to control or request information from the IoT device otherwise he will not be able.

An attacker can try to authenticate himself with the cloud server instead of the legitimate controller and send control messages to the object or he can add an object instead of the real object [29]. We will discuss in the following part the different ways that an attacker can use to achieve his objectives.

Model attack

The adversary can engage various illegal ways to acquire sensitive information of a user. The following part will present the definition of the attack model that can be used by an adversary to acquire information from the system [30].

a) Eavesdropping : If the communication messages between the server and the remote controller or the server and the connected object pass through an insecure channel. The attacker can passively listens to network communications to gain access to private information, such as node identification numbers [27]. The attacker can use this private information to authenticate himself to the server and control the object or authenticate a fake object to the server.

b) Traffic analysis : Traffic analysis is a method to analyze the messages eavesdropped during communication between the device and the server. By analyzing this information, the attacker can acquire information needed to authenticate the fake object to the server or usurps the identity of the real controller.

Cryptography is the standard defense against eavesdropping and traffic analysis attacks. However, cryptographic primitives are not suitable for low-resource smart devices due to their low computation power, limited battery life, small size, small memory, and limited power supply[31].

c) Man-in-the-middle attack : It is one of the classic attacks that can be executed in a wireless sensor network environment. The attacker attempts to establish an independent connection between the server and the controller or between the object and the server by pretending to be a object of the network by using IP spoofing [32]. The attacker may be in a passive or active state. In a passive state, it simply relays each message between the object and the controller with the intention of launching a spying attack. In active state, it can alter the intercepted data to try to break the authentication process. The following figure gives an example of a MiTM between the authentication server and the object .

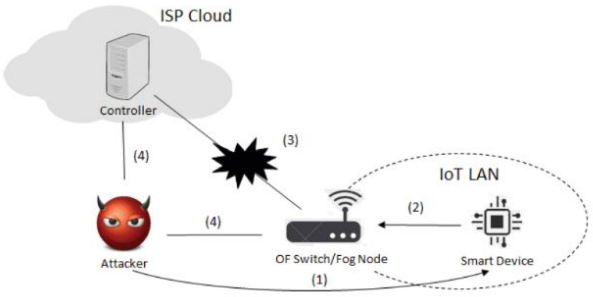

Fig. 11. MiTM Attack [32]

d) Denial of service : A denial of service attack is an attempt to make a system resource unavailable to the intended users. An attacker can send a large number of authentication requests to make the cloud server unavailable and thus prevent the controller from identifying and sending control messages to the object or retrieve information .

e) Replay attack : This attack is similar to a Man-in-the-middle attack. It occurs when an unauthorized user captures network traffic, then communicates with the recipient (the cloud server) while acting as the original sender (the controller or object)[27]. To gain the trust of the authentication server, the adversary transmits a message obtained by indiscreet listening of a regular communication between the server and a device during the authentication process (the controller or the object)[28]. The attacker usually sends a packet already received by the server. For example, messages from an authorized user connecting to the object may be sniffed by an attacker and replayed the next day. Although messages can be encrypted without the attacker knowing what keys and passwords are, re-transmitting valid login messages allows access to the object.

f) Leak of verifier attack : In this type of attack, the attacker enters the system and steals information from the server database. This information is often encrypted, for this the attacker will do what is called a offline dictionary attack, he can then use this information to calculate vital information such as the password of a legitimate user [27].

5) Sensing and notification: As we have seen before, an IoT device is composed of three components : a thing, the controller and the cloud. That thing, can be smart. Indeed, it may sense the brightness of the room it is in, and notify the user if the brightness is too low or too high. There are a lot of others parameters that a thing can sense : temperature, magnetic fields, pressure, smoke or gas...

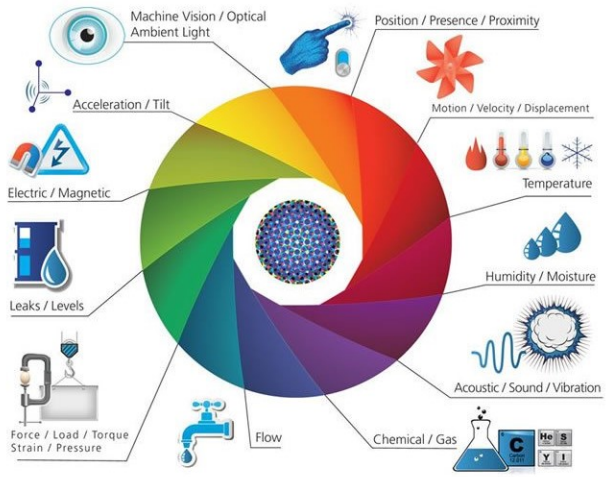

Fig. 12. The different types of sensors [33]

Sensing is also about notifying the user of unusual behaviour. If an attack is detected, for example if there were too many combinations of login-passwords used to enter the system, a thing can send a notification.

This system is not completely safe.

a) Physical influence over the sensors.

To get measures, there have to be sensors and attackers may place an external element aimed at modifying what the sensor measures. For example, put a device that produces heat next to the temperature sensor so that it indicates a higher temperature than the one in the room. It can have a lot of consequences. Indeed, if the change of the temperature measured provokes some events to regulate it, modifying this value will enable the attacker to start a sequence of events that would never happen in the current situation. In the case of temperature sensor in a server room, attackers can make the temperature look lower than it is, so that the system reacts and increases it. The actual temperature in the room may exceed the threshold and finally leads to overheating servers. This type of physical attack requires a direct access to the sensors and the knowledge of the system's reactions so that it leads to the wanted outcome.

The devices may also be attacked physically. Usually, the debugging ports remain opened after testing and validation process[1]. Among them, there are the JTAG debugging ports. On Intel processors, the DCI (Direct Connect Interface) can make JTAG functionalities available simply connecting via USB 3.0 [34]. DCI can be activated through :

- EFI Human Interface Infrastructure
- PCH Strap (Intel Flash Image Tool)
- P2SB device

For example, via a PS2B device, if the BIOS does not prevent us from writing into the ECTRL register, we can "-by using the ability to save the configuration between restarts and after power down- enable DCI once and then use the JTAG interface as a hardware backdoor to the system (and bypass the lock screen, for example)" [34]. Many motherboard manufacturers do not block this register. Consequently, we can get into the BIOS bypassing all the security measures. Attackers can then access private data, such as the measures from a sensor or information on the IoT device. They can even send false notifications or modify the code so that it changes the sensing features.

b) Prevent notifications from being sent

Nowadays, most of the communications between sensors are wireless and use WiFi, zigBee or BLE (Bluetooth Low Energy) to only talk about some protocols. Jamming is a radical solution to prevent sensors from communicating. As we know, they use a predetermined range of frequencies to transmit data : Wifi for example operates on the ISM band at 2.4 GHz or the UNII one at 5GHz. All the attackers have to do, is analyse the frequency spectrum so that they get the frequency at which sensors are transmitting. Then for a few hundreds euros, they can get a jammer that can work up to tens of meters from the emitting source. The sensors's notifications will not be able to be sent.

c) Sending legitimate looking notifications.

An attacker can also exploit the protocols used by the IoT devices. To transfer data between devices, the most widely used protocol is MQTT (Message Queuing Telemetry Transport). MQTT is a publish-subscribe protocol, in which communications are mediated by brokers. Clients can publish messages to a broker and they also can subscribe to it, in order to receive some messages from it.

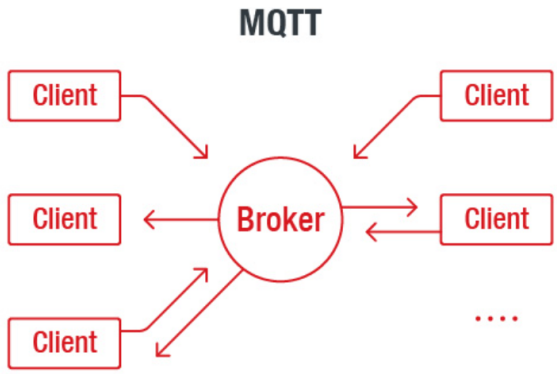

Fig. 13. Interactions model of MQTT [35]

This protocol does not include security in his default version : usernames and passwords are sent in clear if encryption is not independently handled with SSL [36]. A eavesdropping attack can be performed to intercept these information, and the message format. Hence, attackers may conceive packets similar to the one produced by a normal sensor and make it look like a legitimate one. Attacks have already been made, showing that attackers can open the garage door without the owner's consent.

What can also be done is preventing communications from reaching their targets. In MQTT protocol, clients publish messages to a broker which redistribute it. The

weak point of this way of communication is the broker. According to CVE-2017-7653, a vulnerability has been found in Mosquitto, the most popular broker : "The Eclipse Mosquitto broker up to version 1.4.15 does not reject strings that are not valid UTF-8. A malicious client could cause other clients that do reject invalid UTF-8 strings to disconnect themselves from the broker by sending a topic string which is not valid UTF-8, and so cause a denial of service for the clients" [37]. We can find a quick demonstration of this attack here [35] through a miniature version of a Rover used to lift heavy loads. It receives commands via MQTT and actuates it. Usually, the broker sends orders to the Rover. When the broker is vulnerable, a malicious device can subscribe to the broker using an invalid string. It crashes the broker that is now in DoS position and disconnects connected clients. As a consequence, the notification system doesn't work, and the different parts of the system are no longer communicating with each other.

As we have seen, there are a lot of options to get into the sensing and notification process. The sensor can be physically accessed, influenced by external parameters, hacked through defaults in protocols or by the lack of security features.

D. Conclusion

This paper investigates the major security issues for each basic functionalities in IoT system i.e upgrading, pairing, binding, local authentication, local control, remote authentication, remote control, relay, big data analysis and sensing and notification. A detailed explanation of some of the different possible attacks in each feature has been given after introducing it.

We have seen that, if unsafely manufactured, the Internet of Things can become an open gate to security attacks, especially considering the amount of sensitive data it carries. Computer systems can be exploited in many different ways, and a such a wide remote-operating network is only as secure as its weakest component. It is primordial that manufacturers are provided with security guidelines and best practices that will enable them to build up appropriate security plans for each device available on the market, but also for their cloud services.

III. Project organisation

Here are some elements concerning our organisation during this student research project.

A. Project parts

We tried to separate our time into three main parts :

1) Designing the object: We first tried to find a connected object that we can use in our everyday lives. We chose the technologies and protocols we wanted to use. For this phase, we had to do it taking in account the attacks that could be made on these protocols. We used a lot the research we had made for the state of the art part. Once we had decided what to do and we wanted to use, we had to chose which hardware to use among those available in the department.

2) Implementing the object: In order to implement this project, we have distributed the tasks between us :

- Tissot E. : IoT node, bridge MQTT-BLE
- Djebar L. : Handling of the Peltier Module, 433 MHz beetween Raspberry and Arduino
- Anser Y. : Remote wifi connection
- Tanem A. : Management of the main station, Handling of the screen
- Soussi E. : Monitor data remotely

3) Implementing the attacks: We worked together to implement the attacks on different parts of the object.

B. Organisation tools

We used Trello to manage the tasks we had to do. It is a simple collaboration tool which allows to group information into boards. We can easily see what has been done, what we are currently working on, and what we have left to do.

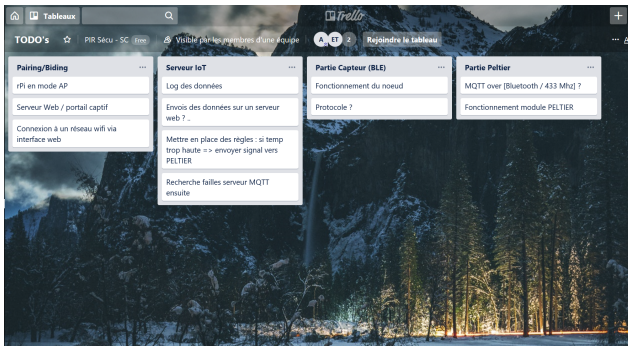

Fig. 14. Trello's view

All our code are on a git repository at https ://github.com/alexandretanem/pir_secu.

C. Planning

Designing	Object	Attacks
1st March 2019	1st April 2019	20th May 2019

We faced some problems designing the object :

- Connecting the screen to the Raspberry Pi 3+
- Dealing with the AP mode of the Raspberry PI
- Operating the 433 Mhz communication between the Raspberry PI and the Arduino

That's why we were late on our original planning, and spent more time finishing the implementation of the smart thermostat.

During the project, we had group working sessions, in which ones we had meetings at the beginning. Meeting's purposes were to talk about the progress of each one, discuss the problems we had and ask for help if we needed it.

IV. Our connected object : the smart thermostat

For the purpose of this project, we had to develop a connected object which could use typical IoT protocols and services. The main characteristic of an IoT object is to communicate with the user remotely. To fit this idea, we have developed a connected smart thermostat. It is composed of several components. There is a main station, and two deported objects. The first one is an IoT node which measures temperature, pressure and humidity. The second one, is a Peltier module which can heat or cool in order to regulate the temperature measured by the IoT node. This thermostat is a common object that we can find in our everyday life, possibly with bigger heater to have a real effect heating up large rooms.

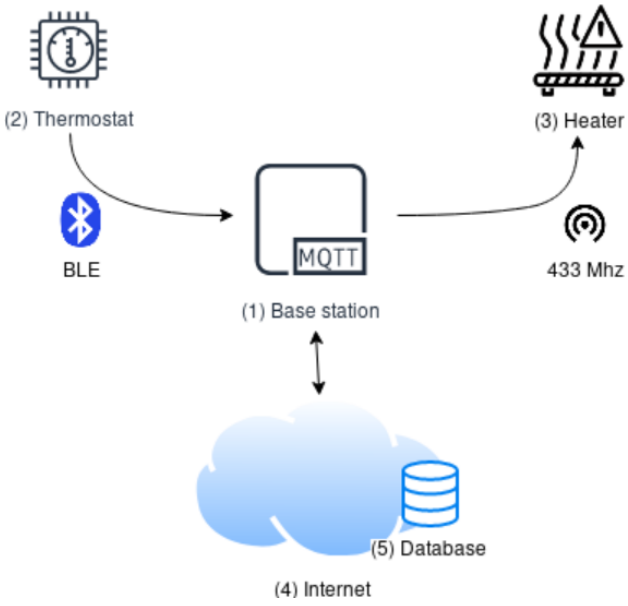

Fig. 15. The smart connected thermostat

The core of the system is implemented on a Raspberry Pi 3+. This base station (1) handles the communication between all the parts of the system. It is also connected to the Internet. The base station is connected to the IoT node B-L475E. The connection is made through BLE (Bluetooth Low Energy). A bridge is used on the base station to convert the information we gather from BLE to an MQTT broker. The main purpose of this link is to gather temperature, pressure and humidity from the IoT node (2).

When it receives the data, the base station interprets the measurements of temperature. It then decides if its environment is warm enough or not. To do so, it compares the measurement of temperature received from the IoT Node with a threshold. If the temperature exceeds it, a command is sent to the Peltier module through a 433MHz connection. The module is held by an Arduino card, whose purpose is to turn it on or off physically. As a consequence, the command is sent to the Arduino card, the Peltier module (3) has to heat up the room.

Eventually, the base station sends all the data from the IoT node to a web server (5). Every 15 minutes, we send requests to the web server to get the maximum value of each field since last request. The results of the query are compared to thresholds and displayed on a screen.

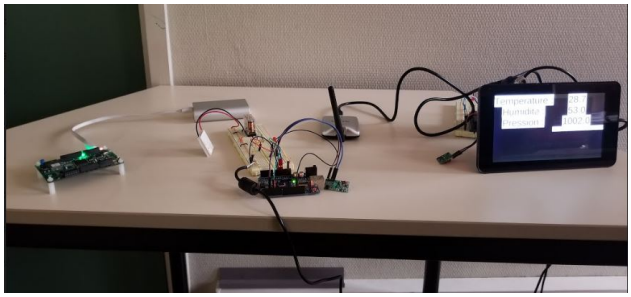

Fig. 16. Our built system

We can see on Figure 16 our built system. On the right, we can obviously see our screen. Behind it, is the raspberry Pi, hidden in the picture. In the middle we can see the Peltier module, and on the left, the IoT node. We can easily put the different parts of the object away from each other.

A. Manage connections

To allow the user to connect to the object, it needs to be able to choose the Wifi network he wants to use. To make this possible we used two Wifi cards, one as an access point, and the other one to connect to a wireless network. We first used Hostapd (Host access point daemon) which is based on the IEEE 802.11. It is a user space software access point capable of turning normal network interface cards into access points. Then we installed raspAP which is a simple responsive web interface to control Wifi. RaspAP uses lighttpd as a web server which is very conservative in its resource usage and memory footprint. To prevent unauthorized people from accessing we have added an HTTP authentication.

B. Collect measurements

1) Sense the environment with the B-L475E-IOT01A board: The B-L475E-IOT01A is a board suited to IOT applications since it is composed of many sensors (pressure, humidity, temperature, gyroscope, magnetometer) and communication interfaces (BLE, Wi-Fi, NFC). The whole system is managed by an ARM Cortex M4 microcontroller. In our project this node acts as a measuring device which sends the measurements to the base station via BLE. ARM Mbed OS, an embedded operating system, is deployed on the board in order to make the development easier. In this way, Bluetooth connection can be initiated with only few lines of code. We also make use of the library provided by STMicroelectronics to retrieve measurements from the sensors.

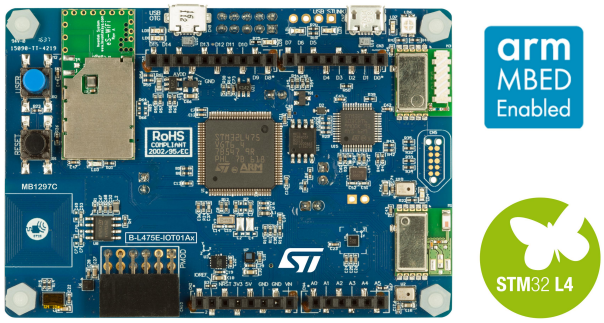

Fig. 17. B-L475E-IOT01A board

2) Bluetooth Low Energy connection: BLE is a wireless personal area network technology. This protocol is composed of a physical layer, a GAP layer and a GATT layer. Physical layer is responsible of the for translation of digital symbols over the air. GAP layer is used to discover devices and to manage connections. GATT layer is used for data communication between two connected devices.
GATT is an acronym for the Generic Attribute Profile, and it defines the way that two Bluetooth Low Energy devices transfer data using concepts called Services and Characteristics.

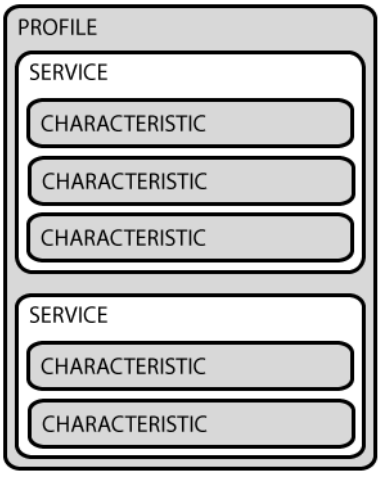

Fig. 18. GATT Structure [38]

Characteristic is an entity which encapsulates values (e.g. Heart rate measurement characteristic is composed of the heart rate measurement value in bpm, the energy expended value in Joule and others informative values...). Characteristic has also a unique ID, properties (read, write, notify) and permissions. Service is a collection of characteristics. GATT is a client/server mechanism. The server holds the services and characteristics definition. The client reads or writes data from or to the server.
Concerning our project, the node acts as the server while the base station acts as the client. Environmental Sensing Service is exchanged between the two devices. This service is composed of the pressure, humidity and temperature characteristics. The behaviour of the data transfer is configurable by setting different property values. If the property notify is set, the node send a value to the base station each time the measurement changes. If the property readable is set, the base station has to request the value to the node. We decided to implement the second option since the measurements always change (tiny fluctuations of temperature). If we had chosen the first option many values would have been sent leading to a high power consumption.

3) Receive measurements on the base station: The base station receives the measurements thanks to a program written in Python. The bluepy library is used to manage the Bluetooth Low Energy connection. Once a measurement is received, it must be shared with the entity which manages the Peltier module. MQTT protocol is a good choice to achieve this goal. It is widely used in IOT applications. It has a client/server model. A client connect to a broker (server) and can subscribe or publish to a certain topic. When a client publish a message to a certain topic, each client who subscribed to this topic will receive the message. A MQTT broker runs on the base station. When a measurement is received via BLE, it is published to the relevant topic (e.g. measures/temperature). This step is realized inside the same Python program that receives the measurement via BLE. The PahoMQTT library allow us to connect and to subscribe to the broker.
The measurements are displayed on the screen of the base station (Fig. 19).

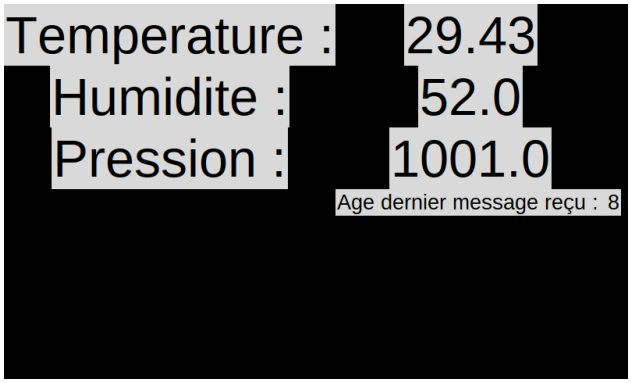

Fig. 19. Graphical interface

C. Screen

To display the measures values received from the BLE node, we have used a 15cm LCD screen. A python program using the Tkinter library is automatically started when the Raspberry is powered on. The program subscribe to the MQTT topics of temperature, humidity and pressure and update the display whenever a new message is received. The use of a screen is also interesting because it's an additional indicator of an abnormal behaviour of the system that can be easily seen.

D. Control Peltier module

We chose a Peltier module to simulate a heat source. The module is directly connected to an arduino board but the request to switch it on/off comes from the base station. A 433Mhz link allows a communication between the base station and the arduino. We developed two modes to control the Peltier module. The manual mode allows the user to switch on/off the module thanks to a switch which is connected to the GPIO ports of the base station. The automatic mode lets the system choose if the module needs to be switched on/off. The decision is made using a comparison between the temperature measurement and a threshold. From a technical point of view, another python program is used to provide this functionality.

1) 433Mhz communication: 433Mhz wireless communications is commonly used for IOT applications since it is cheap and energy-efficient. We thought it would be a good idea to implement this technology in our system. The 433Mhz link is realized thanks to the FS1000A emitter and the MX-RM-5V receiver. According to the data sheet, it is a communication based on ASK modulation and it offers a range up to 100 meters. The protocol we used is not reliable and offers no guarantee (no acknowledgment).

Fig. 20. FS1000A and MX-RM-5V

2) Arduino: As explained before, the purpose of the Arduino card is to switch on/off the Peltier module. Therefore, its programming was easy once we understood how to operate the 433 MHz modules. A code has been assigned to the on/off action and its reception results in the heating of the Peltier module or not. The decision making is located in the Raspeberry Pi, not in the arduino. We chose not to secure the 433 MHz communication because the libraries we used already contains some redundancies and to allow the possible use of vulnerabilities.

E. Monitor data remotely

The interesting thing to do when we receive data from the IoT node is to save it on a database that we can access from anywhere. To fit with the remote characteristic, we have implemented a web server to save and retrieve data. Thanks to http ://etud.insa-toulouse.fr, an INSA web server, we all have a personal space on which we can develop services accessible from the Internet. The interesting part is that we can access a database through etud. Hence, we have implemented the server with the database on one of our personal space at http ://etud.insa-toulouse.fr/s̃oussi.

There are two scripts running on http ://etud.insa-toulouse.fr/s̃oussi :

- The first one is called logs.php.
  This script is used to receive data from the base station in order to save it into the database. It receives GET requests with three parameters : temperature, pressure and humidity. It connects to the database and inserts the three measures on it.
- The second one is called traitements.php.
  This script is used to make requests to the database. As we want to get data on a given period of time, we just need to specify this period. Hence, the server handles a GET request with only one parameter, a time in DATETIME format. When it receives an http request, it performs an SQL query to get the maximum value stored in each row of the database (i.e temperature, pressure and humidity) for the last 15 minutes. It returns these values to the emitter of the request.

V. Attacks

We managed to realize different kinds of attacks to illustrate that IoT devices are vulnerable from multiple points.

A. Jamming

Jamming is a denial of service attack in which radio communication is disrupted by an attacker. It consists of emitting high power signals in a wide band covering the area of emission.

1) Jamming 433Mhz connection: We were able to jam 433Mhz leading to a malfunction of our system. The arduino did not receive the messages sent from the base station and so the Peltier module cannot be switched on or off. We scanned the 433Mhz frequency using Gqrx during a functional operation (top of the Fig. 21) and during an attack (bottom of the Fig. 21). The blue color represents low energy, the yellow/red colors represent high energy. We can see clearly that our signal is perceptible on the top figure while it is difficult to discern it on the bottom figure. The attacker manages to increase the noise level to -40dBm which it corresponds to our signal emission intensity. This attack can easily be done for any attacker without any particular knowledge. However, we cannot be far away from the object so that the noise level is high enough to interfere with the communication.

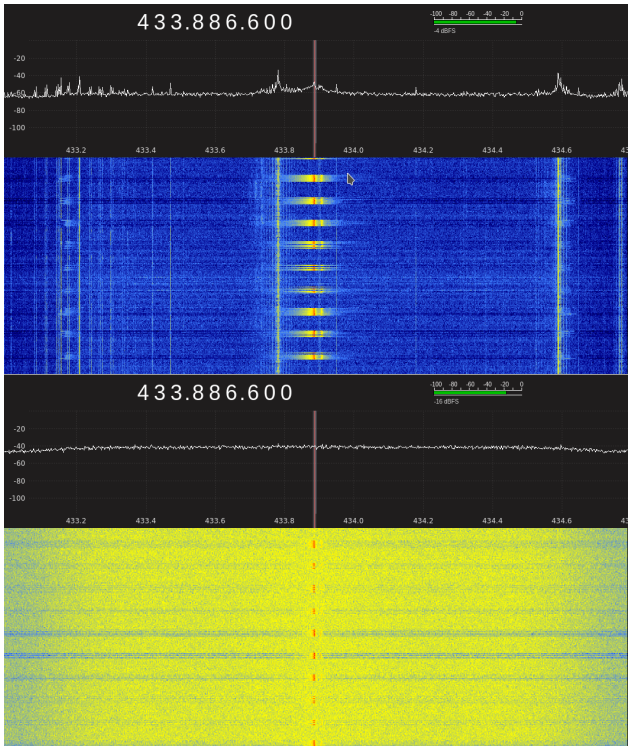

Fig. 21. Normal 433Mhz communication (top) - Jamed 433Mhz communication (bottom)

2) Reactive jamming - Wi-Fi: We have also implemented another type of Jamming. While our 433 Mhz is constantly emitting, we used another technique to jam the Wi-Fi signal : reactive jamming. It consists in waiting for the start of Wifi transmission. The jammer has to react quickly after this detection to emit its jamming radio signal. This has the advantage to allow the attacker to chose the target. Indeed he can start the jamming only if the firsts bits correspond to the target address. It also has the advantage of being less easily detectable.

B. Denial of service

1) MQTT Broker: A vulnerability was found in Eclipse Mosquitto 1.4.15 and earlier (referenced as CVE-2017-7654). An unauthenticated client can send a huge amount of connected packet leading to a denial of service in the broker. To perform this attack, we used a program written in C++ which runs multiple thread to connect to the MQTT broker. Unfortunately, we were not able to crash the broker with only one attacker. The attack would certainly be successful if we increase the number of attacker.

C. Bruteforce

Brute force attack is a method to gain access to anything protected by a password by trying large amount of combinations of usernames and passwords until the right one is found.

1) SSH Bruteforce: We were able to bruteforce the SSH protocol on the Raspberry Pi that allows a user to connect remotely to an host as administrator using username/password with Hydra, a brute-forcing tool available on Linux. The tool only requires a username list, a password list and the IP address of the device to bruteforce the SSH protocol sending multiple requests. We believe this kind of attack can be very interesting because the highly elevated number of ssh requests makes the detection of this kind of attack easier for anyone listening on the network.

2) HTTP Authentication Bruteforce: We were able to bruteforce HTTP Authentication which protects the web page that allows the user to choose the Wifi network he want to use,or that we used a python script that asks for input : the target url ,username list and password list. The output gives the correct user name and password.

D. Man in the Middle

1) Bluetooth Low Energy: We managed to perform attacks on Bluetooth Low Energy using the Gattacker tool. It is a Node.js package which allows several attacks such as eavesdrops, replay, man in the middle... The principle of the attack is describe on the Fig. 22. Two different entities (a master and a slave) are needed since this attack needs two independent Bluetooth interfaces. The slave gathers advertisement and services obtained from the targeted device (sensing device in our case). It is transmitted to the master which emits the same advertisement at a higher frequency. As a result, the other targeted device (the base station in our case) connects to the master device rather than the sensing device. The master gets all the requests of the base

station and forwards it to the slave which forwards it to the sensing device. The attacker is in a middle position and is offered many opportunities : he can listen the trafic, intercept and/or modify the transmitted requests and responses... The tool logs all the advertisements, discovered services and requests/responses in JSON files.

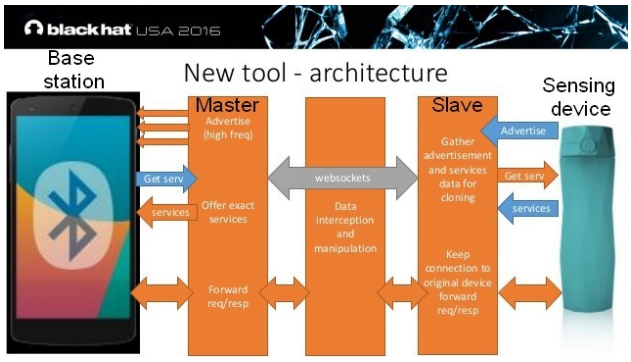

Fig. 22. BLE MITM architecture [39]

E. Wifi Deauthentification attack and evil twink

In order to see with which distant server the connected object is connected to and which information is exchanged we have set up an "evil twin" Wifi access point. Our access point has the same SSID and BSSID as the legitimate. The scenario of this attack consist in cloning this access point using a larger transmit power. Once the malicious access point is operating, we send a deauthentification frame to the access point, we spoof the source address using the MAC address of our object. It will then try to reconnect to the strongest signal : our malicious AP. We are now able to sniff all the traffic by redirecting it to Internet.

F. Wifi Deauthentification attack and capture of the handshake

For this attack we decided to use the classic aircrack-ng tool. With the command : airmon-ng we switch the network card in monitor mode to be able to listen to the nearest access points. Then using airodump-ng we start capturing data belonging to the selected station. At the same time, we have to also use the aireplay-ng command to send desauthentication requests to the station that will disconnect the connected devices. The disconnect devices will automatically try to reconnect to the station and doing so, allowing us to capture the WPA2 Handshake. Once the handshake is captured, we use aircrack-ng to bruteforce the handshake with a wordlist. For this step to be quick, the WPA2 password we have to find has to be very easy or we will directly add it to the wordlist.

G. HTTP breaches

When we have a simple GET request to make to add data to a database, we can easily insert anything we want to it if there is not enough security. On our system, we perform a request on the database to get the maximum value in each field for the last 15 minutes. The goal of the SQL injection is to gather information from a database that we should not be able to access. For us, the goal would be to get the maximum value of temperature, humidity and pressure since the creation of the database. It is not critical data, but this could have an important confidentiality impact : private data can be retrieved and exploited for example to find out whether or not the housing is heated.

1) Connection to the AP: The first thing to do is to connect to the access point that the object is connected to. In order to do that we need to get the WPA2 key to connect to the station. We can use one of the previous method we explained to get the WPA2 key.

2) Network scanning - ARP Spoof: Once we are connected, we need to get the network address of the default gateway. That will be the same network than the one used by the object to connect to the Internet.
Then we use the arpspoof command to force all the network traffic to go through us. For that purpose, we need to enable the ip forwarding.

There may be a lot of devices connected to the AP and we will not try to analyse all the traffic going through. The device we want to find has an operating system. To identify it we use the nmap -O command. It returns the OS found, along with the version and information about the ports. In the IoT context, the most widely used small processor is a Raspberry, constructed by ASRock. Based on this information, we can easily get the address of the device we want to spy on. Hence, it is easy to launch a tcpdump command to analyse all its traffic.

What we want, is getting the ip addresses used to reach an http server. For that, we need to isolate the traffic destined to port 80 or 8080.

```
[root@mwiws01 ~]# tcpdump -i enp0s8 -s 0 -A 'tcp dst port 80 and tcp[((tcp[12:1]
 & 0xf0) >> 2):4] = 0x47455420 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x48545450
and host 192.168.10.1 '
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), capture size 262144 bytes
07:25:11.134406 IP 192.168.10.1.58920 > mwiws01.http: Flags [P.], seq 3703671002
:3703671164, ack 1048287386, win 4117, options [nop,nop,TS val 396578671 ecr 384
6984], length 162: HTTP: GET /new.html HTTP/1.1
E...$.@.@.....
...

.(.P....>{.............
..Oo.:.HGET /new.html HTTP/1.1
Host: 192.168.10.10
User-Agent: curl/7.54.0
Accept: */*
X-Requested-By: middlewareinventory
TestHeader: TestValue
MyName: SaravAK


07:25:11.136818 IP mwiws01.http > 192.168.10.1.58920: Flags [P.], seq 1:294, ack
 162, win 235, options [nop,nop,TS val 3846987 ecr 396578671], length 293: HTTP:
 HTTP/1.1 200 OK
E..Y+.@.@.xm..

..
..P.(>{.....|...........
.:.K..OoHTTP/1.1 200 OK
Date: Thu, 26 Jul 2018 07:25:11 GMT
Server: Apache/2.4.6 (CentOS)
Last-Modified: Sun, 08 Jul 2018 21:56:05 GMT
ETag: "43-57083f6ab4740"
Accept-Ranges: bytes
Content-Length: 67
Content-Type: text/html

<html>
<body>
<h2>www.middlewareinventory.com</h2>
</body>
</html>
```

Fig. 23. Tcpdump trace of GET and POST requests

On Fig.23 , we can see an example of a tcpdump trace. It

is done to capture traffic on port 80 with 'tcp dst port 80'. It also displays only GET and POST messages through the '0x47455420' and '0x504F5354' codes respectively.

The interesting part is that we can see the HTTP request that is made to reach the server. As our requests are not made through HTTPS but rather HTTP, there isn't any encryption. This is a huge breach of the system's confidentiality.

Now that we have the HTTP request, we know the address of the distant server, which script is used and which parameters the request uses. Then it is easy to access the server on our own.

3) Gathering data from the server: One of the option we have on the system is to try to gather more data than the original request. We use on the base station a get request to access the server, entering the time as a parameter. From the attacker point of view, we have sent HTTP requests at the addresses found before, adding "+OR+1=1" at the end : 'https ://etud.insa-toulouse.fr/s̃oussi/traitements.php ?time="00 :15 :00" +OR+1=1'. It will be interpreted as : the original request of maximum value for the last 15 minutes, or true, which leads to looking for the maximum value since the creation of the database.

4) Adding wrong data to the database: What can also be done, is generating a good looking HTTP request with 3 arguments. We manually chose the values we want to add to the database. We can even try to add letters or dates instead of the numbers we have seen on the legitimate HTTP request. The web server will interpret it as a request from the base station and add the bad data to the database.

H. Others attacks

1) Publish/Subscribe to MQTT without credentials: In order to publish or subscribe to the broker MQTT, a client needs to use correct credentials. The broker uses an ACL and a file where all the encrypted passwords are stored. A vulnerability was discovered (CVE-2018-12551), and allows a user to publish/subscribe without knowing credentials. He just only needs to use a malformed username like a blank chain. This vulnerability comes from the way that the file containing username/password is parsed. In our case, an attacker can publish bad measurements to damage the system (e.g. switch on the Peltier Module while the temperature is above the threshold).

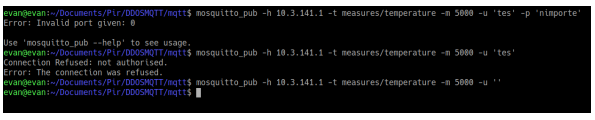

Fig. 24. First two attempts failed while the last attempt (blank username) worked

I. Scripting the behaviour

Since the object will be used by another group to train their detection box, we decided to develop a script on the the raspberry Pi which turns our device into a 'normal' mode. The raspberry collects measurements from the IoT node in Bluetooth BLE and processes them. If the automatic mode was chosen, it activates the Peltier module using 430MHz communication according to the received measurements. Otherwise, if the manual mode was chosen, the Peltier module is activated by pushing a button.
Furthermore, we managed to collect a reasonable amount of attacks. We can include them in larger scripts that will combine different kind of attacks to get to the connected object. Some attacks are performed just by starting a command and are able to be combined by calling them one after another. We managed to develop a more complex attack scenario which combines multiple attacks that depends on each other. This scenario uses the "WiFi Deauthentification attack and capture of the handshake" method explained before to get the WPA passphrase of the raspberry (the object), then connects to its network and bruteforces the SSH and the HTTP protocols running on the card with the method explained in the bruteforce section.

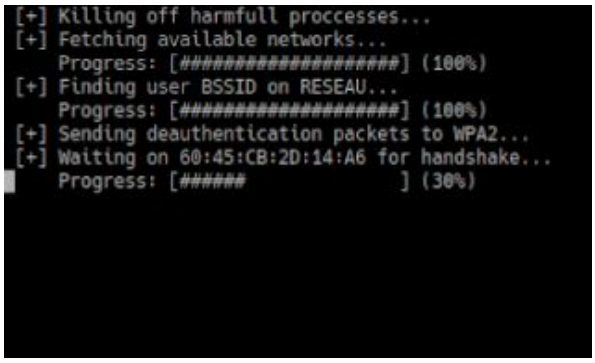

Fig. 25. Script's execution

On figure 25, there is a view of our script's execution. Each step is separated and can be followed thank is to the progress bar.

VI. Conclusion

The aim of the project was to design a typical device which corresponds to the current products available on the market and that is why we chose a smart thermostat. We used technologies that are commonly used in IOT such as Bluetooth Low Energy, Wi-Fi, MQTT, 433Mhz... By initially studying the state of the art, we were able to grasp the main vulnerabilities of IOT devices. Thanks to this part, we searched, found, analysed and exploited vulnerabilities in our built system. The main result of this paper is that vulnerabilities are omnipresent in IOT and their exploitation can lead to potentially serious damages. That is why our project is made along with another project whose purpose is to automatically detect that an attack is occurring on an IoT device by monitoring the network flow and the physical characteristics such as temperature or light.

What could be done to continue this project, is developing an industrial detection system. It would involve a test box with more parameters than what have been done in this project, and a large amount of scripts to attack the system in all the ways that exist. That would enable companies to test their systems before they are marketed. That wouldn't exclude their own security tests but rather complete it.

From a pedagogical point of view, this project allowed us to learn more about connected objects, security and to develop soft skills such as group work, curiosity, personal development and autonomy.

References

[1] Z. Ling, K. Liu, Y. Xu, C. Gao, Y. Jin, C. Zou, X. Fu, and W. Zhao, "IoT security : An end-to-end view and case study." [Online]. Available : http://arxiv.org/abs/1805.05853

[2] S. Sowmya and P. D. S. K. Malarchelvi, "A survey of jamming attack prevention techniques in wireless networks," in *International Conference on Information Communication and Embedded Systems (ICICES2014)*. IEEE, pp. 1–4. [Online]. Available : http://ieeexplore.ieee.org/document/7033757/

[3] M. Pirretti, S. Zhu, N. Vijaykrishnan, P. McDaniel, M. Kandemir, and R. Brooks, "The sleep deprivation attack in sensor networks : Analysis and methods of defense," vol. 2, no. 3, pp. 267–287. [Online]. Available : http://journals.sagepub.com/doi/10.1080/15501320600642718

[4] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *Proceedings of the 7th annual international conference on Mobile computing and networking - MobiCom '01*. ACM Press, pp. 272–287. [Online]. Available : http://portal.acm.org/citation.cfm?doid=381677.381703

[5] J. Deogirikar and A. Vidhate, "Security attacks in IoT : A survey," in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. IEEE, pp. 32–37. [Online]. Available : http://ieeexplore.ieee.org/document/8058363/

[6] D. D. Virmani, A. Soni, S. Chandel, and M. Hemrajani, "Routing attacks in wireless sensor networks : A survey."

[7] P. Ramesh, D. Bhaskari, and C. , "A comprehensive analysis of spoofing," vol. 1, no. 6. [Online]. Available : http://thesai.org/Publications/ViewPaper?Volume=1&Issue=6&Code=IJACSA&SerialNo=23

[8] M. Conti, N. Dragoni, and V. Lesyk, "A survey of man in the middle attacks," vol. 18, no. 3, pp. 2027–2051. [Online]. Available : http://ieeexplore.ieee.org/document/7442758/

[9] What is code injection ? - definition from techopedia. [Online]. Available : https://www.techopedia.com/definition/20022/code-injection

[10] M. Vuagnoux and S. Pasini, "Compromising electromagnetic emanations of wired and wireless keyboards," p. 16.

[11] A. Fouskas, *Automated discovery and binding of IoT devices*, no. Bachelorarbeit, published : Bachelorarbeit : Universität Stuttgart, Institut für Parallele und Verteilte Systeme, Anwendersoftware. [Online]. Available : http://www2.informatik.uni-stuttgart.de/cgi-bin/NCSTRL/NCSTRL_view.pl?id=BCLR-2017-24&engl=0

[12] BlueBorne information from the research team - armis labs. [Online]. Available : https://armis.com/blueborne/

[13] CVE-2017-1000251 : The native bluetooth stack in the linux kernel (BlueZ), starting at the linux kernel version 2.6.32 and up to and includ. [Online]. Available : https://www.cvedetails.com/cve/CVE-2017-1000251/

[14] O. Nakhila and C. Zou, "User-side wi-fi evil twin attack detection using random wireless channel monitoring," in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, pp. 1243–1248.

[15] OWASP internet of things project - OWASP. [Online]. Available : https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=OWASP_Internet_of_Things_Top_10_for_2014

[16] The internet of things (IoT) – threats and countermeasures. [Online]. Available : https://www.cso.com.au/article/575407/internet-things-iot-threats-countermeasures/

[17] Many vulnerabilities among internet of things (IoT) devices to hacking,originate from insecure web or web-based interfaces, according to ilia kolochenko, CEO of web security company high-tech bridge. [Online]. Available : https://www.ifsecglobal.com/global/insecure-web-interfaces-achilles-heel-iot-security-defences-cybersecurity-ceo/

[18] Joe. What is account enumeration ? [Online]. Available : https://affinity-it-security.com/what-is-account-enumeration/

[19] Does two factor authentication actually weaken security ? [Online]. Available : https://paul.reviews/does-two-factor-authentication-actually-weaken-security/

[20] Half the web is now encrypted. that makes everyone safer | WIRED. [Online]. Available : https://www.wired.com/2017/01/half-web-now-encrypted-makes-everyone-safer/

[21] Z. Kerravala. What is transport layer security (TLS) ? [Online]. Available : https://www.networkworld.com/article/2303073/lan-wan/lan-wan-what-is-transport-layer-security-protocol.html

[22] About TLS/SSL. [Online]. Available : https://docs.apigee.com/api-platform/system-administration/about-ssl

[23] How to connect IoT device to the internet. [Online]. Available : cactus.io/tutorials/ethernet/connect-iot-device-to-the-internet

[24] W. Zhou, Y. Jia, Y. Yao, L. Zhu, L. Guan, Y. Mao, P. Liu, and Y. Zhang, "Phantom device attack : Uncovering the security implications of the interactions among devices, IoT cloud, and mobile apps." [Online]. Available : http://arxiv.org/abs/1811.03241

[25] M. A. Hakim, H. Aksu, A. S. Uluagac, and K. Akkaya, "U-PoT : A honeypot framework for UPnP-based IoT devices." [Online]. Available : http://arxiv.org/abs/1812.05558

[26] H. Moore, "Security flaws in universal plug and play. unplug. don't play." [Online]. Available : https://hdm.io/writing/SecurityFlawsUPnP.pdf

[27] S. Kalra and S. K. Sood, "Secure authentication scheme for IoT and cloud servers," vol. 24, pp. 210–223. [Online]. Available : https://linkinghub.elsevier.com/retrieve/pii/S1574119215001510

[28] M. b. Mohamad Noor and W. H. Hassan, "Current research on internet of things (IoT) security : A survey," vol. 148, pp. 283–294. [Online]. Available : https://linkinghub.elsevier.com/retrieve/pii/S1389128618307035

[29] "IoT security : An end-to-end view and case study."

[30] C.-H. Liao, H.-H. Shuai, and L.-C. Wang, "Eavesdropping prevention for heterogeneous internet of things systems," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, pp. 1–2. [Online]. Available : http://ieeexplore.ieee.org/document/8319297/

[31] M. S. Henriques and N. K. Vernekar, "Using symmetric and asymmetric cryptography to secure communication between devices in IoT," in *2017 International Conference on IoT and Application (ICIOT)*. IEEE, pp. 1–4. [Online]. Available : http://ieeexplore.ieee.org/document/8073643/

[32] C. Li, Z. Qin, E. Novak, and Q. Li, "Securing SDN infrastructure of IoT–fog networks from MitM attacks," vol. 4, no. 5, pp. 1156–1164. [Online]. Available : http://ieeexplore.ieee.org/document/7883928/

[33] IoT technology stack - IoT devices, sensors, gateways and platforms. [Online]. Available : https://www.i-scoop.eu/internet-of-things-guide/iot-technology-stack-devices-gateways-platforms/?fbclid=IwAR2SjgqJFdCJu5gxWsOYQW3P-LCoXVTAr1gUKBpG-WgmagJ9Qz5_NWLUD3Q

[34] M. Goryachy and M. Ermolov, "Obtaining full system access via USB," p. 4. [Online]. Available : https://www.ptsecurity.com/ww-en/analytics/where-theres-a-jtag-theres-a-way/

[35] MQTT and CoAP : Security and privacy issues in IoT and IIoT communication protocols - security news - trend micro USA. [Online]. Available : https://www.trendmicro.com/vinfo/us/security/news/internet-of-things/mqtt-and-coap-security-and-privacy-issues-in-iot-and-iiot-communication-protoc fbclid=IwAR0_TLI5igNdeIxdy6lsj-0rcWwUyjl3qM1es_L2lPwdX3vc7RzOUw0awZM

[36] MQTT. [Online]. Available : https://mqtt.org/

[37] NVD-CVE-2017-7653. [Online]. Available : https://nvd.nist.gov/vuln/detail/CVE-2017-7653?fbclid=IwAR3VBPTLyYM2RdYizgtrurO3mArRDFED2JB_I92_VR53Gc5LfoIkS3KCwnE

[38] GATT | Introduction to Bluetooth Low Energy | Adafruit Learning System. [Online]. Available : https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt

[39] Black Hat, "Gattacking Bluetooth Smart Devices - Introducing a New BLE Proxy Tool." [Online]. Available : https://www.youtube.com/watch?v=uKqdb4lF0XU