Security Analysis of the Internet of Medical Things (IoMT): Case Study of the Pacemaker Ecosystem*

Guillaume Bour[1][0000−0003−4456−6279], Anniken Wium Lie, Jakob Stenersen Kok, Bendik Markussen, Marie Elisabeth Gaup Moe[2][0000−0003−1786−1133], and Ravishankar Borgaonkar[1][0000−0003−2874−3650]

[1] SINTEF Digital, Strindvegen 4, Trondheim, Norway
{guillaume.bour, ravi.borgaonkar}@sintef.no
[2] Norwegian University of Science and Technology, Trondheim, Norway
marie.moe@ntnu.no

Abstract. During the pandemic, the Internet of Medical Things (IoMT) has played a key role in reducing unnecessary hospital visits and the burden on health care systems by providing home-based hospital services and ambulatory nursing services. As IoMT devices handle patient data and are connected over the Internet to the complex hospital Information and Communication Technology (ICT) infrastructure, their role in the transformation of healthcare services will introduce a range of new potential risks. Over the past years, several demonstrated attacks in the healthcare domain have indicated cyber security challenges for integrating IoMT devices.

In this paper, we experimentally evaluate the potential risks that accompany the integration of a given IoMT device, here a connected pacemaker, from a hardware and network security perspective. We take a black box testing approach to the pacemaker ecosystem and find key shortcomings that enable several practical and low-cost attacks that impact a patient's safety and privacy. In particular, we demonstrate the ability to gain control over the home monitoring device and to perform man-in-the-middle attacks. We find that it is possible to bypass hardware security protection mechanisms, to perform remote denial of service attacks, and other attacks. Lastly, we discuss the potential trade-offs in security protection choices and mitigation techniques.

Keywords: IoMT · IoT Security · Pacemaker · Medical Device · Cyber Security

1 Introduction

Hospital-at-Home is expected to become the new norm in the coming decade. Technology has made major changes to the way healthcare is provided, and

* This work was funded by Reinforcing the Health Data Infrastructure in Mobility and Assurance through Data Democratization, a five-year project (grant number 28885) under the Norwegian IKTPLUSS-IKT and Digital Innovation programme.

thanks to the fourth industrial revolution, humans will live longer and healthier lives. Consultations will take place over video, and IoMT sensors will take care of monitoring vital in real time, issuing alerts if anything appears abnormal. More intrusive devices such as Implantable Medical Devices (IMD) will also become increasingly connected.

Connecting patients' homes means expanding the attack surface of the system. Patients' safety has always been a key priority in medical devices. Pacemakers, for instance, are built with "fail-safe" modes which they will switch to in case something goes wrong with their programming; this keeps the pacemaker generating a constant pulse until a pacemaker technician can re-progam the device for the patient. Cyber security of medical devices, on the other hand, has not been paid much attention by researchers, nor has it been publicly debated until the last decade. The healthcare domain is, however, not spared by cyber criminals, and attacks like the WannaCry ransomware that struck the world in May 2017 have shown that hospitals and medical devices are at risk for being infected via collateral damage even if the attack was not specifically targeted towards them. A cyber attack can have a real impact on human lives. A poor implementation of cyber security in the edge systems of the Home Hospital could have catastrophic consequences not only for the patients' privacy but also for their safety. Having secure connected medical devices is thus a must-have in order to pave the way to future home-care. Connected pacemakers constitute some of the most life critical medical devices and were among the first to be "connected". As such, they can provide us with evidence of the evolution of cyber security in medical devices over time.

Connected IMD in the form of modern pacemakers are not a new medical innovation but the evolution of technology from the fifties and sixties. In the seventies "on-demand" pacemakers were developed that would sense the patient's cardiac activity and adjust the pacing accordingly. These pacemakers could be remotely programmed through a radio-frequency telemetry link. The first pacemakers driven by microprocessors appeared in the nineties. These devices were able to detect cardiac events and could adapt their internal pacing based on the patient's needs. The first *connected* pacemakers appeared in the early 2000s, with the addition of an external device that would connect wirelessly to the pacemaker and upload its data to a remote server via the Internet, thus reducing the need for patients to go to the clinic for a check-up. Today, this remote connectivity is becoming more and more popular in use. An external device, sometimes called a "bedside monitor", which we in this paper will refer to as the Home Monitoring Unit (HMU), is used to gather the pacemaker's data and upload it to a remote server accessible to the clinician through a web interface.

Over the past three years, we have been analyzing the security of the pacemaker ecosystem of one of the main vendors on the market today. We looked at three different generations of HMU devices and compared their security to document the state-of-the-art and to see how security implementation in these devices has evolved over time.

Our main results, presented in a previous paper [3], suggest that even if the overall security of the devices has improved, the medical device manufacturers are still lagging behind and have failed to implement common security practices. In this paper, we highlight the methodologies used to test the devices from both a hardware and a network perspective. In particular, we present new results obtained through the fuzzing of the generic modem used by the pacemaker HMU. We have implemented a fuzzing framework using a GSM network test bed. The equipment (software defined radio) needed for fuzzing infrastructure is inexpensive and readily available. We followed ethical response procedures and reported our attacks based on identified vulnerabilities to the concerned manufacturers.

The paper is organized as follows. Section 2 provides the background of our work, including a description of the principle of the pacemaker and its ecosystem, along with the interactions between its different components. Here we also review the relevant related work and the threat model used in our research. In Section 3 we outline the methodology used along with the different setups used to perform the security analysis. Section 4 presents our main findings,from a hardware, firmware, communication and infrastructure perspective. Those findings are not pacemaker-specific and can be considered for all connected medical devices that present similar architecture. Potential attack scenarios are also presented. Section 5 provides a discussion of the results along with mitigations. Section 6 concludes the paper.

2 Background

2.1 The pacemaker ecosystem

Pacemakers and Implantable Cardioverter Defibrillators (ICD) are active implantable medical devices, which are defined in the Norwegian regulatory framework [15] as "*Any active medical device which is intended to be totally or partially introduced, surgically or medically, into the human body or by medical intervention into a natural orifice, and which is intended to remain after the procedure.*" Both pacemakers and ICDs are battery-powered devices surgically implanted in a patient to treat a heart related condition. They differ in the conditions they are treating, as ICDs are not only capable of continuous monitoring the heart rhythm and pacing the heart with electrical pulses, but also of delivering an electrical shock to the heart if required. In this paper, because both ICDs and pacemakers are similar devices from the cyber security point of view, they will both be referred to as pacemakers.

Pacemakers are constructed to last for around 10 years varying on their usage, before having to be replaced due to the battery running out. The devices are able to deliver pacing when required and in a way that is adapted to each patient. This means that the clinician needs a way to program the device in a non-invasive way for the patient. As previously mentioned, an RF-telemetry link was introduced to the devices in the 70s to program some parameters in the pacemaker. Since then, pacemakers have evolved into complex embedded devices, driven by a microcontroller. Once implanted, they are not standalone

devices which are left there for ten years waiting to be replaced, but take place in an ecosystem that allows for monitoring the devices and also the patients' condition.

The Home Monitoring Unit is an example of a communication device that enables this remote regular monitoring by connecting wirelessly to the pacemaker, reading and transmitting data from it. Figure 1 presents the pacemaker ecosystem.

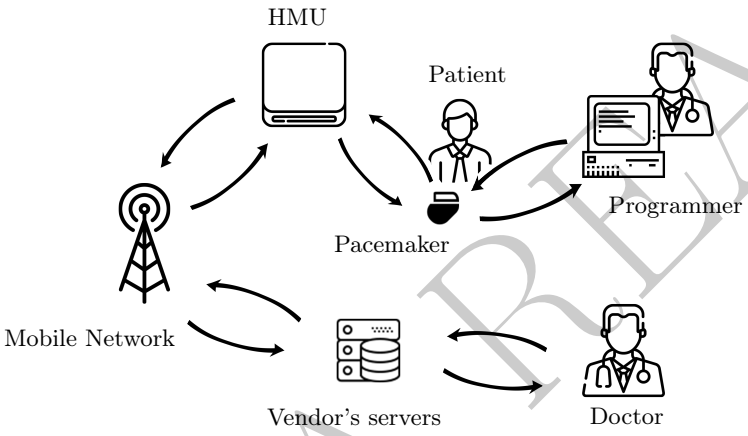

Fig. 1. Diagram of the vendor's pacemaker ecosystem [4]

The pacemaker Implanted in the patient's body, this is the main device of the ecosystem. As already explained, it generates an electric impulse that helps regulate the heart rhythm.

The programmer The programmer is an external computer used by a clinician to program the pacemaker. Programming the pacemaker is achieved wirelessly by placing the programming head of the programmer in close proximity of the pacemaker. While old pacemakers used to communicate with the programmer over the 175 kHz band, newer ones tend to use 402-405 MHz Medical Implant Communications (MICS) band [25]. The communication of the pacemaker with the programmer is triggered by applying a magnetic field on the implant, causing a magnetic switch inside it to close [12]. This magnetic field is emitted by the programming head. It should be noted that pacemakers from different vendors require different programmers due to differences in communication protocols, and that a programmer of a specific vendor usually supports several pacemakers/ICD devices from the same vendor.

The HMU The HMU is a router-like device in charge of collecting telemetry data from the implant and transmitting it. The device is paired with a pacemaker, placed in the patient's home and receives the data sent by the pacemaker

at a pre-configured time (for instance every night at 2:00). The HMU also communicates with the pacemaker over the 402-405 MICS band, which allows for longer range communications than the 175 kHz band. This data is then sent to a backend server, usually owned by the pacemaker manufacturers. Similar to the programmer, pacemakers from different manufacturers require different HMUs. Some newer pacemakers communicate over Bluetooth Low Energy with an app installed on the patient's smartphone, eliminating the need for an external HMU device.

The operator's network In order to transmit the data to the backend server, the HMU needs connectivity. To achieve that without having to rely on patients' internet connection and also for ease of use, manufacturers usually have contracts with Telecom operators. That way, the HMUs are shipped with a SIM card to access the Global System for Mobile Communication (GSM) or 3G networks, or with access to the internet through telephone lines for older versions. The HMU either connects directly to the server which is exposed on the public Internet or connects to a Virtual Private Mobile Network (VPMN) which gives it access to the server. This implementation varies with vendors.

Vendor's backend infrastructure This infrastructure is used to receive data sent by the HMU, process it and make it accessible to the clinician through an online platform. Alerts may also be triggered if something looks irregular, for instance, if no data has been uploaded in a while for a given HMU, or if there is a problem with the patient's condition. This allows the clinician to call in the patient for a follow-up checkup if necessary.

2.2 Related Research

While wireless communication technology has been a feature of pacemakers since the seventies, security researchers have only been taking an interest in this topic for around 15 years. In 2008, Halperin et al. published the first research paper describing a security attack against a commercial pacemaker [12]. Their research targeted the communication between the pacemaker and its programmer. Using Software Defined Radio (SDR), they partially reversed engineered the communication protocol in use and, with that knowledge, were able not only to eavesdrop and decode the communication, but also to perform data replay attacks. They were able to interrogate the pacemaker to reveal the patient's data containing personal information such as patient's name, diagnosis, etc. They were also able to change parameters of the pacemaker, such as the patient's name, implantation date, or even therapies (that includes turning off all therapies). Finally, and more frightening, they were able to trigger a shock on the ICD, which could have fatal consequences on a real patient if delivered at an inappropriate time. They thus highlighted the severity of the lack of security mechanisms for implantable medical devices.

In more recent research from 2016, Marin et al. carried out similar research on the latest generation of pacemakers [19]. Their research highlights several weaknesses in the communication protocol and shows that a weak adversary can perform attacks even with low capabilities. Three kinds of attacks were performed.

First, the researchers managed to access private patient information from the telemetry information, even though some obfuscation technique has been done by the manufacturer. Secondly, they performed Denial-of-Service attacks. By keeping the device in "interrogation" mode, they were able to send messages to the device over a long-range communication channel and thus drastically reduce the implant battery life. Finally, they found that there is no mechanism against replay attacks and that an adversary without any knowledge of the protocol could simply replay captured messages and spoof the programmer.

A report exposing vulnerabilities in the pacemakers and HMUs manufactured by St. Jude (now Abbot) was published by Muddy Waters Capital LLC in 2016 [2]. Amongst the vulnerabilities that were presented was a way to perform a battery-draining attack on the pacemakers or forcing them to pace at a rhythm that would be potentially fatal for the patient. These attacks were carried out by first compromising the HMU, which was then used to attack the pacemaker. Even if no attack has been publicly reported exploiting these vulnerabilities, the disclosure of this report had a potentially severe impact on the 260 000 HMUs deployed in patients' homes at the time. As a result, the vendor issued a firmware update at the beginning of 2017 to mitigate the vulnerabilities.

In 2017, Rios and Butts evaluated the security of the pacemaker ecosystems of the four major vendors [23]. They presented several weaknesses, in the programmers, the pacemaker implants, and the HMUs. Weaknesses include vulnerable third-party software, lack of authentication between devices, unencrypted filesystems and firmware, removable hard-drives, and unsigned firmware. The conclusion is that the whole industry is quite immature in terms of cyber security. They highlight that this is not only the case for one unique vendor but that all vendors are impacted.

Attacking the mobile network interface of embedded device is not new either: in 2011 Mulliner et al. published their research on the possibility of launching large scale attacks against phones by exploiting vulnerabilities in the Short Message Service (SMS)-client software [20,11]. In particular, they developed a platform-agnostic testing framework which allows fuzzing SMS messages over a software-based GSM network. They concluded that an attacker can send malformed SMSs resulting in the receiver's phone being disconnected from the network forcing the user to reboot it. Similar work was conducted by Weinmann who focused on the whole cellular basebands from different manufacturers rather than on the SMS-interface only [27]. They combined Mulliner et al.'s approach with reverse engineering of different stacks to identify flaws. In today's devices, the baseband's stacks and applications commonly run on a separate co-processor or chip with their own operating system and memory. Despite this separation, their work demonstrated the ability for an attacker to fully compromise a device by compromising the baseband's processor.

2.3 Threat Model

In this paper, we aim at understanding the evolution of the security measures in the pacemaker ecosystem and to evaluate its current maturity when it comes to cyber security. In our research, we have considered two classes of adversaries:

With physical access to an HMU It is possible to buy HMU devices online, sometimes at the low price range of $20 - $50. Since these are much easier to obtain compared to a pacemaker or a pacemaker programmer, one can afford to experiment with them without the fear of breaking an expensive device.
Capable of setting up a Fake Base Station (FBS) Such an attacker has access to Software Defined Radio equipment, which is also affordable. The HackRF One[3] manufactured by Great Scott Gadgets costs around $350. In order to perform fuzzing of the devices' modems, an attacker might also want to acquire the modems separately as this will allow for easier debugging (for instance by using it together with a RaspberryPi instead of the real device).

The two main facets that we want to look at in this research are the safety and privacy of the patient. To do this, we study the impact of different attacks on the patient's treatment and how it could be interfered with, directly or indirectly. Home monitoring has proven to save lives and as such, any attacks allowing an attacker to disrupt the service is also considered [10]. Regarding the patient's privacy, we look at what attacks would enable an attacker to access any kind of private data about the patient.

As mentioned in the introduction of this paper, motivations to attack the pacemaker ecosystem vary. Attacks against the patient's privacy are mostly driven by financial motives, in order to monetize the medical data on the black market. These attacks can have a great impact if they can be leveraged on a large scale. Safety related attacks could also be motivated by financial profit, for example we can imagine that an adversary could leverage a vulnerability in an extortion attempt by threatening a patient or maybe even a medical device manufacturer asking for a ransom. Targeted attacks against a single individual in order to harm or kill are less likely, except if it is a person of high interest. In both cases, one can imagine that we are facing organized crime or a nation-state threat actor. However, we cannot exclude single opportunistic attackers.

3 Methodology

3.1 Targets

Medical devices either speak to a gateway (the HMU in the case of the pacemaker) or are powerful enough to communicate with a network infrastructure directly. As such, the HMU is a good case study for security testing, and this is why we decided to look more into it. More specifically, we focused on two main

[3] See https://greatscottgadgets.com/hackrf/

attack vectors: *physical* and *network*. In both cases, a black box testing methodology was followed, as the tested components were proprietary hardware and software of which we had very little knowledge. In order to be as close to a real-world scenario as possible, we used commercial off-the-shelf (COTS) equipment whenever possible, and tried to keep the cost of an attack as low as possible.

The targeted devices from our lab were acquired second hand and are all BIOTRONIK's devices. This manufacturer was chosen because no prior security research had been published for this particular pacemaker brand, and no known vulnerabilities had previously been disclosed for its devices. Devices in our research project include three different generations of the HMU:

V1 : From 2000 to 2010, one of the first HMUs on the market, using the GSM network for connectivity. This version is not used anymore.
V2 : From the 2010s, in two models: one using the standard land line service for its connectivity, the other one using the GSM network. This version is not commercially available anymore, but is still in use.
V3 : From 2016, using the 3G network for connectivity. This is the current commercially available version. It is worth noting this version is "mobile" and thus not necessarily only accessible in the patient's home.

3.2 Black Box Testing

Definition The Black Box Testing methodology is a way to assess a software, a device or more generally a system from the outside while having very little knowledge about its internals. The attacker analyzes the outputs of the box obtained by sending some inputs or just by passively listening, and then tries to deduce the internals of the target. Having made some guesses, the attacker can adjust her inputs to confirm her thoughts or to exploit the target (see Figure 2).

Fig. 2. High Level Diagram of the Black Box Testing Methodology [4]

This methodology has several advantages compared to others that can be used to assess a system. Indeed, its primary objective is to test a system under real conditions, to emulate a real attack scenario. This means that such a test might catch errors made during the deployment of the system such as default passwords, misconfigurations in general or even the lack of security trainings of operators (weak passwords). This methodology also presents a low false positive ratio as the security expert can assess the risks associated with a vulnerability directly, i.e., if the vulnerability can be exploited or not. Black Box Testing also

has some drawbacks. By definition, the attacker has very little information about the target and might miss some vulnerabilities that would have been detected by code and/or configuration review. Despite this, it remains an excellent way to assess how a system stands against attacks and to get an idea of the path an attacker would take to compromise the solution, thus giving indications on how to tackle potential low hanging fruits. It can be later be completed by a deeper assessment following a grey or white box approach if applicable.

Ordinarily, the ecosystems for connected medical devices are for the most part proprietary. They usually rely on commercially available components and communicate using standardized frequency bands (such as MICS) but the rest (communication protocol with the gateway or with the backend infrastructure) is custom-made. The Black Box Testing methodology thus fits well not only for our research on the pacemaker ecosystem, but for security research on any connected medical devices. This is valid for the three approaches presented in the next subsections.

3.3 Hardware Testing Methodology & Setup

Methodology Our process can be split into five different tasks (see Figure 3). The first is the *Hardware Analysis*. Once we acquire a device, we start analyzing its components, which means knowing what the exposed interfaces are, debug interfaces, but also the chips that are on the board. To know that, opening the device to access the Printed Circuit Board (PCB) and analyze it is often required.

Knowing the components and available interfaces, we can then start looking for *documentation* such as datasheets, Request for Comments (RFC) or any other relevant information about the device. The goal of that second step is to understand the overall system and come up with some first hypotheses about it.

From those hypotheses, we then come with *testing* scenarios to be performed on the device and that will have two possible outcomes: either a success or a failure. However, the success or the failure of a testing scenario is determined by the expected result, which means in reality that even failure brings us information about the system.

Indeed, the results of a specific scenario need to be interpreted. This interpretation will be called a *finding*. Those findings are then used to look for new documentation and/or infer new hypotheses about the device. For example, if the hypothesis is "the device has debug ports exposed on the Universal Asynchronous Receiver Transmitter (UART) pins and is providing the attacker with a shell when connected to it", then the testing of that particular scenario will lead to either a confirmation or a rejection of the hypothesis. The interpretation is here quite easy as it is the hypothesis itself. This finding can then be reinjected in the documentation phase to infer new testing hypotheses like "the attacker is given a root shell when connecting to the UART console" or "the attacker can access the filesystem when connecting to the UART console".

Those findings are finally gathered to be *reported*. The reporting step is the one where the device is considered back in its whole ecosystem. That means the findings are interpreted again, but then with regard to different metrics. In the case of the above example, one can wonder what is the impact of the attacker having access to the filesystem on a medical device. The interpretation will be different when linked to other findings such as "The users' data are stored in cleartext on the device" or "The users' data are not stored on the device". Mitigation measures are also included in the reporting step.

Those steps can be mapped with the Open Source Security Testing Methodology Manual (OSSTMM) which is widely used to assess ITs' systems' security [14]. Indeed, the first two steps (*hardware analysis* and *documentation*) correspond to the *information gathering* (or approach) phase in the OSSTMM. The *contact* phase is then used followed by the *exploitation* phase, which are here mapped with the testing and findings phases. In the OSSTMM, the information gathered during the first phase along with the information gathered directly by the *contact* phase is then use to exploit the system and gain access. In our process, the information required to exploit, and gain access comes from previous testings. Finally comes the *reporting phase*, including mitigation. In the OSSTMM, one more phase is sometimes used depending on the engagement: the persistence one. In our case, persistence is studied as a hypothesis which is then tested and reported as any other findings.

Our methodology also follows the OWASP Firmware Security Testing Methodology [1]. While this methodology is more targeted towards more complex systems (running a Linux-based Operating System for instance), it remains applicable for simpler firmware. The hardware testing presented in this paper covers all 9 stages to some extent (considering that there are no file systems in the extracted firmware).

Testing Setup We used a combination of several COTS equipment to perform hardware testing on the devices. This equipment varied, depending on the stage we were at. *The shikra* was used to interface with low-level data interfaces via USB (it proved to be more reliable than a standard USB-to-TTL adapter). We used a *logic analyzer* to detect non standard baud rates and the *JTAGulator* to identify the pinout of the JTAG interface when it was not labelled. The latter does so by trying all possible permutations. Once the pins were known, we used a *Raspberry Pi zero* as our UART/JTAG/SPI connector. Having one Raspberry Pi Zero configured with all the required tools allowed us to gain time to collect the content of the different Integrated Circuits (IC) on the PCB.

3.4 Network Testing Setup

From a network perspective, we used two different setups, to match the two kinds of network interfaces available on the different devices and used to send data to the backend server:

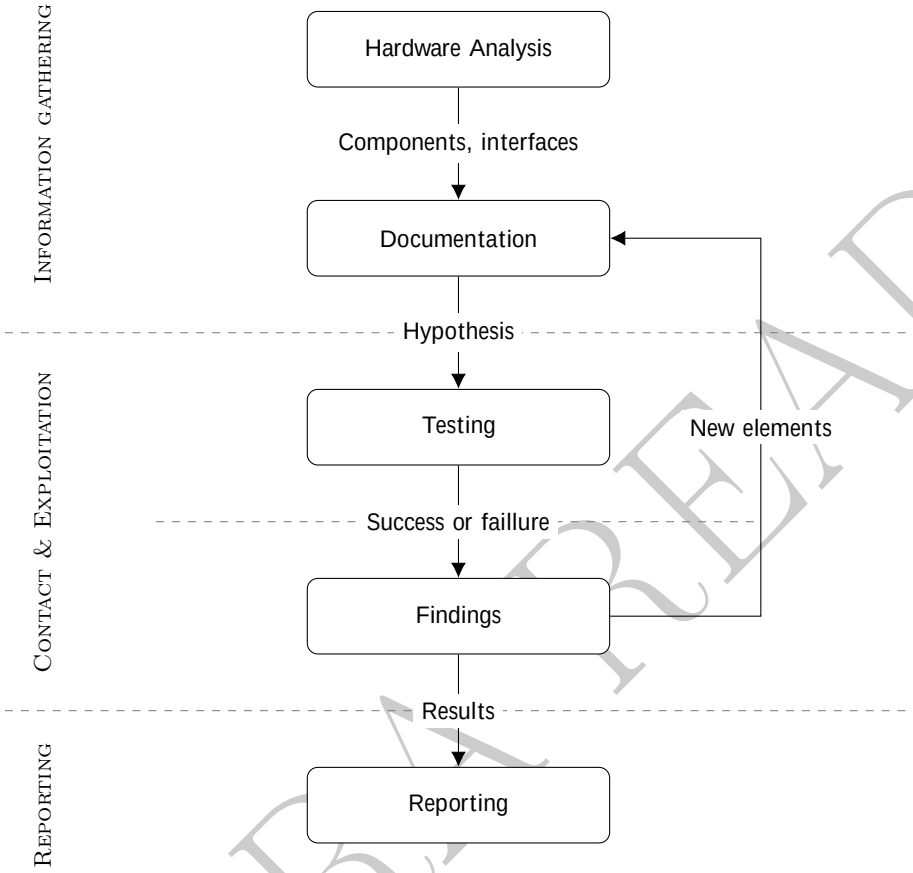

Fig. 3. Black Box Methodology iterative cycles used for our research [4]

- We developed a *modem emulator* to interact with the device using a telephone line (HMU (V2)).
- We setup a *Fake Base Station* (based on OpenBTS) to interact with the GSM (V2) and 3G (V3) versions of the HMUs. In addition, a network jammer was used to prevent the HMU from connecting via the 3G network, forcing a so-called downgrade attack. A virtual machine was set up to emulate the backend server of the manufacturer (as shown in Figure 4) [4,18].

These two setups allowed us to interact with the devices not only at the mobile network level, but also to directly capture the proprietary protocol used to exchange patient's data and the device's logs.

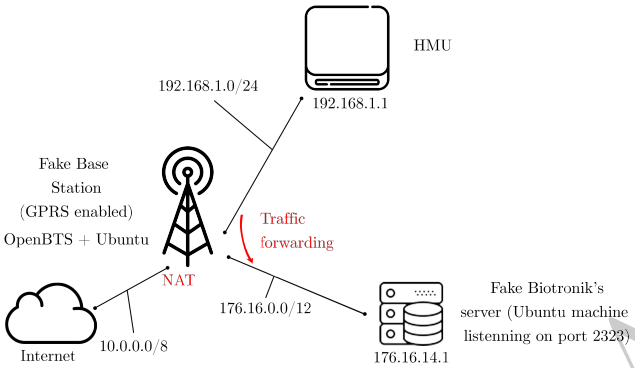

Fig. 4. Network diagram of the emulated network [4]

3.5 Modem Fuzzing Setup

The OWASP Foundation defines Fuzzing as "*a Black Box software testing technique, which basically consists in finding implementation bugs using malformed /semi-malformed data injection in an automated fashion.*"

We developed a custom fuzzing framework to fuzz the modem of the latest version of the HMU. Our framework is inspired by the initial work of Mulliner et al. [20] and built on top of the network testing setup from Lie [18]. Our fuzzing framework has the following components and characteristics:

- An infrastructure to send malformed SMS to the device. Achieving this required us to re-introduce the *testcall* feature in OpenBTS.
- A payload generator. This is not performed randomly, but rather in a context aware way: we used the Python package *smspdu* together with the *Intruder Payloads* from BurpSuite to generate our SMS PDU payloads.
- A monitoring service. When fuzzing, it is necessary to detect when a given payload triggers a bug or a vulnerability in the fuzzed device. This also allows for reproduceability. Traffic monitoring using *tcpdump* was used for this.

Figure 5 shows the overall Architecture of the Fuzzing Framework. Details on its development and capabilities can be found in [16].

3.6 Ethical Considerations

Given that the devices available in our lab have been acquired on the second-hand market, and that some of them were not new, they could have contained potentially sensitive data. This data has been systematically redacted from this paper and from previous publications. The Norwegian Centre for Research Data (NSD) was notified at the beginning of our project, and approved our patient data protection plan.

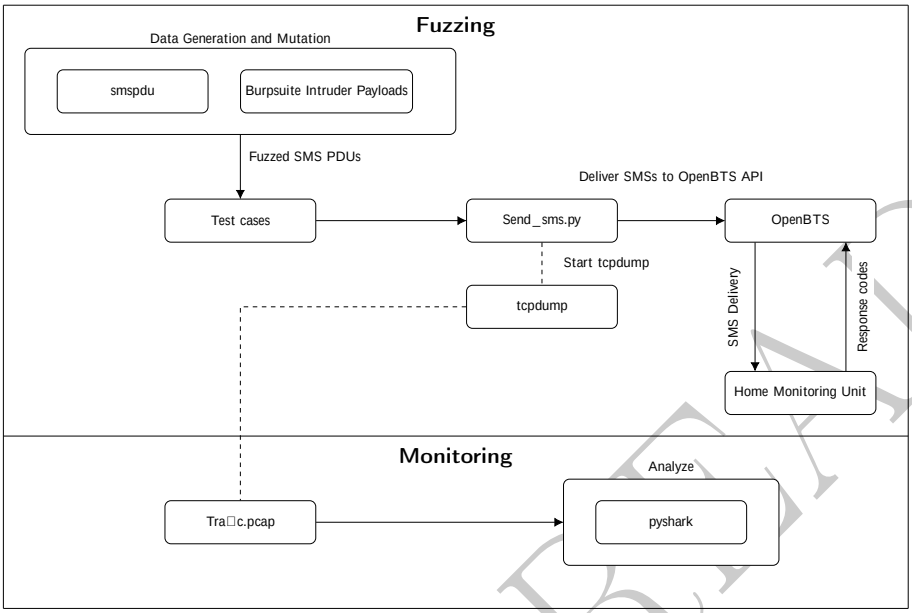

Fig. 5. Architecture of the Fuzzing Framework [16]

As the vulnerabilities discovered in the pacemaker ecosystem during our research could have had a potential impact on patients' safety and security, our findings were kept under embargo for one year. During this time, the research findings were shared with the vendor (BIOTRONIK) in the form of a vulnerability report. The vendor cooperated according to a coordinated vulnerability disclosure process and appropriately analyzed and validated our report. They then shared their responses to each reported vulnerability, and we discussed each point in detail. During these discussions, they also provided sufficient information to confirm that patient harm arising from the vulnerabilities is very unlikely. BIOTRONIK recommends that healthcare providers and patients continue to use the investigated devices as intended and follow device labelling. The coordinated vulnerability disclosure process also involved the German Federal Office for Information Security (BSI), the German Federal Office for Medical Devices (BfArM), the US Cyber Security and Infrastructure Agency (CISA) and the US Food and Drug Administration (FDA). As a result of this process, CISA issued an advisory [6].

The vulnerabilities discovered as part of the modem fuzzing not only impact the vendor's devices, but also all devices using that specific modem (and potentially all devices using modems from that vendor sharing the same software stack). Additionally, this is why we shared our findings with the modem's manufacturer and kept our results under embargo for two years. The vendor was informed about the vulnerabilities and how to re-produce attack-scenarios documentations.

4 Security Analysis

In this section, we present the findings from our security analysis on the different versions of the HMU. We first cover raw findings, split into four categories: Hardware, Firmware, Communication and Infrastructure. Then, we detail how an attacker can chain several of the highlighted weaknesses to mount an attack.

4.1 Hardware

Debug interfaces available. On all HMU versions analyzed, we were able to discover the UART and JTAG interfaces. On versions 1 and 2 the pins were not labelled, making it harder to determine the JTAG interface. On the latest version, however, pins were labelled. On all versions, the UART interface seemed disabled, and it was not possible to interact with it. The JTAG interface, on the other hand, was enabled and it is possible to fully control the microcontroller using it. That includes dumping the contents of the Random Access Memory (RAM) as well as the Flash Memory, which gave access to the firmware of the device.

4.2 Firmware

Data stored unencrypted on the external Flash Having physical access to a device, an attacker can use tools such as *flashrom* to dump the content of the external Flash on the PCB, connecting to it directly via SPI. By doing so, we could see that data was stored on it in cleartext.

Firmware not protected or obfuscated Once the firmware was dumped via the JTAG interface, reverse-engineering revealed that it was not encrypted or protected in any way. There was no trace of obfuscation of the code. On the contrary, log strings used by the device were explicit enough to ease the process of reverse engineering. This made it possible to create a script to easily fetch the credentials previously acquired via eavesdropping on the communication channel directly from the firmware, along with other credentials used by the device to connect to the backend server hosted by the manufacturer.

Memory not protected. The memory was not protected either, meaning that anyone with physical access to the HMU ccould copy it via the JTAG interface and access the data going through the HMU, including the patient's data, if any.

Hard-coded credentials and cryptographic keys. The credentials used by the devices to connect to the network and backend servers were hard-coded and stayed the same for each connection attempt. We observed, however, that they were unique for every device (two different HMUs will use different credentials). On the latest version they were stored on the external flash which is not encrypted and whose content can be read via the Serial Peripheral Interface (SPI). Cryptographic material such as Data Encryption Standard (DES) and Advanced Encryption Standard (AES) keys used in the proprietary protocol were also stored in a similar way.

Broken or risky algorithm. In the case of patient data, the proprietary protocol uses AES CBC as the encryption algorithm, however single DES is used in the case of log data going over SMS. DES is a broken algorithm from a security perspective, and log data can thus easily be obtained by an attacker that is able to set up a Fake Base Station in the proximity of the HMU. An attacker having had physical access once to the HMU could also perform the same attack on patient data by getting hold of the AES key. The keys (AES and DES) were, however, random and unique per device.

4.3 Communication

When analyzing the security of the communication link between the HMU and the backend server, we identified several weaknesses in the communication protocol.

Credentials sent in clear text to the modem. When analyzing the version 2 of the HMU, we were able to eavesdrop on the communication between the microcontroller and the modem as the pins of the modem were exposed on the PCB. This allowed us to get access to the credentials used by the device to connect to the manufacturer's VPMN, since these were sent in clear text.
No mutual authentication. We were able to spoof the backend server and trick the HMU into sending its data to us, highlighting the lack of mutual authentication between the backend server and the HMU on the first two versions. We did so at two different levels: first at the modem level on the Telephone line version, where we spoofed both the modem and the backend server, and second at the network level, where we used a virtual machine, connected to the same network as our OpenBTS machine, with the proper IP address requested by the HMU, to respond to the TCP request of the HMU (see Figure 4). The data obtained was encrypted for the most part. However, credentials to connect to the service were sent in cleartext before switching to the encrypted communication.
Usage of a proprietary protocol over an insecure transport protocol. Versions 2 and 3 of the HMU use both GPRS and SMS to send data. On both channels, the data is sent using a proprietary communication protocol on top of TCP. Leveraging the hardware vulnerabilities exposed above and the raw network data obtained by interacting with the devices, we were able to reverse engineer the protocol. It packs, compresses (when using GPRS) and encrypts the data. The detailed structure of a data packet is presented in Figure 6.
Credentials reuse. The credentials used to connect the VPMN and the backend services are the same and are sent unencrypted in both cases. They are thus very easy to obtain.
Unencrypted communication with the pacemaker Even though we have not done exhaustive research at this interface due to limited access to working compatible pacemakers in our lab, we found that there is no encryption of the data exchanged between the pacemaker and the HMU. That means that attackers who can intercept the radio signal from the pacemaker (the radio band is already known) can also access the patient's data.

Denial of Service of the modem. Results from the fuzz testing show that an attacker can perform a Denial of Service attack against the modem of the latest version of the HMU (V3) by sending it a specially crafted SMS. This will effectively prevent the HMU from communicating with the backend service before being rebooted.

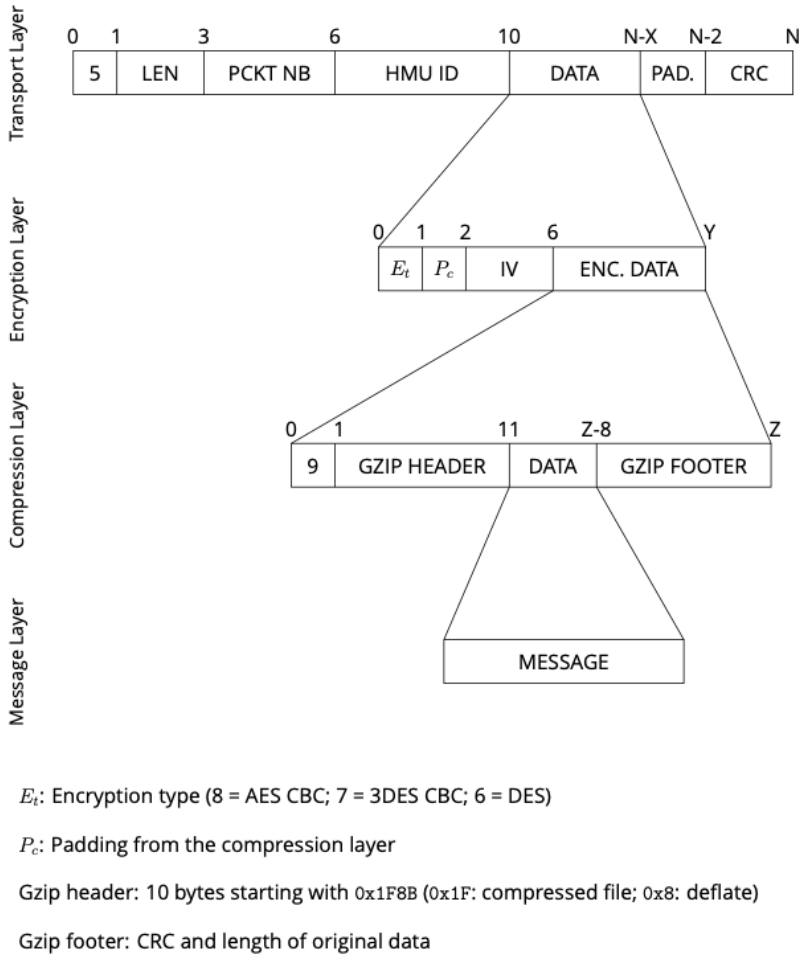

Fig. 6. Structure of the communication protocol's packet

4.4 Infrastructure

Improper devices management The HMU has two sets of credentials: the first to connect to the network and access the manufacturer VPMN; the second

to connect to the service on the backend server. To verify the validity of the credentials, we used them on a phone with the HMU SIM card and manually entered the settings in order to connect to the VPMN. However, when using the version 2 HMU SIM card, we were unable to connect because the SIM card was not valid anymore.

It turned out that using a SIM card from an old first version HMU on a newer second version HMU worked: we were able to connect to the VPMN and obtain an IP address inside the VPMN. To ensure we were in the right network, a successful ping request was sent to the server hosting the telemetry collection service. No other testing was performed as this was outside of our research scope and could potentially interfere with the manufacturer's service.

The VPMN is an additional security measure, even though this is not its main purpose. It prevents the patient data telemetry servers from being publicly exposed to the Internet, something that for instance protects against Denial of Service (DoS) attacks. However, our research showed that this protection can be bypassed by an attacker who acquires an old device with a valid SIM card, highlighting the need for proper decommissioning procedures for old devices.

4.5 General Considerations & Attack Scenarios

By chaining several of the vulnerabilities, we were able to *weaponize* the second version of the HMU. With physical access to the device, an attacker can install a physical device with a wireless communication interface inside of it (the inside of the HMU casing is big enough to add a *RaspberryPi zero*), and that way gain *remote access* to the device. This allows an attacker to not only eavesdrop on all communications between the HMU and the backend server, but also to act as a *Man-in-the-Middle*, the proprietary protocol being known. Such an adversary can also get access to all the data sent by the pacemaker to the HMU. This would enable an attacker to modify the pacemaker telemetry data in order to hide a possible problem, or to create a problem by deleting or modifying pacemaker alerts and warnings that were meant to be sent to the backend server.

In this subsection, we describe three attack scenarios against the HMU and more generally against the whole pacemaker ecosystem. Figures 7, 8 and 9 present the attack trees for these scenarios. Arrows indicate a requirement. An arc between several arrows indicates an "AND" condition while single arrows indicate an "OR" condition.

The first scenario is the *Man in the Middle* presented in Figure 7. Given the vulnerabilities described earlier, an attacker can spoof the identity of the HMU for the backend server and vice-versa. This means that an adversary can have full control over the information that is sent between these two entities. In order to target a patient, an attacker could for instance constantly send good reports, suppressing any alerts or warnings from the pacemaker. This could trick a clinician into thinking that the patient is doing great while in reality, the patient might be in urgent need of a check-up, for example, due to the pacemaker battery running out. Having an HMU would thus be more dangerous than having no

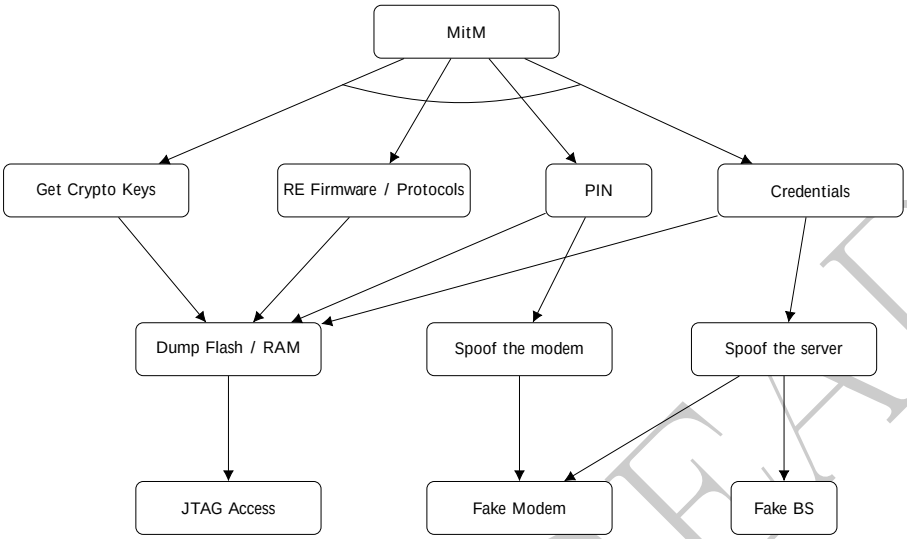

Fig. 7. Attack tree for the "MitM scenario" [3]

home monitoring enabled, due to a false sense of security and potentially fewer visits to the clinic.

The second scenario can be described as *Unauthorized access to the backend server* and is presented in Figure 8. We believe that this is possible with both versions of the HMU, given that the attacker can access credentials that are still valid. The attack tree shows only the GSM attack tree; the attack tree for the T-Line would be similar but easier since it only requires a working telephone line and no valid PIN or SIM. If an attacker can access the Virtual Private Network (VPN) with their computer using the credentials of the HMU, they would have direct access to the backend server (and all machines that reside in the same private network unless proper network segmentation with security monitoring is in place). If any of these machines are compromised, the result could be a significant data leak of personal data. Second-hand HMUs can be bought for a very low price on the internet, some come with their SIM cards still valid as we have demonstrated in our research, thus enabling an attacker to perform such an attack.

The last scenario is a large scale DoS attack on all the HMU devices of the pacemaker ecosystem. Our research has shown that an attacker can make an HMU crash by sending a specially crafted SMS to it. To recover from that crash, the device needs to be rebooted. Performing this attack requires knowledge of the phone number of the targeted device, thus making a large scale attack difficult. Such information could, however, be obtained in case of a data leak from the manufacturer or from the network provider. It could also come from an internal source. The attack tree for this attack is presented in Figure 9.

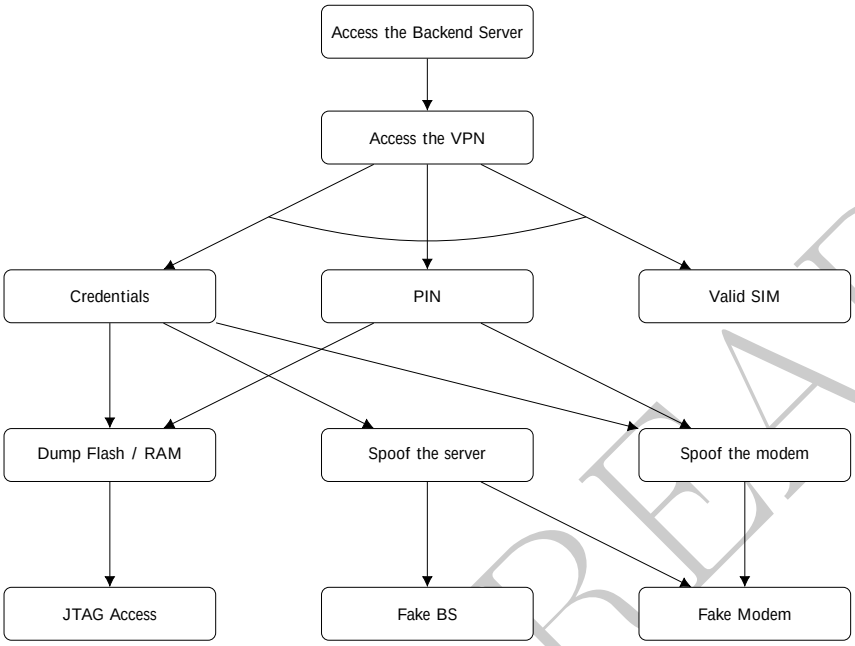

Fig. 8. Attack tree for the "Unauthorized access the backend server" scenario [3]

5 Discussion

5.1 Results

Our research confirms what was highlighted by Rios and Butts: the industry is overall still quite immature when it comes to cybersecurity [23]. Indeed, from a hardware point of view, an attacker with physical access to a device can easily get access to patients' data with no need for extensive knowledge or expensive equipment. From our observations, best security practices were not applied when it comes to hardware security given our findings of vulnerabilities that can all be described as commonly found in embedded devices. From a network perspective, several weaknesses have been identified in the protocol that is used by the HMU to communicate with the backend server, such as the credentials sent in clear text over TCP, the usage of a weak cipher to send data using SMS or the lack of mutual authentication in the second version of the HMU.

To be fair, it is also important to highlight that there is a notable evolution in terms of security between the versions. The latest version of the HMU seems to implement mutual authentication and stronger cryptographic ciphers than the previous versions. We can also point out that even though the second version has been found to have several weaknesses and vulnerabilities, the telemetry data was already encrypted using AES CBC with the keys being randomly generated and unique per device.

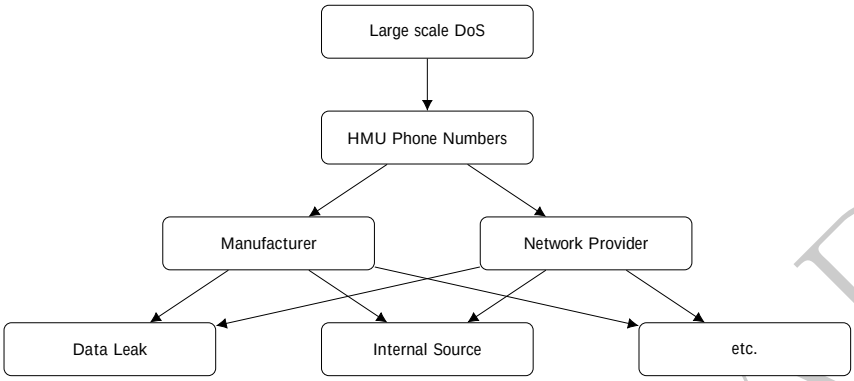

Fig. 9. Attack tree for the "Large scale DoS" scenario

5.2 Trade-offs in the Medical Industry

When designing IMDs, there are several security objectives to keep in mind. These are the regular six following properties: Confidentiality, Integrity, Availability, Non-repudiation, Authorization and Authentication. One also needs to consider the two modes under which these security properties have to be respected: *normal operation mode* and *emergency mode*. In the normal operation mode, the patient is in a state where it is reasonable to assume strict control of which devices can interact with the IMD, and it is feasible to implement strong access control, through the mean of cryptographic protocols for instance. Camara et al. explain that ideally, the device should not be detectable by unauthorized parties in this mode, and should "*ignore data requests or device reprogramming commands*" [5]. In emergency mode, even though the previously mentioned security objectives are important, it is vital that the device be accessible, for example if the patient is to undergo an emergency procedure for which the pacemaker must be deactivated.

It is thus a challenge for manufacturers to develop devices that fulfill all these characteristics. Zheng et al. highlight the trade-offs that come with the pacemaker ecosystem [30]. The first one is related to the emergency mode: *security vs accessibility*. Indeed, the purpose of the pacemaker is to save the patient's life and should not be an obstacle during an emergency surgery. The second trade-off is, *emergency access vs secure checkup access*. Securing the regular access while having an emergency access, which is almost like a security backdoor, is a challenge, especially when one must also take into account the battery life of the device. This leads to the third trade-off which is *limited resources vs strong cryptography*. Indeed, to secure the device, one needs to implement strong cryptography which requires intense processing power, this conflicts with the with low power capabilities and the long battery life time required by the implanted devices. This can even be abused by an attacker that launches Denial of Service (DoS) attacks in the form of constant wireless communication requests to drain the device battery, leading to a premature pacemaker battery depletion

which requires surgery and thus setting the patient's life at higher risk from complications.

5.3 Device Management & Credentials Invalidation

As shown in our research, hospitals and device manufacturers already have challenges when it comes to asset management of devices through their entire lifecycle. One example of this is the lack of de-commissioning of old HMU devices containing SIM cards that still had valid credentials for connecting to the manufacturer's back end systems. Asset management might become an even greater challenge when it comes to managing the future "hospital at home". Getting rid of hardcoded credentials, deprecating certificates and expired cryptographic keys and also the detection of potential rogue devices joining the network will have to be implemented to mitigate the risk.

5.4 Mitigation & Defense

As mentioned in the previous section, building safe and secure medical devices means facing several trade-offs. Several solutions have been proposed to solve the problem of having a secure access to the device while allowing access in emergency situations. Zheng et al. wrote a review [28] of the different mechanisms that could be used:

External proxy-based solutions. This idea was first proposed by Denning et al., and consists, as the name indicates, in having an external device called the Communication Cloakers to protect the implant [7]. This external device is carried by the patient and protects the implant from attacks in everyday life. In an emergency situation, when the clinician does not have access to the distributed key, she can simply remove the proxy. However, this also means that if patients forget or lose their proxy device, their implant becomes vulnerable to attacks again.

Biometric-based access control. This type of solution uses patients' biometric features in order to provide access. For instance, the Heart-to-Heart (H2H) scheme [24] makes sure that the pacemaker can only be accessed by a programmer in physical contact with the patient by using ECG signals to generate the crypto material to establish a secure wireless communication. Other solutions might use different biometric features, such as fingerprints, iris or even voice [29].

Proximity-based security schemes. In these schemes, the proximity of the device is used to determine whether or not a functionality is available. For instance, changing the device settings, which is a critical operation, requires close proximity (a few centimeters) while home monitoring is allowed up to 10m. If some authentication scheme such as Ultrasonic-AC [22] combine proximity and security credentials, others can be based on magnetic fields or short-range communication protocol. This is in fact what is currently used to secure pacemakers. Proximity-based security schemes have however been proven to be vulnerable if the attacker uses strong magnetic fields or simply use powerful and sensitive transceivers and high gain antennas [19].

Key distribution supporting emergency access. These schemes rely on cryptography to achieve secure access in normal situations while also keeping an emergency access. This includes symmetric cryptography, in which the key is distributed to authorized devices. For emergency situations, it is proposed that the key is carried by the patient, either with a smart card, on a bracelet or simply tattooed on the skin (with UV ink for instance) [26]. Public key cryptography can also be used. However, in the case of an emergency situation, the programmer needs to contact a trusted party to obtain the certificate that can be used to derive a symmetric key, and this requires access to the Internet. In addition, public key cryptography is not compatible with the low energy requirements of the pacemakers. Finally, it is possible to use biometric features to generate keys, as already explained for biometric-based solutions.

In their review, Zheng et al. also suggest possible solutions to address the resource constraints of IMDs [28].

Lightweight cryptography. In order to preserve the implant energy, manufacturers need to use lightweight cryptography protocols. Marin et al. propose a key agreement protocol that is an alternative to the proposal of Halperin et al., that was to add a standard symmetric key authentication and encryption between the ICD and the programmer, thus requiring the key to be safely stored on the programmer and opening the door to it being leaked. Marin et al. propose a semi-offline protocol: the IMD is in charge of computing a new key for the new period. To do so, the programmers need to contact the vendor to obtain the key for the period. That way, if the programmer is lost, or not in use anymore, it will not receive any updated key, and thus the ecosystem goes back to a secure state when the key is changed. Even though the new key computation is expensive for the IMD, this is a rare event and is thus not a problem.

Energy Harvesting. Another way to protect medical devices is through energy harvesting. Halperin et al. propose zero-power defenses for IMDs. These defenses include detection of attacks, prevention of attacks and a key exchange mechanism. As a detection mechanism, they propose to add a way to make the patient aware that there is something out of the ordinary happening, by for instance playing a "beep" if the security is disabled on the implant. The zero-power idea is to use a piezo-element driven by wireless communication (thus alerting a patient that wireless communication is taking place).

Separate security unit. Last but not least, the usage of a separate security unit that would be in charge of the security can mitigate the impact on the battery of the implant. This is for instance something that can be pushed to the external proxy device proposed above.

Moving away from the communication channel, another area that needs improvements is the software security. Li et al. propose a way to improve the trustworthiness of medical device software with formal verification methods [17]. They applied their approach to the firmware of a pacemaker and "demonstrated its ability to detect a range of software vulnerabilities that compromise security

and safety." The idea behind formal verification is not only to check for common vulnerabilities such as buffer overflows, use after free, etc. but also to go from the device specifications to verifiable properties. This can be, for example, the voltage of the pacing for a pacemaker in a given configuration. This approach allows the verification of real-world properties.

The healthcare domain has recently been plagued with cyber attacks in the form of ransomware attacks, where the intrusion often comes as a result of poor practices related to software patching and software inventory management. One mitigation that might help IT staff in deciding which software security updates need to be applied for securing medical devices is the introduction of a *Software Bill of Materials (SBOM)*, where the manufacturer declares all software components in a device. In 2018, the FDA published a Medical Device Safety Action Plan where one of the proposed actions was to require medical device manufacturers to include an SBOM as part of their premarket submissions.

Securing devices to which an attacker might have physical access is difficult. As mentioned in the Microsoft's Ten immutable Laws of Security, "*If a bad guy has unrestricted physical access to your computer, it's not your computer anymore.*" This is even more true for embedded medical devices, which usually do not come with as strong security defense mechanisms as computers. Indeed, adding strong hardware security to medical devices such as the Home Monitoring Unit also has a cost, and manufacturers might have to make a choice between security and costs, given that the money that is invested in the security of an HMU is not being used for developing treatment functionality, which saves lives. In addition to this, there is also the race to market and the strict certification process that does not allow easily making changes to an already approved design.

Fuzzing is a powerful methodology which can be used by both attackers and developers to discover bugs and potential vulnerabilities. Fuzzing might, however, not catch all issues, and manufacturers should ensure their devices are resilient to events such as crashes of external components (a modem for instance).

The industry is unfortunately not yet at the point where we can expect very strong cyber security in medical devices. As demonstrated by our research, basic security practices remain to be applied. A first step towards a more secure pacemaker ecosystem is the implementation of well-known best practices for hardware security, which, even if they do not protect against all attackers, can surely raise the cost of an attack, and simply discourage many attackers. Guides such the Secure Design Best Practice Guide by the IoT Security Foundation [13] provide a checklist of security measures to be adapted to a product, already during the design phase. When it comes to securing the firmware, the OWASP foundation offers a project for Embedded Application Security [21] that should be taken into account.

5.5 New Regulations

The European Union (EU) currently has two regulations medical devices manufacturers have to comply with: REGULATION (EU) 2017/745 (MDR) and REGULATION (EU) 2017/746 (IVDR) which apply from the 26th of May 2021

and 2022 respectively. These two new regulations replace older EU directives and aim at creating a regulatory framework for medical devices, to improve safety, quality and reliability.

In particular, the new regulation is paving the way for more testing of medical devices and should ease the work for manufacturers when it comes to fixing software (for instance, no need to re-certify for every minor change). This should also benefits security researchers as it will ease the responsible disclosure process.

Unfortunately, while great on paper, putting the regulation in practice comes with challenges. There are currently few devices certified with the new scheme as medical devices tend to have a long life. It will thus be hard to make legacy devices compliant with the new regulations, and this for quite some time still due to the also slow roll out of new devices.

6 Conclusion

Security of IoMT is a hot topic today, especially in the healthcare domain due to the need for patient safety and protection of critical data. In this paper, we evaluated the security aspects of the pacemaker ecosystem with a focus on hardware and network security architecture.

We have shown how the security protection features of several components in the pacemaker ecosystem are flawed. Leveraging physical access to the device and following a reverse engineering approach, we demonstrated that an attacker can expose the patient's data. Fuzzing the wireless communication chip of the device revealed vulnerabilities allowing an attacker to remotely trigger a DoS attack on the device.

Our focus was on the impacts of insecurity of the pacemaker ecosystem on patient safety. However, our results also imply that integrating such IoMT devices with major shortcomings into the complex hospital infrastructure could open the door to attacks with impacts that are potentially as devastating as a ransomware attack on hospital services [8,9].

Acknowledgments

We very much appreciate the contributions of Éireann Leverett that did some of the initial hardware testing to discover the HMU debug interfaces. Finally, we are grateful to Snorre Aunet and Ingulf Helland from NTNU who took time to help us solder a connector on the HMU.

This work was funded by Reinforcing the Health Data Infrastructure in Mobility and Assurance through Data Democratization, a five-year project (grant number 28885) under the Norwegian IKTPLUSS-IKT and Digital Innovation programme. The authors gratefully acknowledge the financial support from the Research Council of Norway.

References

1. OWASP firmware security testing methodology, https://scriptingxss.gitbook.io/firmware-security-testing-methodology/
2. Block, C.C.: Muddy waters report - st. jude medical, inc. Tech. rep., Muddy Waters Capital LLC (August 2016), http://www.muddywatersresearch.com/research/stj/mw-is-short-stj/
3. Bour, G., Moe, M.E.G., Borgaonkar, R.: Experimental security analysis of connected pacemakers. In: Roque, A.C.A., Fred, A.L.N., Gamboa, H. (eds.) Proceedings of the 15th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2022, Volume 1: BIODEVICES, Online Streaming, February 9-11, 2022. pp. 35–45. SCITEPRESS (2022). https://doi.org/10.5220/0010816900003123, https://doi.org/10.5220/0010816900003123
4. Bour, G.N.: Security Analysis of the Pacemaker Home Monitoring Unit: A Black-Box Approach. Master's thesis, NTNU (2019)
5. Camara, C., Peris-Lopez, P., Tapiador, J.E.: Security and privacy issues in implantable medical devices: A comprehensive survey. Journal of biomedical informatics **55**, 272–289 (2015)
6. CISA: ICS Medical Advisory (ICSMA-20-170-05). https://us-cert.cisa.gov/ics/advisories/icsma-20-170-05 (2020), [Online; accessed 30-Sep-2021]
7. Denning, T., Fu, K., Kohno, T.: Absence makes the heart grow fonder: New directions for implantable medical device security. In: HotSec (2008)
8. Digital, N.: A clear and present danger (2022), https://digital.nhs.uk/features/a-real-and-present-danger
9. Europol: Covid-19 sparks upward trend in cybercrime (2022), https://www.europol.europa.eu/media-press/newsroom/news/covid-19-sparks-upward-trend-in-cybercrime
10. Færestrand, S.: Telekardiologi for jernmonitorering av pacemaker og icd (2010), https://www.legeforeningen.no/contentassets/4896657d08894a6886de725113d89de4/hjerteforum3-2010web08telemedisin.pdf
11. Golde, N., Feldmann, A.: SMS Vulnerability Analysis on Feature Phones. Master's thesis (2011)
12. Halperin, D., Heydt-Benjamin, T.S., Ransford, B., Clark, S.S., Defend, B., Morgan, W., Fu, K., Kohno, T., Maisel, W.H.: Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In: 2008 IEEE Symposium on Security and Privacy (sp 2008). pp. 129–142. IEEE (2008)
13. Secure design best practice guide (2020), https://www.iotsecurityfoundation.org/wp-content/uploads/2019/03/Best-Practice-Guides-Release-1.2.1.pdf, [Online]
14. ISECOM: OSSTMM, https://www.isecom.org/OSSTMM.3.pdf
15. Justis- og beredskapsdepartementet, Helse- og omsorgsdepartementet: Forskrift om medisinsk utstyr (2005), https://lovdata.no/dokument/SF/forskrift/2005-12-15-1690/%2FT1%2Ftextsection1-5#/T1/textsection1-5
16. Kok, J.S., Markussen, B.A.: Fuzzing the Pacemaker Home Monitoring Unit. Master's thesis, NTNU (2020)
17. Li, C., Raghunathan, A., Jha, N.K.: Improving the trustworthiness of medical device software with formal verification methods. IEEE Embedded Systems Letters **5**(3), 50–53 (2013)
18. Lie, A.W.: Security Analysis of Wireless Home Monitoring Units in the Pacemaker Ecosystem. Master's thesis, NTNU (2019)

19. Marin, E., Singelée, D., Garcia, F.D., Chothia, T., Willems, R., Preneel, B.: On the (in) security of the latest generation implantable cardiac defibrillators and how to secure them. In: Proceedings of the 32nd annual conference on computer security applications. pp. 226–236 (2016)
20. Mulliner, C., Golde, N., Seifert, J.P.: {SMS} of death: From analyzing to attacking mobile phones on a large scale (2011), https://www.usenix.org/conference/usenix-security-11/sms-death-analyzing-attacking-mobile-phones-large-scale
21. OWASP embedded application security (2020), https://owasp.org/www-project-embedded-application-security/, [Online]
22. Rasmussen, K.B., Castelluccia, C., Heydt-Benjamin, T.S., Capkun, S.: Proximity-based access control for implantable medical devices. In: Proceedings of the 16th ACM conference on Computer and communications security. pp. 410–419 (2009)
23. Rios, B., Butts, J.: Security evaluation of the implantable cardiac device ecosystem architecture and implementation interdependencies (2017)
24. Rostami, M., Juels, A., Koushanfar, F.: Heart-to-heart (h2h) authentication for implanted medical devices. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. pp. 1099–1112 (2013)
25. Savci, H.S., Sula, A., Wang, Z., Dogan, N.S., Arvas, E.: Mics transceivers: regulatory standards and applications [medical implant communications service]. In: Proceedings. IEEE SoutheastCon, 2005. pp. 179–182. IEEE (2005)
26. Schechter, S.: Security that is meant to be skin deep using ultraviolet micropigmentation to store emergency-access keys for implantable medical devices (2010)
27. Weinmann, R.P.: Baseband attacks: Remote exploitation of memory corruptions in cellular protocol stacks. In: WOOT. pp. 12–21 (2012)
28. Zheng, G., Shankaran, R., Orgun, M.A., Qiao, L., Saleem, K.: Ideas and challenges for securing wireless implantable medical devices: A review. IEEE Sensors Journal **17**(3), 562–576 (2016)
29. Zheng, G., Yang, W., Valli, C., Qiao, L., Shankaran, R., Orgun, M.A., Mukhopadhyay, S.C.: Finger-to-heart (f2h): Authentication for wireless implantable medical devices. IEEE journal of biomedical and health informatics **23**(4), 1546–1557 (2018)
30. Zheng, G., Zhang, G., Yang, W., Valli, C., Shankaran, R., Orgun, M.A.: From wannacry to wannadie: Security trade-offs and design for implantable medical devices. In: 2017 17th International Symposium on Communications and Information Technologies (ISCIT). pp. 1–5. IEEE (2017)