DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
*STOCKHOLM, SWEDEN 2021*

# Security evaluation of a smart lock system

**RAIHANA HASSANI**

**KTH ROYAL INSTITUTE OF TECHNOLOGY**
**SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE**

# Security evaluation of a smart lock system

Raihana Hassani

Bachelor in Computer Science
Date: February 8, 2021
Supervisor: Martin Jacobsson
Supervisor NSE: Pontus Johnson
Examiner: Ibrahim Orhan
Swedish title: Säkerhetsutvärdering av ett smart låssystem

# Abstract

Cyber attacks are an increasing problem in the society today. They increase dramatically, especially on IoT products, such as smart locks. This project aims to evaluate the security of the Verisure smart lock system in hopes of contributing to a safer development of IoT products and highlighting the existing flaws of today's society. This is achieved by identifying and attempting to exploit potential vulnerabilities with threat modeling and penetration testing. The results showed that the system is relatively secure. No major vulnerabilities were found, only a few weaknesses, including the possibility of a successful DoS attack, inconsistent password policy, the possibility of gaining sensitive information of a user and cloning the key tag used for locking/unlocking the smart lock.



**Keywords:** Cyber security, threat modeling, penetration testing, ethical hacking, smart locks, STRIDE, DREAD.

# Sammanfattning

Cyberattacker är ett ökande problem i samhället idag. De ökar markant, särskilt mot IoT-produkter, såsom smarta lås. Detta projekt syftar till att utvärdera säkerheten i Verisures smarta låssystem i hopp om att bidra till en säkrare utveckling av IoT-produkter och belysa de befintliga bristerna i dagens samhälle. Detta uppnås genom att identifiera och försöka utnyttja potentiella sårbarheter med hotmodellering och penetrationstestning. Resultaten visade att systemet är relativt säkert. Inga större sårbarheter hittades, bara några svagheter, inklusive möjligheten till en lyckad DoS-attack, inkonsekvent lösenordspolicy, möjligheten att få känslig information från en användare och kloning av nyckelbrickan som används för att låsa/låsa upp smarta låset.



**Nyckelord:** Cybersäkerhet, hotmodellering, penetrationstestning, etisk hackning, smarta lås, STRIDE, DREAD.

# Contents

# 1. Introduction

According to Check Point's Software Security Report 2020 [1], cyber attacks continue to rise. For example, they report that in January 2019 21 million passwords and more than 770 million email addresses were exposed and in June 2019 approximately 20 million patients got their sensitive information exposed in a data breach. One of the biggest reasons for the rise of cyber attacks is the massive increase in online services and connected devices, called IoT, Internet of Things, devices [1].

In late 2019, security researchers at F-secure managed to find a vulnerability in the KeyWe smart lock. This vulnerability allowed hackers to unlock the smart lock and break into the user's home [2]. A year earlier, in 2018, Forbes reported about a Z-wave hack that left up to 100 million smart home devices, such as smart locks, exposed [3]. Furthermore, Itai Greenberg predicts, in Check Point's security report, an immense increase in IoT products with the distribution of 5G. He also claims that most IoT devices are not protected at all, leading to an increased risk of cyber attacks performed successfully. He urges: "Now is the time to take action and secure IoT the same way we secure IT" [1]. The first step to achieving this goal is through extensive threat modeling and penetration testing [4].

One category of connected devices increasing in popularity is smart locks. Smart locks have an attractive market because of the many new features they offer, features that classic locks lack. One can for instance lock and unlock remotely via their respective mobile or web application.

## 1.1 Problem definition

Smart locks can also provide better supervision for the user, such as history log of all activities of the smart lock and distribution of temporary codes to family and friends for smoother access to the user's home. Verisure's smart lock system, Yale Doorman V2N together with Verisure's V-module and V-Box Mini, provide all these features and more [5].

However, the question remains: how secure is the smart lock system? Can unauthorized people gain access to the system and cause damage? These are important issues for the safety of all users.

## 1.2 Objective

One of the most effective and used methods and techniques towards a more secure system is threat modeling. With threat modeling, one can detect potential threats at an early stage and find effective countermeasures against them [6]. In this project, threat modeling and penetration testing will be used as a method for conducting a security analysis of an IoT product.

The purpose of penetration testing is to evaluate the security of a system. This includes identifying vulnerabilities and attempting to exploit them, to determine if unauthorized access to the system or other malicious activity is possible [6].

In this project Verisure's smart lock system will be examined from a security perspective. The aim is to conduct a list of possible attacks and apply them on the Verisure smart lock system, as well as to contribute to a safe development of IoT devices and highlight the existing flaws of today's society.

## 1.3 Delimitations

Depending on the results from the threat modeling, the most harmful and probable attacks are selected for the penetration tests, i.e. the ones with a high risk rate. However, the law must be taken into consideration and the Swedish law "Brottsbalken 4 kap. 9c §" Lag (2014:302) [7] states that it is not allowed to hack into someone else's property. This indicates that access to Verisure's servers without consent is not permitted during this project as it is their property. Other delimitations are

time and tools, it is impossible to perform all penetration tests within the allotted time and without sophisticated tools. Due to the time constraint, the security evaluation does not include the hardware.

## 1.4 Report outline

In this chapter the problem, objective and delimitations were introduced, and now the report is outlined. After this initial chapter, there are eight more to come:

Chapter 2 – describes the necessary theory and background.

Chapter 3 – describes the followed methodology, both for threat modeling and penetration testing.

Chapter 4 – presents the conducted threat model of the system under consideration, all the assets, architecture diagram and threats.

Chapter 5 – presents all the penetration tests performed in this thesis, their background, methods used, results and discussions.

Chapter 6 – summarizes all the findings from the penetration tests presented in the previous chapter.

Chapter 7 – summarizes the discussions from all the penetration tests in chapter 5.

Chapter 8 – discusses the sustainability and ethical considerations made during this thesis.

Chapter 9 – contains the drawn conclusion of the thesis and future work.

# 2. Background

In this chapter, the necessary theory and background is introduced. Threat modeling and ethical hacking is presented in section 2.1 and 2.2, previous work related to this thesis is presented in section 2.3, and the system under consideration is presented in section 2.4.

## 2.1 Threat modeling

Threat modeling is the first and most important step towards safer systems. Several historical attacks could have been prevented through the effective use of threat modeling, one example being the Mirai DDoS attacks that took place in 2016 [4].

Threat modeling is a structured analyzing process meant to find vulnerabilities in the system in order to defend the product against cyber attacks. It is conducted by the product's developers and researchers in the interest of keeping their company secure and trustworthy. More specifically, it is a list of different methodologies used to map out the system by finding the potential attacker's goals. This process is documented to retain all potential threats on the system flaws. It is important to understand that threat modeling is all about finding these vulnerabilities before the attackers and working proactively. [8, 9]

There are several different renowned methodologies for conducting a threat model. However, regardless of which methodology is chosen, after the threat modeling, the companies will have a clean and well documented product design. This will make their work easier and more reliable in the long term. [8]

## 2.2 Ethical hacking

Ethical hacking, also called penetration testing, is when a hacker legally performs attacks in order to gain access to a system unauthorized. The purpose of performing penetration tests is to identify vulnerabilities, to report them and suggest countermeasures before a hacker exploits them [10]. Penetration tests can be done either through a black box perspective or a white box perspective. Black box testing is when the tests are done without any knowledge of the internal structure beforehand. White box testing is the opposite, tests done with knowledge of the internal structure. [4]

## 2.3 Previous work

Smart locks have been researched on multiple occasions before. Two of these studies are "Security evaluation of smart locks" [11] and "Finding Vulnerabilities in IoT Devices: Ethical Hacking of Electronic Locks" [12]. In these studies, they have, similar to this project, used threat modeling and penetration testing.

### 2.3.1 Security evaluation of smart locks

The project "Security evaluation of smart locks" was done in 2019 by Arvid Viderberg [11]. As the title suggests, this project investigated the security aspects of smart locks by conducting a threat model and performing penetration tests. A list of 71 previously common attacks on smart locks and similar IoT devices was presented but only three of them were tested on a set of smart locks in this project. There was also a list of smart locks to choose between. Yale Doorman was included twice, one with Aptus module and the other without. Two other smart locks were chosen from the list, they are called Net Smart lock and Ali Lock.

The aim of the project was to investigate whether previous acknowledged attacks would work on comparably newer smart locks available in 2019 and concluded that some previous vulnerabilities did

indeed still exist. The results showed several deficiencies. Area of vulnerabilities included state consistency, password policies and password reset mechanisms.

### 2.3.2 Finding Vulnerabilities in IoT Devices: Ethical Hacking of Electronic Locks

The project “Finding Vulnerabilities in IoT Devices: Ethical Hacking of Electronic Locks”, also from 2019, was done by Joachim Toft and Christopher Robberts [12]. This thesis tests the security status of Bluetooth smart locks. Just like Viderberg in the previous section, Toft and Robberts started with threat modeling, followed by performing well-known penetration tests. They concluded that some low-impact design flaws existed.

## 2.4 The system under consideration

The system under consideration consists of the smart lock, Yale Doorman V2N (figure 1), the Verisure V-module and the Verisure V-Box Mini (figure 2). The smart lock can be used separately, then it acts as a lock only. It can be used with key tags, user codes or both combined for increased security. However, if used with the Verisure V-module and V-Box Mini, a lot more features can be utilized. These extensions to the lock enable the use of the mobile application and web application, with which one can for instance lock and unlock at a distance. [13]

The applications have multiple useful features in addition to locking and unlocking from a distance. These activities, locking and unlocking, as well as other activities, are logged in the application under “History”. Parents can in this way for example be informed of when the kids come home. The user can also give out temporary codes to relatives, friends or a worker, allowing them to enter the house when nobody is home. The user can also activate notifications, providing them with immediate information about a change in the system, such as low battery on the Yale Doorman V2N or disturbances in the connection between the different components. Another feature is automatic locking, Yale Doorman V2N locks automatically after 10 seconds. [14]

There are security requirements for smart locks in Scandinavia and this is tested by the SSF, The Swedish Theft Prevention Association. Yale Doorman V2N is approved despite SSF´s high standards; 3522:1 safety class 3 and can be found of their "Safety guide". Yale Doorman V2N is also certified by SBSC, the Swedish Fire and Safety Certification. [15]

Figure 1: Yale Doorman V2N

Figure 2: Verisure V-module (left) and Verisure V-Box Mini (right)

# 3. Methodology

A literature review was conducted to gather as much information as possible within the allotted time frame. The information was taken from various relevant books, journal articles and web pages, which were first processed and analyzed critically. The keywords “threat modeling”, “penetration testing”, “wireless hacking” and “STRIDE” were used, among others, to find the relevant information. These keywords were used in several databases, including Primo, Youtube and Google’s search engines.

Thereafter, with the help of the information obtained from the literature review, a threat model was developed for the system under consideration. Based on this threat model, penetration tests were performed to find potential vulnerabilities in the Verisure system. These tests were performed on a virtual machine, more specifically, Kali Linux.

## 3.1 Threat model

A threat model was developed for Verisure’s smart lock system to obtain a better overview of the threats to the system. There are several different methodologies and models that can be used for threat modeling, such as STRIDE, PASTA, DREAD and CVSS. For the purpose of this project, STRIDE and DREAD were most suitable, as they are well-documented and efficient models with a focus on the product [16]. In comparison, PASTA and CVSS are more suitable for corporations, PASTA includes analysis of business effects, which leads to a more complicated methodology and involvement of several different sectors in addition to the IT department [9, 17]. Guzman and Gupta [4] comment on CVSS: ”CVSS can be quite useful for reporting vulnerabilities to vendors but may not be as straightforward for threat modeling purposes.”.

One very popular approach to threat modeling that uses the STRIDE and DREAD model, and one that Guzman and Gupta [4] suggest and OWASP [18] refers to, is that of Microsoft. Hence, the choice was made to follow this approach. However, regardless of which approach is chosen, when conducting a threat model, it is important to work iteratively, i.e. to return to the document and update it whenever new information is acquired since this new information can change the threats to the system. [4, 19]

### 3.1.1 Microsoft’s threat modeling process

Microsoft’s threat modeling process consists of six steps [19]:

1. Identify IoT assets
2. Create an IoT device architecture overview
3. Decompose the IoT device
4. Identify threats
5. Document threats
6. Rate the threats

**Identify assets**

In this first step assets that are valuable and in need of protection are identified [19]. This is done to obtain more knowledge about where to concentrate the more plausible attacks to. If an asset comprises a public vulnerability, it will be a lot easier and/or quicker to exploit the system. The assets are usually documented in a table with their description. [4]

**Create an architecture overview**

Creating an architecture overview of the IoT device is the second step in Microsoft’s threat modeling process. This is done to visualize the attack surfaces of the system, to obtain a better understanding of the system’s features and functionality and thus discover flaws easier. It will be accomplished

through listing use cases and drawing an architectural diagram of the system to identify which technologies are used. [4, 19]

**Decompose the IoT device**
When decomposing the IoT device, the data flow, trust boundaries and entry points of the system are identified and analyzed. This is accomplished by creating a data flow diagram that displays the aforementioned data flow, trust boundaries and entry points. After the entry points and protocols in use are identified, a better understanding of the attack surfaces and potential vulnerabilities are gained, which accelerates the hacking process. [4, 19]

**Identify threats**
Threats are identified using the STRIDE model. STRIDE stands for Spoofing identity, Tampering with data, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege, which are different categories for security threats [20]. Microsoft [20] describes these six categories well:

- *Spoofing identity* refers to the use of another person's identity over the internet, with the intent to gain access to a system illegally.
- *Tampering with data* involves the manipulation of data by an unauthorized person. This tampering may be on persistent data or data flowing throughout the system.
- *Repudiation* is the denial of an action, an action that cannot be proved to have taken place by the accused.
- *Information disclosure* means that an unauthorized person has access to information.
- *Denial of Service*, or abbreviated DoS, is the overload of a system/server, which denies users access to the provided services.
- *Elevation of privilege* is when a user somehow manages to get more privilege. Consequently, an unprivileged user can exploit the same rights as a privileged user.

**Document threats**
Now the threats identified in the previous step are documented. This is preferably done in tables, with each threat having its description, threat targets, attack techniques and countermeasures described. [4]

**Rate the threats**
The last step in Microsoft's threat modeling process is to rate the threats identified and documented in the previous steps. As stated in 3.1 the DREAD model was chosen for this task. DREAD stands for Damage potential, Reproducibility, Exploitability, Affected users and Discoverability. For each one of these five categories, the risk is rated from 1-3, with 1 being low risk and 3 being high risk, see figure 3 below [4, 19]. The figure is taken from Microsoft [19]. Thus, the final result of a threat, which is calculated by adding together the rates for all the five categories, will be between 5-15.

| | Rating | High (3) | Medium (2) | Low (1) |
|---|---|---|---|---|
| D | Damage potential | The attacker can subvert the security system; get full trust authorization; run as administrator; upload content. | Leaking sensitive information | Leaking trivial information |
| R | Reproducibility | The attack can be reproduced every time and does not require a timing window. | The attack can be reproduced, but only with a timing window and a particular race situation. | The attack is very difficult to reproduce, even with knowledge of the security hole. |
| E | Exploitability | A novice programmer could make the attack in a short time. | A skilled programmer could make the attack, then repeat the steps. | The attack requires an extremely skilled person and in-depth knowledge every time to exploit. |
| A | Affected users | All users, default configuration, key customers | Some users, non-default configuration | Very small percentage of users, obscure feature; affects anonymous users |
| D | Discoverability | Published information explains the attack. The vulnerability is found in the most commonly used feature and is very noticeable. | The vulnerability is in a seldom-used part of the product, and only a few users should come across it. It would take some thinking to see malicious use. | The bug is obscure, and it is unlikely that users will work out damage potential. |

Figure 3, Microsoft's rating table for DREAD.

## 3.2 Penetration testing

When deciding which penetration tests to perform, the delimitations specified in chapter 1.3 is taken into consideration. The attacks are performed on Kali Linux, which is chosen because of its convenience, as it comes preinstalled with a lot of hacking tools such as Wireshark and Ettercap [21]. Kali Linux is installed and run as a virtual machine on Oracle VM VirtualBox. The penetration testing is performed from a black box perspective since it was done independently without Yale's or Verisure's participation and thus without knowledge about the internal structure of the system.

# 4. Threat model

This chapter is a threat model of the system under consideration, the structure follows Microsoft's threat modelling steps. In 4.1 assets are identified, in 4.2 an architecture overview of the Verisure system is created and the system is decomposed, in 4.3 threats are identified, in 4.4 these threats are documented and in 4.5 they are rated. This threat model is then used for the penetration testing process.

## 4.1 Identifying assets

The assets of the system under consideration were identified and documented in the table below. Some of the information was found on Verisure's web page and user manual and others by analyzing the traffic and scanning the network with the network scanning tool Nmap. With Nmap the IP-address and MAC-address of the Verisure hub were found, as well as the manufacturer; Securitas Direct AB. It was also found during port scanning that all ports were either closed or filtered.

Table 1: The assets and their description of the Verisure smart lock system.

| ID | Assets | Description |
|---|---|---|
| 1 | Yale Doorman V2N (smart lock) | Can also be used separately, i.e. without the Verisure system. Can be locked/unlocked via user codes and/or key tags. The key tags are Mifare Classic 1K tags. Runs on four AA batteries. A temporary four-digit code can be set and erased. User codes and key tags can be blocked at any time. [22] |
| 2 | V-module (Verisure smart lock module) | Connected physically to the lock, handles the communication between the lock and the Verisure hub, V-Box Mini. |
| 3 | V-Box Mini (Verisure hub) | V-Box Mini is the hub and can control up to six V-modules [23]. It is connected to the router through an ethernet cable. It also has an GSM (Global System for Mobile Communications) connection to the Verisure server. Does not run on batteries, it is instead plugged to the electrical outlet for power. It communicates via SCTP and DTLS. The SCTP packets are encapsulated in UDP, see figure 4. |
| 4 | Mobile application and web application | The mobile application can be downloaded from both Google Play on Android devices and App Store on iOS devices.<br>The mobile and web application allows for effective and quick control over the system remotely and adds features not available on the Yale Doorman V2N by itself. The mobile and web application communicate with the Verisure server via HTTPS, on port 443. |
| 5 | Firmware | Yale Doorman V2N: FW v2.1. The firmware is out of scope in this project. |
| 6 | Wireless communication | The communication between the V-module and V-Box Mini is an RF communication.<br>The communication between the mobile application/web application and the router is Wi-Fi. |

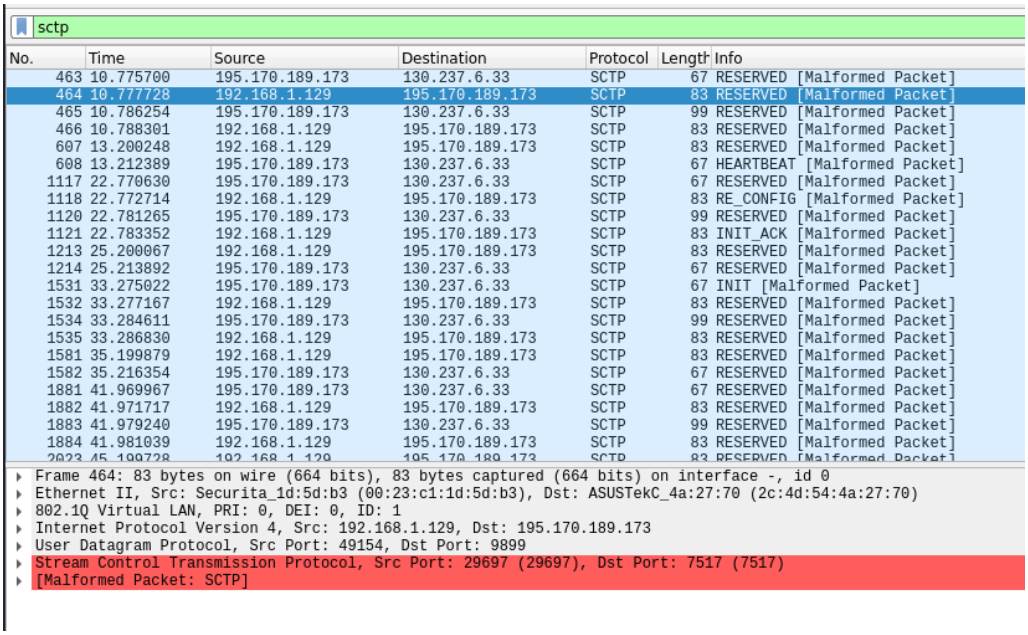


Figure 4. A screenshot from Wireshark of a packet sent to Verisure's server. As seen, the packet is a an SCTP packet encapsulated in UDP. The V-Box Mini connects to the SCTP port 9899.

## 4.2 Use cases and data flow diagram

Below are a few use cases that were created for the purpose of documenting the functionalities and features of the system under consideration. After the use cases, the data flow diagram of the system under consideration is presented in figure 5 in 4.2.1.

**Use case 1: User locks or unlocks the smart lock remotely via the mobile application**

1. User downloads the mobile application.
2. User creates an account if they do not already have one.
3. User logs in to "my pages".
4. User installs the needed components using the eight-character serial number found on the components if they have not already done it via the website.
5. User clicks on the lock icon and types in the PIN or uses Touch-ID/FaceID/Biometrics.
6. User gets a push notification if they have push notifications activated.

**Use case 2: User checks log via the mobile application**

1-4. Same as use case 1.
5. User goes to "History".

**Use case 3: User checks connectivity between V-Box and Verisure's server**

1. User presses the button on V-Box Mini.
2. User receives an update of the connectivity via email.

**Use case 4: User activates push notifications**

1-4. Same as use case 1.
5. User goes to "Users", selects the user, clicks on "Notifications" and turns on push notifications.

**Use case 5: User emergency opens the smart lock from outside in case of drained batteries and changes the battery once inside**

1. User holds a 9V battery to the bottom of the lock.
2. User unlocks the lock with their electronic key.
3. User unscrews the battery cover.
4. User replaces the old batteries with new ones.
5. User screws back the battery cover.

### 4.2.1 Data flow diagram

A data flow diagram is created to display the data flow, trust boundaries and entry points of the system under consideration. This is done to gain better understanding of the attack surfaces and potential vulnerabilities.

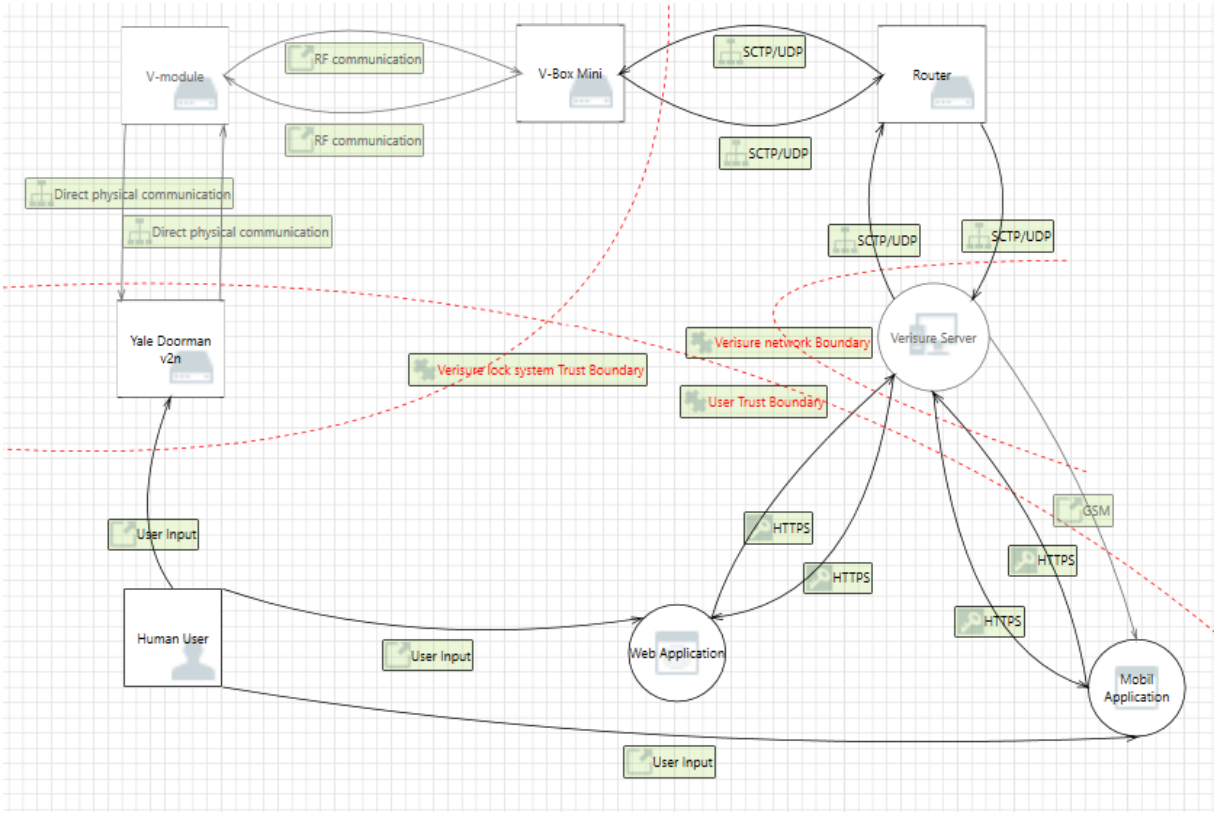


Figure 5: Data flow diagram of the system under consideration designed in Microsoft Threat Modeling Tool. The circles and boxes represent the system devices and agents communicating with the system, the dashed red lines represent the trust boundaries and the arrows illustrate the data flow, i.e. the communication paths, of the system. The Verisure server, the V-module, the RF communication and the GSM connection are out of scope, which is shown in the Microsoft Threat Modeling Tool by making them greyed out.

## 4.3 Identifying threats

When identifying threats, the Microsoft Threat Modeling Tool, OWASP's top 10 lists and Adam Shostack's book "Threat Modeling: Designing for Security" [24] were used. The Microsoft Threat Modeling Tool generates threats based on the created data flow diagram in a threat modeling report. The top 10 lists used were top 10 IoT threats [25], top 10 web application threats [26] and top 10 mobile application threats [27].

Table 2: The identified threats with the STRIDE model applied.

| Threat category | Description |
|---|---|
| Spoofing | Spoofing the identity of a user or Verisure's components and server by sniffing the traffic.<br>Spoofing a user by cloning their Yale key tag.<br>Spoofing a user by brute forcing. |
| Tampering | Redirecting and modifying the packets sent throughout the system.<br>Modifying the android application code or files.<br>Modifying a Yale key tag. |
| Repudiation | |
| Information disclosure | Analyzing and sniffing information from the communication through the router.<br>Reverse engineering the android application code or files to get access to information not intended for disclosure.<br>Gaining user credentials via the password reset function.<br>Skimming a Yale key tag to extract the information from the key tag. |
| Denial of Service | Denying the user service to the Verisure system. |
| Elevation of Privilege | Gaining privileges through horizontal privilege escalation. |

## 4.4 Documenting the threats

The threats identified in table 2 with the help of STRIDE, are now documented through tables 3-12.

Table 3: Threat #1

| | |
|---|---|
| Threat description | The attacker gains authentication information of a user and therefore gains unauthorized access to the lock, mobile application or web application. |
| Threat target | The lock, mobile application and web application. |
| Attack techniques | The attacker can intercept the Internet communication between the Verisure hub and the Verisure Server and likewise between the mobile application/web application and the Verisure Server through a MITM attack. The attacker can also exploit vulnerabilities of the web application through a CSRF, cross-site request forgery, attack. |
| Countermeasures | Encrypt all communications. Use a standard authentication mechanism to identify the user. Use anti-CSRF tokens. |

Table 4: Threat #2

| | |
|---|---|
| Threat description | The attacker clones the Yale key tag. |
| Threat target | The lock. |
| Attack techniques | The attacker can with an NFC reader read and then clone the key tag. |
| Countermeasures | Encrypt the key information. Carry an anti-skimming shield card. |

Table 5: Threat #3

| | |
|---|---|
| Threat description | The attacker captures packets sent between the system components and analyses and modifies the packets before sending the modified packets to the desired destination. |
| Threat target | The Wi-Fi communication. |
| Attack techniques | Replay attacks and other forms of MITM attacks. |
| Countermeasures | Use one-time passwords (OTPs), one-time session keys and/or timestamps on the messages. |

Table 6: Threat #4

| | |
|---|---|
| Threat description | The attacker modifies the downloaded APK (Android Package Kit) files. |
| Threat target | The Android mobile application. |
| Attack techniques | Reverse engineering the files to perhaps find vulnerabilities to use and to modify the files. |
| Countermeasures | Obfuscate the code to make it more difficult to understand. |

Table 7: Threat #5

| | |
|---|---|
| Threat description | The attacker modifies the Yale key tag. The modification can lead to denial of service if an important part is changed or erased. |
| Threat target | The lock's key tag. |
| Attack techniques | The attacker can with an NFC reader overwrite chosen blocks of the key tag. |
| Countermeasures | Make the tag read-only. |

Table 8: Threat #6

| | |
|---|---|
| Threat description | Reverse engineering the android application code or files to get access to information not intended for disclosure. |
| Threat target | The Android mobile application. |
| Attack techniques | Reverse engineering. |
| Countermeasures | Obfuscate the code to make it more difficult to understand. |

Table 9: Threat #7

| | |
|---|---|
| Threat description | The attacker gains user credentials via the password reset function. |
| Threat target | The mobile and web application. |
| Attack techniques | The attacker can gain information with user enumeration and dictionary attacks. |
| Countermeasures | Follow OWASP's suggestions regarding the password reset function. |

Table 10: Threat #8

| | |
|---|---|
| Threat description | The attacker gains access to information by skimming the Yale key tag. |
| Threat target | The lock's key tag. |
| Attack techniques | The attacker can with an NFC reader read all the blocks in the key tag. |
| Countermeasures | Encrypt the key information. Carry an anti-skimming shield card. |

Table 11: Threat #9

| | |
|---|---|
| Threat description | The attacker brute forces passwords. |
| Threat target | The mobile application and web application. |
| Attack techniques | Brute forcing. |
| Countermeasures | Implement two-factor authentication. Implement CAPTCHA. |

Table 12: Threat #10

| | |
|---|---|
| Threat description | The attacker overloads the Verisure server and components by replaying packets many times fast. |
| Threat target | The Verisure server and components. |
| Attack techniques | Replay attacks. |
| Countermeasures | Use one-time passwords (OTPs), one-time session keys and/or timestamps on the messages. |

Table 13: Threat #11

| | |
|---|---|
| Threat description | The attacker performs a CSRF attack to add users or spoofs an existing user to gain privileges. |
| Threat target | The mobile application and web application. |
| Attack techniques | The attacker sends a CSRF request to the user that will automatically add a user. A MITM attack can be performed to spoof the identity of a user. |
| Countermeasures | Encrypt the traffic, use an authentication mechanism to identify the user. Use anti-CSRF tokens. |

## 4.5 Rating the threats

The threats are rated using Microsoft's rating table for DREAD, which is shown in figure 3. If a threat's total result is 5-7, it is considered to be of low risk, 8-11 is considered as medium risk and 12-15 as high risk [17].

Table 13: the rated threats using the DREAD model.

| Threat | D | R | E | A | D | Total score | Risk |
|---|---|---|---|---|---|---|---|
| #1 | 3 | 3 | 2 | 2 | 3 | 13 | High |
| #2 | 3 | 2 | 2 | 2 | 3 | 12 | High |
| #3 | 3 | 3 | 2 | 2 | 3 | 13 | High |
| #4 | 3 | 3 | 1 | 3 | 2 | 12 | High |
| #5 | 2 | 1 | 2 | 1 | 3 | 9 | Medium |
| #6 | 3 | 3 | 3 | 2 | 2 | 13 | High |
| #7 | 2 | 3 | 3 | 2 | 3 | 13 | High |
| #8 | 2 | 3 | 3 | 2 | 1 | 11 | Medium |
| #9 | 3 | 3 | 3 | 2 | 3 | 14 | High |
| #10 | 3 | 3 | 2 | 3 | 3 | 14 | High |
| #11 | 3 | 3 | 2 | 2 | 3 | 13 | High |

# 5. Penetration testing

In this chapter, all penetration tests and their results are presented and discussed. In each penetration test the necessary theory and appropriate tools are also explained.

## 5.1 Penetration test #1: Router MITM

In this test, the attempted method is spoofing as described in threat #1, with DREAD rating 13. If authentication information of a user such as login credentials is successfully sniffed, the attacker could spoof the identity of the user. To achieve this a MITM attack was performed on the router connected to the Verisure hub, V-Box Mini, via an ethernet cable as all the commands to and from Verisure's applications and Yale Doorman V2N go through the router, making the router a very convenient entry point to attack.

### 5.1.1 Background

Tcpdump [28] and Wireshark [29] are both powerful network analyzing tools. Tcpdump is, however, an CLI, command-line interface, tool while Wireshark is a GUI, graphical user interface, tool. Tcpdump and oftentimes Wireshark use the library libpcap, which was developed to enable packet capturing and analyzing a network. They also provide the means to filter traffic based on for instance protocols, ports and hosts.

### 5.1.2 Methodology

This penetration test followed the tutorial on NSE's web page [30]. Via "ssh root@x.x.x.x", where x.x.x.x is the IP address of the targeted router, tcpdump commands can be performed on said router to capture traffic passing through it. The data from the captured traffic is then piped into the hacker's computer for analysis with Wireshark.

### 5.1.3 Result

All the traffic is encrypted, therefore no passwords or other authentication information can be gained and no identity can be spoofed. However, other information was successfully sniffed, for instance the protocols and ports in use, the size and time interval of the packets sent/received and the different Verisure hosts handling the communication between the Verisure server and the Verisure components like the V-Box Mini and the mobile application/web application. The results found in this penetration test were updated in the assets table in the threat model, table 1 section 4.1. The V-Box Mini communicates via SCTP packets encapsulated in UDP, it connects to SCTP port 9899. The mobile application and web application communicates via HTTPS, port 443. This was also added in the data flow diagram found in section 4.2, figure 5.

### 5.1.4 Discussion

This penetration test was performed to see if all communications were encrypted and Verisure uses standard authentication mechanisms to identify the user or if the user could be spoofed. The result was satisfactory, as it showed that all communications were indeed encrypted, making the Verisure system secure against basic spoofing attacks.

## 5.2 Penetration test #2: DoS on the user's Verisure account

In this test, the attack vector is DoS through brute forcing passwords as described in threat #9. The idea is to deny the user access to their Verisure account either by managing to find the correct password and then changing the password and other user information/credentials such as phone number and email-address or by testing wrong passwords enough times to lock out the user. The latter can happen if Verisure has a countermeasure for brute forcing in the form of limited amount of login attempts in a certain timeframe.

### 5.2.1 Background

A brute force attack is when a hacker tries all possible password combinations. A dictionary attack on the other hand is a form of brute force attack where instead of trying all possible password combinations, a list of the most common passwords, a so called dictionary list, is used [31].

### 5.2.2 Methodology

On NordPass's web page [32] a list of the most common passwords of 2020 was found and used as a guide for this test. This list was iterated through from the top and filtered to accommodate Verisure's password policy. Next, the passwords, starting from the top of the list, were used to attempt to compromise a user's Verisure account.

### 5.2.3 Result

After four tries, an error message is given, stating: "Too many failed login attempts were recorded. Your account is temporarily locked for 15 minutes.", thus successfully denying the legitimate user from logging in. However, if the user already is logged in, they will continue to have access to their account, i.e. they will not be forcefully logged out. Nevertheless, in case of finding the correct password and gaining access to a user's account, it is still not possible to change any user credentials without an SMS code verification, as seen in figure 6 below.

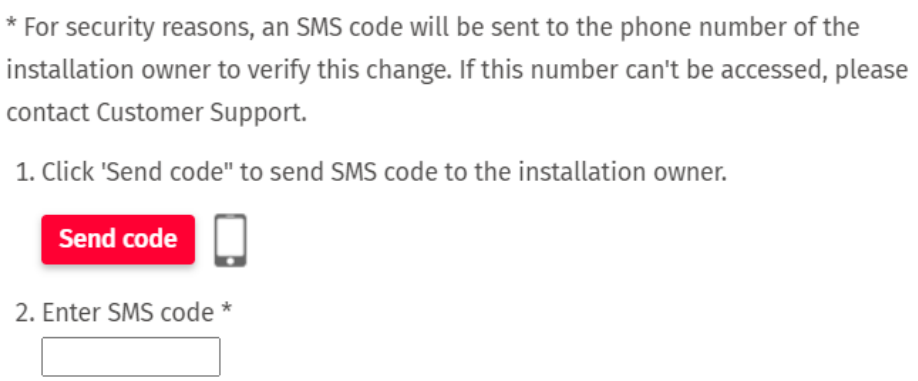


Figure 6, screenshot of Verisure's SMS verification requirement.

### 5.2.4 Discussion

Verisure does indeed have a countermeasure against brute force attacks by limiting the amount of allowed failed login attempts to four before locking out the user for 15 minutes. However, this countermeasure is not done properly, as it can lead to a successful DoS attack instead, denying the user access for 15 minutes each time. An improvement to this countermeasure, to avoid the user denial of service, can be to apply two-factor authentication [33]. Instead of completely locking out the user, Verisure can add the choice to unlock their account before the 15 minutes is up by an SMS verification for example. This way the legitimate user is not affected by the DoS attack.

Another common countermeasure against brute force attacks, more specifically dictionary attacks, is good password policies [33]. Weak, guessable or hardcoded passwords are the number one IoT threat according to OWASP [25]. They suggest that at least 3 out of 4 complexity rules are met, which are [34]:

- at least 1 uppercase character (A-Z)
- at least 1 lowercase character (a-z)
- at least 1 digit (0-9)
- at least 1 special character

Verisure does not follow this suggestion regarding password change via “My Pages”. As seen in figure 7, they only require two of the abovementioned complexity rules, making it possible to have weak passwords such as “Password” or “password1”.

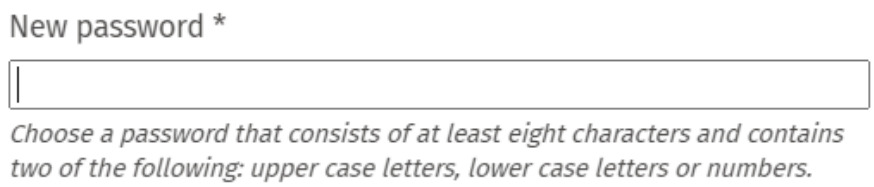


Figure 7, screenshot of the requirements regarding password change in “My Pages”.

Consequently, if a user chooses a weak password, it can easily be brute forced. For example, even with the 15 minutes restriction after four failed attempts, a dictionary attack using the “John the Ripper” dictionary list in Kali Linux would require less than 73 hours to find the password “Password”, as it is ranked 1167$^{th}$. However, “password1”, which is in 4$^{th}$ place, would take less than a second to crack. Other countermeasures for brute force attacks are having a maximum number of failed attempts before blocking the user’s account or using techniques like CAPTCHA to ensure it is a human user [35]. However, a proper brute forcing could not be performed due to legal reasons. More on this can be read in chapter 6.

## 5.3 Penetration test #3: Information disclosure via the password reset function

In this test the attack vector is information disclosure, as described in threat #7, can information about a user, such as a username, be disclosed to an unauthorized person via Verisure’s password reset function?

### 5.3.1 Methodology

A password reset was requested from Verisure’s login page, and the instructions on the password reset page were followed.

### 5.3.2 Result

Verisure requires both the correct email address and the correct phone number when requesting a password reset. Figure 8 shows a screenshot of Verisure’s error message, which does not disclose any information. They do not confirm the email address or the phone number. Figure 9 shows the procedure to create a new password and verify it, which is the next step after entering the correct email address and phone number. After the password is successfully reset, a confirmation is sent to the provided email address.

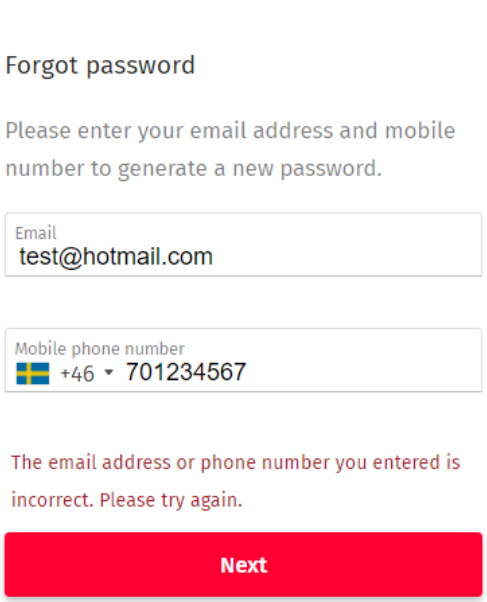


Figure 8, The password reset page

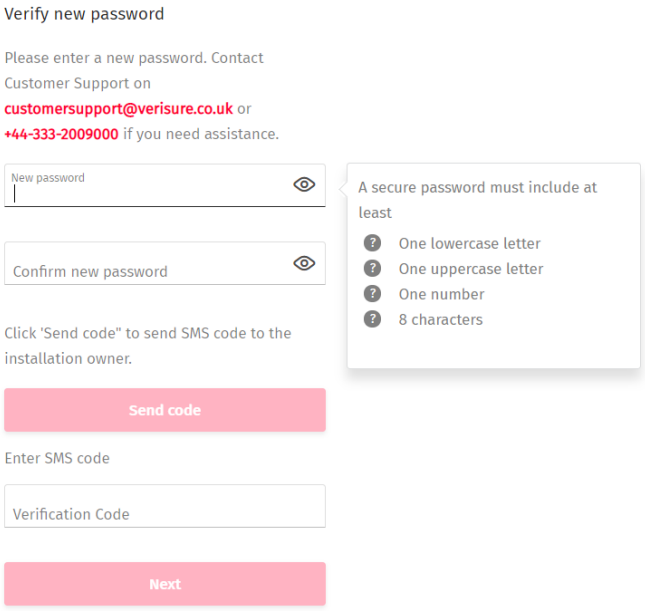


Figure 9, Creating and verifying a new password.

### 5.3.3 Discussion

From a security perspective, it is important to not reveal any kind of user information during the reset process, which are usually obtained from error messages [33]. Some examples of revealing error messages and consequently bad security practices are “Incorrect email address” or “Incorrect phone number”, as they reveal that the other is correct. However, Verisure’s error message does not reveal anything.

OWASP also emphasizes the importance of secure and consistent password policies. Unfortunately, Verisure is not consistent with their password policies, as seen in figure 8 they have four conditions that must be met when choosing a new password. They have the same requirements when creating a new account. However, they have one less condition when changing the password on “My Pages”, as seen in figure 7.

Other important measures include sending a confirmation without revealing the new password and automatically log out the user on all devices. The usage of CAPTCHA is also a good measure against user enumerations [36]. Verisure does not use CAPTCHA but they do send a confirmation email on the password change, as well as log out the user.

## 5.4 Penetration test #4: MITM with mitmproxy

The attack vector in this test is the same as in penetration test #1, which is spoofing and information disclosure. The difference is the methodology used, even though both tests are a MITM attack, they use different tools and target different entry points.

### 5.4.1 Background

HTTP is the application layer protocol used to load web pages. HTTP uses port 80 for communication over the web. The problem with HTTP is the lack of security measures such as encrypted communication, which enables traffic interception. It is therefore not recommended to use. HTTPS on the other hand is secure, as all traffic between the web server and web browser is encrypted with SSL or TLS [37].

Mitmproxy is an interactive HTTPS proxy, which is used to bypass the SSL/TLS encryption [38].

ARPspoof is a tool preinstalled in Kali Linux used for arpspoofing/arppoisoning. Arpspoofing/arppoisoning is when the hacker channels all the traffic through their machine by fooling the targeted devices on the LAN to believe the hacker's machine has the legitimate recipient's IP address. For instance, the hacker can arpspoof the traffic between device A and B, by telling device A they have the IP address of device B and vice versa, hence since device A believes the hacker is device B, it will send the packets destinated to device B, to the hacker's machine instead, and vice versa.

### 5.4.2 Methodology

The tutorials followed for this test are from mitmproxy's web page [39] and Valbrux's blog [40]. Mitmproxy was used to bypass the HTTPS encrypted traffic. In order for mitmproxy to work and be able to decrypt the encrypted traffic, a mitmproxy certificate must be installed on the targeted device and the correct proxy settings has to be configured [41]. However, the latter can be skipped by using ARPspoof instead. ARPspoof is done between the targeted device and the default gateway, i.e. the router.

### 5.4.3 Result

All traffic between the targeted device and the router was successfully decrypted and sniffed. Consequently, all confidential data such as usernames, passwords and user codes can be seen in plaintext, as seen in figure 10.

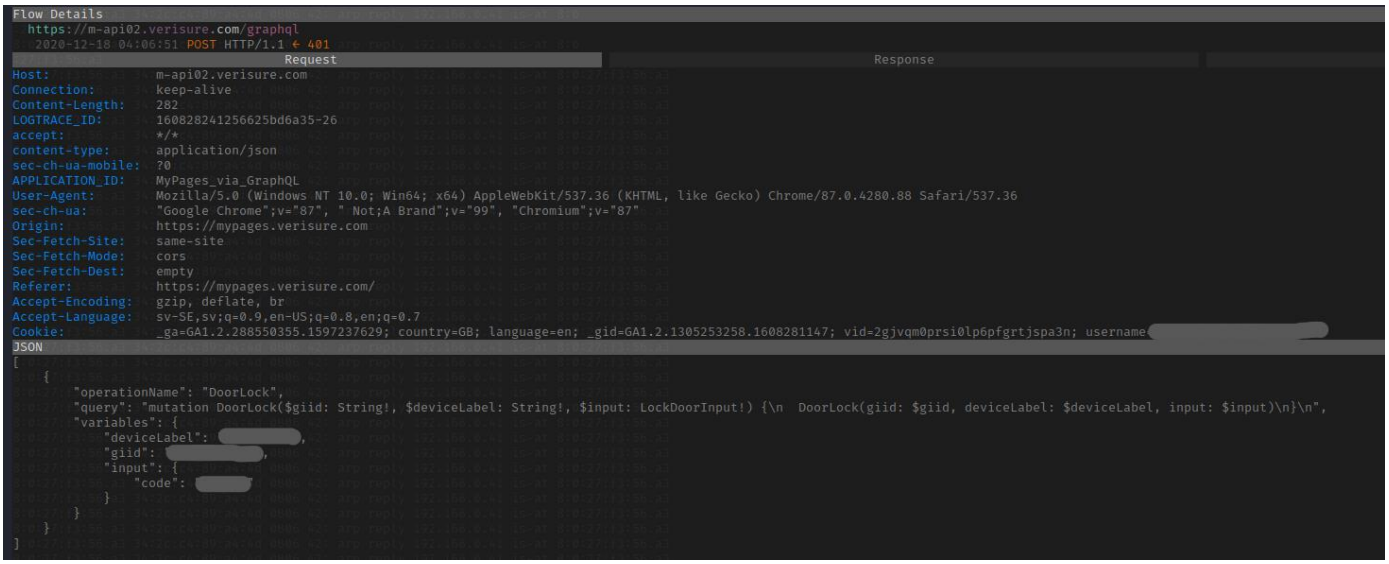


Figure 10, screenshot of an HTTPS request in mitmproxy, with some sensitive information crossed out.

### 5.4.4 Discussion

The result was as expected, as there are many previous works and tutorials on mitmproxy. But for this MITM attack to work, the mitmproxy CA certificate has to be installed on the targeted device, this attack is thus only possible if the hacker has access to the targeted device. However, they only need access for a very short period of time, within minutes the certificate can be installed and hidden. This attack shows that even HTTPS is not completely safe from hackers.

## 5.5 Penetration test #5: Skimming and cloning the Yale Doorman V2N key tag

In this test the attack vectors are information disclosure and spoofing as described in threats #7 and #2. Yale Doorman V2N uses Mifare Classic 1K tags for locking and unlocking the smart lock. Mifare Classic 1K cards are vulnerable to skimming, cloning and modifications and are therefore not

considered secure anymore [42]. Skimming is the act of illegally collecting the data from an NFC/RFID card/tag [43].

### 5.5.1 Background

Mifare Classic 1K tags/cards uses NFC, Near Field Communication, technique with radio frequency of 13,56 MHz. The 1K in the name indicates the size of the memory. A 1K tag/card has 16 sectors, with 4 blocks, containing 16 bytes of data, each. In NFC, as indicated by the name, the NFC reader and tag/card must be in close proximity to each other, in theory they can operate in a distance up to 10 cm [44]. However, as demonstrated by Kevin Mitnick [45], the range can be increased to up to 0,91 m. NFC is widely used in today's society, for example, it is used in public transportation, in universities and in credit cards. There is an active NFC device, called the reader, that scans for passive devices, the NFC cards/tags. If the cards/tags are closer than 10 cm, the reader establishes a connection with them, enabling transmission of data. One of the most commonly used NFC techniques is the Mifare Classic. Mifare Classic uses their own encryption protocol, called Crypto-1 [44]. This encryption protocol is according to researchers not secure, as it can easily be cracked [42].

Each sector of a Mifare Classic 1K tag/card contains 2 access keys, Key A and Key B. These keys define the access rights of each sector. Moreover, Block 0 in Sector 0, called the manufacturer block, contains the unique ID of the tag, the UID, and data about the manufacturer [44]. In order to clone successfully**,** a special NFC card called Chinese Magic Card or generation 1 Mifare Classic 1K card, is required. What makes this specific card special is that the UID, which is usually non-writable, is in this card writable [46, 47].

### 5.5.2 Methodology

For this test Tim Theeuwes's [48] tutorial was followed. To read the tag, a Samsung phone was used, as they have an inbuilt NFC reader already. In the settings, the phone's NFC reader was turned on. Then, MIFARE Classic Tool application was downloaded from Google Play. With this application, one can easily read, write, save and clone the tags. The Yale Doorman V2N key tag was read, saved and then attempted to be cloned to another Mifare Classic 1K card.

### 5.5.3 Result

It was possible to read the Yale Doorman V2N key tag. In figure 11 below, the first five sectors of the key tag can be seen. Key A is hidden with hyphens. Therefore, the attempt to clone the tag was not successful, as seen in figure 12, where the application states "Blocks containing unknown data (---) will be skipped!".


Figure 11, screenshot of the Yale Doorman V2N key tag dump from Mifare Classic Tool. The UID is crossed over for security reasons.

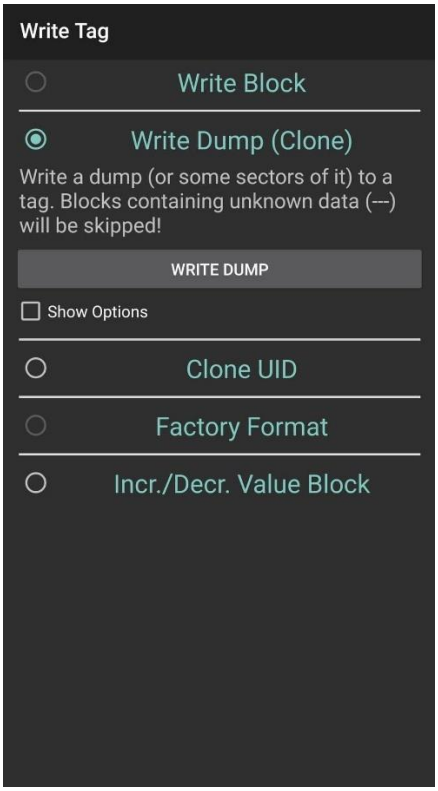


Figure 12, screenshot of the Write Tag page from Mifare Classic Tool.

### 5.5.4 Discussion

The skimming, reading and saving the data from the Yale Doorman V2N key tag, was successful, but the cloning was not. However, this does not indicate that Yale's Mifare Classic 1K card is secure against cloning. It only indicates that the key tag is safe against cloning done via the Android application Mifare Classic Tool.

On the contrary, with the right tools, cloning of the Yale Doorman V2N key tag can be achieved. The access keys can be cracked as described by Mehlmauer [42]. Furthermore, as stated by Chipster [46], a high frequency NFC reader, Proxmark3 or ACR122U, and its corresponding open source software, can be used to crack and bypass the available security mechanisms. However, they also state that cloning the Yale Doorman V2N key tag can only be done once due to some unknown security measurements established by Yale.

## 5.6 Penetration test #6: Reverse engineering the APK files

In this test the attack vector is information disclosure, by reverse engineering the APK (Android Package Kit) files from the android application, as seen in threat #6. Information that could be disclosed are hardcoded passwords, usernames, API keys and other sensitive information.

### 5.6.1 Background

MobSF [49], Mobile Security Framework, is an all-in-one, open source, automated penetration testing framework for mobile applications. MobSF consists of a collection of tools that runs in the background and presents the results in a graphical interface.

### 5.6.2 Methodology

The app “Apk Extractor” is downloaded from Google Play to extract the APK file from the Verisure android application [50]. MobSF is then used to decompile and analyze the APK file [51]. The result of the analysis contains potential vulnerabilities, which are reported in different categories and graded by severity.

### 5.6.3 Result

The code is obfuscated to make it more difficult to understand. However, MobSF still reported 68 potential vulnerabilities, of which 13 was from the code analysis category. The only new and alarming one, is the report of hardcoded sensitive information in two files, as seen in figure 13 below. However, when looking into these two files, no hardcoded sensitive information could be found. Figure 14 and 15 shows what MobSF was complaining about in yellow highlight, it is not hardcoded sensitive information, it is only hardcoded String constants.

| | | | | |
|---|---|---|---|---|
| 8 | Files may contain hardcoded sensitive informations like usernames, passwords, keys etc. | high | **CVSS V2:** 7.4 (high)<br>**CWE:** CWE-312 Cleartext Storage of Sensitive Information<br>**OWASP Top 10:** M9: Reverse Engineering<br>**OWASP MASVS:** MSTG-STORAGE-14 | j/m/b/a/i/t/a.java<br>i/b/p/i/g.java |

Figure 13, screenshot of a vulnerability with a high severity found by MobSF. The last column shows in which files this vulnerability is found in.

a.java

```
1.  package j.m.b.a.i.t;
2.
3.  import android.os.Bundle;
4.  import j.m.b.a.i.g;
5.  import java.net.URLEncoder;
6.  import java.util.ArrayList;
7.  import java.util.HashMap;
8.  import java.util.List;
9.  import java.util.Map;
10.
11. public class a {
12.     public static final String PARAM_BODY = "body";
13.     public static final String PARAM_COMMAND = "command";
14.     public static final String PARAM_HEADER_PARAMS_KEY_ARRAY = "header_keys";
15.     public static final String PARAM_HEADER_PARAMS_VALUE_ARRAY = "header_values";
16.     public static final String PARAM_PASSWORD = "password";
17.     public static final String PARAM_URL_PARAMS = "url_parameters";
18.     public static final String PARAM_USERNAME = "username";
19.     public String body;
20.     public HashMap<String, String> headerParameters = new HashMap<>();
21.     public String password;
22.     public EnumC0270a requestType;
23.     public g<String, String> urlParameters = new g<>();
24.     public String userName;
```

Figure 14, a screenshot of the first file reported by MobSF for containing hardcoded sensitive information.

```
30.
31.  public class g implements i.i.i.a.a {
32.      public static final String ACTION_VIEW_STATES_KEY = "android:menu:actionviewstates";
33.      public static final String EXPANDED_ACTION_VIEW_ID = "android:menu:expandedactionview";
34.      public static final String PRESENTER_KEY = "android:menu:presenters";
35.      public static final String TAG = "MenuBuilder";
36.      public static final int[] sCategoryToOrder = {1, 4, 5, 3, 2, 0};
```

Figure 15, a screenshot of the second file reported by MobSF for containing hardcoded sensitive information.

### 5.6.4 Discussion

The android application contains a large number of files, which requires a considerable amount of time to analyze. Therefore, MobSF is used to automate this process. However, the source code is obfuscated, making parts of it unreadable and thus difficult to analyze. In the allotted time, it was not possible to analyze the whole source code, but the analysis done, lead by MobSF, gave the conclusion that the source code was not disclosing any sensitive information.

# 6. Result

A few weaknesses are found during the penetration testing phase. Details on each penetration test can be found in chapter 5. The first weakness found is the possibility of locking out the user from their account and therefore deny them that service for 15 minutes. If the user is already logged in, they will not be affected by this DoS attack on their Verisure account. However, if the user wishes to log in to their account during these 15 minutes, they will be affected. The second weakness found is Verisure's password policy, more specifically the weaker policy during the password change from Verisure's "MyPages". Here, they are not following OWASP's [34] recommendations. The third weakness found is the possibility to gain access to sensitive information by decrypting the traffic with mitmproxy. This weakness, however, requires client side manipulation. The last weakness is found during the fifth penetration test, performed on the Yale Doorman V2N key tag. It was possible to read and save the information on the key tag via the application Mifare Classic Tool. However, more complex and professional tools are required for cloning the key tag. Another weakness discovered is that the mobile application always stays logged in unless the password is changed.

During penetration tests #1 and #6 no vulnerabilities or weaknesses are found. All HTTPS traffic is encrypted, and no sensitive information is disclosed in the source code of the Android application.

# 7. Discussion

The penetration tests detailed in chapter 5 are repeated several times, and each time the same results were obtained. This indicates that the results are reliable. The results show that there are some weaknesses in the Verisure smart lock system and however small they may be, they still need to be taken into consideration.

Verisure can easily implement countermeasures for almost all the weaknesses presented in chapter 7. For the DoS attack on the user's Verisure account, Verisure can apply two-factor authentication and add the option for the user to unlock their account when they are denied access after four failed attempts by an SMS or email verification. They can adjust their password policy to follow OWASP's suggestions and be consistent. They can add the requirement to have a pin code for every key tag activated and thus deny the attacker access to the users' home, even after a successful cloning attack on the key tags, as the pin code is also needed.

Lastly, they can implement an automatic log out function on the mobile and web application after a certain amount of time has passed. At the moment, the user is never automatically logged out from the application, with the exception of a password change. This can be problematic provided that the user has their lock screen security turned off on their mobile phone and they lose it or some malicious person gets access to their phone for a short period. In this case, the attacker gains complete control of the system under consideration, i.e. they can change and configure anything they desire, since all verifications occur via SMS. However, if the lost phone is not the registered phone number for the SMS verifications, the attacker does not gain full access to the system, but instead obtains sensitive information such as the username and log. Verisure does warn the user about this risk, as seen in figure 16, however, nothing more is done and the responsibility is instead simply put on the user. On the other hand, one can argue that for Verisure this is the desired feature, designed to make the application more user friendly. Moreover, they might also have done a risk assessment and come to the conclusion that the risk is too low, so low, in fact, that a warning is enough.


Figure 16, screenshot of the warning message displayed in Verisure's mobile application. The message translates to "To protect your phone from unauthorized use, we recommend that you enable screen lock (PIN/password)".

# 8. Sustainability and ethics

Everybody has a moral obligation to work as sustainable as possible, thus it is also considered in this project. The system under consideration was bought specifically for this project, during which the system was not damaged in any way. As a result, the system can be reused by other students or sold to an interested buyer, consequently decreasing the effect on the economic sustainability. Since it is not wasted, the only significant impact this project has on the environment is its small consumption of electricity.

One way to preserve and widen the general knowledge is to publish the research, and together advance as a society through open data and transparency, leading to a more secure smart lock, and other IoT products, in the market. However, the potential vulnerabilities found, if published, may result in everything from catastrophic consequences to slight discomfort for the companies involved. Therefore, whenever the research involves ethical hacking, ethical considerations must be discussed. In order to successfully conduct penetration tests, one must ask oneself if the tests performed are permitted or/and if they can somehow harm the companies or the general public. One must also ask how the vulnerabilities should be presented, fully or responsibly disclosed? Responsible disclosure means that the vulnerabilities found will first be reported to the companies so they may patch it within the agreed timeframe before it is disclosed to the public. Fully disclosure on the other hand means that the vulnerabilities are published directly since an agreement with the companies has already been made.

The law is part of the ethical consideration that must be discussed, more specifically, in Sweden, ”Brottsbalken 4 kap. 9c §” Lag (2014:302) [7] has to be followed. The law states that you are not allowed to hack into someone else’s property. In this project the system under consideration was bought and is therefore the property of the researcher during the penetration testing. The testing was done only on these owned items and there were no attempts to hack into Verisure’s servers.

# 9. Conclusion

From the penetration tests performed, one can conclude that the system under consideration is relatively secure. There are a few weaknesses that could be targeted. In fact, even a novice hacker could successfully perform the penetration tests and do some damage. However, that outcome is unlikely and some weaknesses also require client side manipulation. Nonetheless, if Verisure implements some countermeasures, such as those mentioned in chapter 7, the system becomes more secure.

## 9.1 Future work

Within the allotted timeframe, a complete security evaluation could unfortunately not be done. Due to the time constraint some components of the system under consideration are out of scope in this project and some threats could not be tested, such as the CSRF threats. These can, however, be further investigated in future work. Some penetration tests performed in this project could also be investigated further, they could be dug deeper with better tools for instance.

# References


[1] Check Point Software Technologies Ltd. Cyber Security Report 2020 [Internet]. Check Point Software Technologies Ltd.; 2020. [cited 2020 Dec 21]. Available from: https://www.ntsc.org/assets/pdfs/cyber-security-report-2020.pdf

[2] Marciniak K. Digital lockpicking - stealing keys to the kingdom [Internet]. F-Secure; 2019. [cited 2021 Jan 29]. Available from: https://labs.f-secure.com/blog/digital-lockpicking-stealing-keys-to-the-kingdom

[3] Brewster T. A Basic Z-Wave Hack Exposes Up To 100 Million Smart Home Devices. Forbes [Internet]. 2018 May 24 [cited 2021 Jan 29]; Available from: https://www.forbes.com/sites/thomasbrewster/2018/05/24/z-wave-hack-threatens-to-expose-100-million-smart-homes/#73d8911e4517

[4] Guzman A, Gupta A. IoT Penetration Testing Cookbook [Internet]. 1st edition. Birmingham: Packt Publishing; 2017. [cited 2020 Nov 17]. Available from: https://learning.oreilly.com/library/view/iot-penetration-testing/9781787280571/

[5] Verisure. Digitalt Dörrlås [Internet]. Verisure. [cited 2020 Nov 10]. Available from: https://www.verisure.se/hemlarm/produkter/digitalt-dorrlas.html

[6] Microsoft. Threat Modeling Security Fundamentals [Internet]. Microsoft. [cited 2020 Nov 10]. Available from: https://docs.microsoft.com/sv-se/learn/paths/tm-threat-modeling-fundamentals/

[7] Brottsbalk (1962:700). Riksdagen. [cited 2020 Dec 10]. Available from: https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/brottsbalk-1962700_sfs-1962-700

[8] Tarandach I, Coles M. Threat Modeling [Internet]. 1st edition. O'Reilly Media, Inc.; 2020. [cited 2020 Nov 16]. Available from: https://learning.oreilly.com/library/view/threat-modeling/9781492056546/

[9] Shevchenko N, Chick TA, O'Riordan P, Scanlon TP, Woody C. THREAT MODELING: A SUMMARY OF AVAILABLE METHODS [Internet]. Carnegie Mellon University; 2018. [cited 2020 Nov 18]. Available from: Threat Modeling: A Summary of Available Methods (cmu.edu)

[10] Synopsys. Ethical Hacking [Internet]. Synopsys. [cited 2020 Dec 19]. Available from: https://www.synopsys.com/glossary/what-is-ethical-hacking.html

[11] Viderberg A. Security evaluation of smart door locks. [Master's thesis]. Stockholm: KTH, School of Electrical Engineering and Computer Science (EECS); 2019. [cited 2020 Dec 22].

[12] Toft J, Robberts C. Finding Vulnerabilities in IoT Devices: Ethical Hacking of Electronic Locks. [Bachelor's thesis]. Stockholm: KTH, School of Electrical Engineering and Computer Science (EECS); 2019. [cited 2020 Dec 22].

[13] Låskompaniet. Smart lås sats Verisure (smartlock) [Internet]. Låskompaniet. [cited 2020 Nov 1]. Available from: https://www.laskompaniet.se/product/smart-las-sats-verisure-smartlock-

[14] Bauhaus. LÅSSATS YALE DOORMAN V2N SVART BUNDLE [Internet]. Bauhaus. [cited 2020 Nov 1]. Available from: https://www.bauhaus.se/yale-doorman-lassats-v2n-svart-bundle#go-to-attributes

[15] Verisure. VANLIGA FRÅGOR OM YALE DOORMAN [Internet]. Verisure. [cited 2020 Nov 1]. Available from: https://www.verisure.se/kontakt/yale-doorman.html

[16] Microsoft. Threat modeling for drivers [Internet]. Microsoft; 2018. [cited 2020 Nov 21]. Available from: https://docs.microsoft.com/en-us/windows-hardware/drivers/driversecurity/threat-modeling-for-drivers

[17] ThreatModeler. Stride, VAST, Trike, & More: Which Threat Modeling Methodology is Right For Your Organization? [Internet]. ThreatModeler; 2019. [cited 2020 Nov 21]. Available from: https://threatmodeler.com/threat-modeling-methodologies-overview-for-your-business/

[18] OWASP. Threat Modeling [Internet]. OWASP. [cited 2021 Feb 2]. Available from: https://owasp.org/www-community/Threat_Modeling

[19] Microsoft. Chapter 3 – Threat Modeling [Internet]. Microsoft; 2010. [cited 2020 Nov 20]. Available from: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff648644(v=pandp.10)

[20] Microsoft. The STRIDE Threat Model [Internet]. Microsoft; 2009. [cited 2020 Nov 22]. Available from: https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)

[21] g0tmi1k. What is Kali Linux? [Internet]. Kali. [cited 2020 Nov 24]. Available from: https://www.kali.org/docs/introduction/what-is-kali-linux/

[22] Yale. Installation Guide User Manual [Internet]. [cited 2020 Nov 28]. Available from: https://www.yalehome.se/Yale/YalehomeSE/Filer/manualer/Yale%20Doorman%20V2N/YaleDoorman-ManualV2N.pdf

[23] Låskompaniet. Verisure V-box Microenhet [Internet]. [cited 2020 Nov 28]. Available from: https://www.laskompaniet.se/product/verisure-v-box-microenhet

[24] Shostack A. Threat Modeling: Designing for Security [Internet]. 1st edition. Indianapolis: Wiley; 2014. [cited 2020 Dec 5]. Available from: https://learning.oreilly.com/library/view/threat-modeling-designing/9781118810057/

[25] The OWASP IoT Security Team. Top 10 IoT 2018. OWASP; 2018. [cited 2020 Dec 5]. Available from: https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf

[26] OWASP. OWASP Top Ten [Internet]. OWASP. [cited 2020 Dec 5]. Available from: https://owasp.org/www-project-top-ten/

[27] OWASP. OWASP Mobile Top 10 [Internet]. OWASP; 2016. [cited 2020 Dec 5]. Available from: https://owasp.org/www-project-mobile-top-10/

[28] man2html. Man page of TCPDUMP [Internet]. Tcpdump; 2020. [cited 2021 Jan 2]. Available from: https://www.tcpdump.org/manpages/tcpdump.1.html

[29] Wireshark. About Wireshark [Internet]. Wireshark. [cited 2021 Jan 2]. Available from: https://www.wireshark.org/

[30] NSE Lab. Wi-Fi Router MITM [Internet]. NSE Lab. [cited 2021 Jan 2]. Available from: https://nse.digital/pages/guides/wifi-mitm

[31] Diogenes Y, Ozkaya E. Cybersecurity - Attack and Defense Strategies [Internet]. 1st edition. Birmingham: Packt Publishing; 2018. [cited 2021 Jan 5]. Available from: https://learning.oreilly.com/library/view/cybersecurity-attack/9781788475297/

[32] NordPass. Top 200 most common passwords of the year 2020 [Internet]. NordPass. [cited 2021 Jan 5]. Available from: https://nordpass.com/most-common-passwords-list/

[33] OWASP. Authentication Cheat Sheet [Internet]. OWASP; 2021. [cited 2021 Jan 6]. Available from:https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Authentication_Cheat_Sheet.md#prevent-brute-force-attacks.

[34] Keary E, Manico J, Goosen T, Krawczyk P, Neuhaus S, Morales MA. Authentication [Internet]. OWASP. [cited 2021 Jan 6]. Available from: https://owasp.deteact.com/cheat/cheatsheets/Authentication_Cheat_Sheet.html

[35] Sentor. Vad är en brute-force-attack? [Internet]. Sentor. [cited 2021 Jan 6]. Available from: https://www.sentor.se/kunskapsbank-it-sakerhet/losenord/vad-ar-en-brute-force-attack/

[36] OWASP. Forgot Password Cheat Sheet [Internet]. OWASP; 2020. [cited 2021 Jan 8]. Available from:https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Forgot_Password_Cheat_Sheet.md

[37] Stashchuk B. SSL Complete Guide - HTTP to HTTPS [Internet]. 1st edition. 2019. [cited 2021 Jan 8]. Available from: https://learning.oreilly.com/videos/ssl-complete-guide/9781839211508

[38] mitmproxy. Introduction [Internet]. Mitmproxy. [cited 2021 Jan 8]. Available from: https://docs.mitmproxy.org/stable/

[39] mitmproxy. Transparent Proxying [Internet]. Mitmproxy. [cited 2021 Jan 9]. Available from: https://docs.mitmproxy.org/stable/howto-transparent/

[40] Valbrux. MITM using arpspoof + Burp or mitmproxy on Kali Linux [Internet]. Valbrux; 2017. [cited 2021 Jan 9]. Available from: https://www.valbrux.it/blog/2017/11/26/mitm-using-arpspoof-burp-or-mitmproxy-on-kali-linux/

[41] mitmproxy. About Certificates [Internet]. Mitmproxy. [cited 2021 Jan 9]. Available from: https://docs.mitmproxy.org/stable/concepts-certificates/

[42] Mehlmauer C. How to Crack Mifare Classic Cards [Internet]. Firefart; 2015. [cited 2021 Jan 14]. Available from: https://firefart.at/post/how-to-crack-mifare-classic-cards/

[43] Jilkén O. Säkerhet och integritet i Närfältskommunikation [Bachelor's thesis]. Karlskrona: Blekinge Institute of Technology; 2014. [cited 2021 Jan 14].

[44] NXP. MF1S50YYX_V1 MIFARE Classic EV1 1K - Mainstream contactless smart card IC for fast and easy solution development [Internet]. NXP;2018. [cited 2021 Jan 14]. Available from: https://www.nxp.com/docs/en/data-sheet/MF1S50YYX_V1.pdf

[45] Cyber Investing Summit. World's Most Famous Hacker Kevin Mitnick & KnowBe4's Stu Sjouwerman Opening Keynote [Video file]. 2017, June 5 [cited 2021 Jan 14]. Available from: https://www.youtube.com/watch?v=iFGve5MUUnE

[46] Chipster. Om att kopiera Yale Doorman V2N-taggar [Internet]. Chipster; 2019. [cited 2021 Jan 14]. Available from: https://chipster.se/om-att-kopiera-yale-doorman-v2n-taggar/

[47] Chipster. Mifare Classic 1k-tagg (Generation 1) [Internet]. Chipster. [cited 2021 Jan 14]. Available from: https://chipster.se/produkt/mifare-classic-1k-tagg-generation-1/

[48] Theeuwes T. Using a mobile phone to clone a MIFARE card [Internet]. Timdows; 2016. [cited 2021 Jan 15]. Available from: https://timdows.com/projects/using-a-mobile-phone-to-clone-a-mifare-card/

[49] Abraham A, Magaofei, Dobrushin M, Nadal V. Getting started [Internet]. MobSF. [cited 2021 Jan 18]. Available from: https://mobsf.github.io/docs/#/

[50] Dwarkani V. How To Reverse Engineer An Android Application In 3 Easy Steps [Internet]. Medium; 2020. [cited 2021 Jan 18]. Available from: https://medium.com/dwarsoft/how-to-reverse-engineer-an-android-application-in-3-easy-steps-dwarsoft-mobile-880d268bdc90

[51] Renato P. How to Extract an API Key from a Mobile App by Static Binary Analysis [Internet]. Approov; 2019. [cited 2021 Jan 18]. Available from: https://blog.approov.io/how-to-extract-an-api-key-from-a-mobile-app-with-static-binary-analysis

TRITA -CBH-GRU-2020:292