EXAMENSARBETE INOM INFORMATIONSTEKNIK,
AVANCERAD NIVÅ, 30 HP

STOCKHOLM, SVERIGE 2019

Security evaluation of smart door locks

ARVID VIDERBERG

KTH
SKOLAN FÖR ELEKTROTEKNIK OCH DATAVETENSKAP

Security evaluation of smart door locks

ARVID VIDERBERG

Master in Computer Science
Date: June 17, 2019
Supervisor: Pontus Johnson
Examiner: Robert Lagerström
School of Electrical Engineering and Computer Science
Host company: Subset AB
Swedish title: Säkerhetsutvärdering av smarta dörrlås

Abstract

Smart locks are a part of the up and rising Internet of Things (IoT). They're used as a complement to traditional locks in order to make it easier to share keys electronically, rather than physically. Smart door locks are mounted on a existing door and enables users to control the state of the lock with their smartphone. In this report we want to investigate if previous acknowledged attacks on smart locks still exists in today's available locks for the Swedish market. By studying previous work and conduct a threat model, we create attacks to apply on the locks in order to answer what deficiencies exist in smart door locks today.

The results conclude that there are several deficiencies in today's smart locks, where the most severe has been reported under responsible disclosure to the manufacturer. The locks investigated shows vulnerabilities in state consistency, password policies and password reset mechanism.

Sammanfattning

Smarta lås är en del av sakernas internet (IoT). Smarta lås används som ett komplement till traditionella lås för att göra det enklare att dela nycklar elektroniskt. Smarta dörrlås monteras på befintlig dörr och gör det möjligt för användare att styra låset med sin smartphone. I den här rapporten vill vi undersöka om tidigare kända attacker på smarta lås fortfarande finns i dagens tillgängliga lås på svenska marknaden. Genom att studera tidigare arbete och genomföra en hotmodell skapar vi attacker för att applicera på låsen, för att kunna svara på vilka brister som finns i smarta dörrlås idag.

Resultatet drar slutsatsen att det finns flera brister i dagens smarta lås, där de allvarligaste har rapporterats till tillverkaren. De undersökta låsen visar sårbarhet i konsistens mellan tillstånd, lösenordspolicys och lösenordsåterställning.

Contents

Chapter 1

Introduction

Smart home is a concept within *Internet of Things* (IoT), where the general purpose is to automate and control ubiquitous systems in a household. The technology for achieving a smart home has been available since the 1970s via technologies such as motion sensors, video cameras and programmatic lightning[1]. Today however, the technology has evolved and everyday household products like toasters, pre-amps, and weather stations are connected with each other. When combined together, they are used to make life more comfortable by providing context-aware services, which often can be controlled remotely[2]. Smart home devices are growing fast, as the estimated revenue growth of sold smart home appliances are predicted from $40m in 2012 to $26bn in 2019[3].

One category of smart home devices is smart door locks, which are locks that are mounted on an existing door and then connects to a smartphone via Bluetooth or WiFi. The user configures the device, and then controls whether the door is locked or not from their smartphone. If the lock is connected with WiFi, it's also possible to remotely control the lock even if the user isn't physically nearby the lock. These smart locks often offer services like temporary or scheduled based access, which can be handy if neighbours are bringing in mail, or a craftsman doing a job during office hours when the owner isn't at home.

1.1 Background

As the rate of smart homes devices increases, so does the complexity. In research done by Mennicken, Vermeulen, and Huang [4], wireless networks and increasing amount of interaction between devices has increased the complexity of maintenance and security. Their work shows that consumers use smart homes to provide “peace of mind".

On the other hand, during studies of social obstacles when implementing smart homes, one concern for customers was to keep their system secure[1, 5]. For instance, a study conducted by Brush et al. [1] of households that use home automation in their everyday life showed that when it came to security and remote access, the households in general thought remote access was a good idea. However, 7 out of 14 households raised concerns regarding the security of remote access. They were most concerned about remote access of door locks and cameras.

Data leaks or breaches in smart homes raises concerns regarding privacy and security. Smart door locks is no exception, with the main security risk of leaving your home inattentively unlocked. Because of this, there is a need to do more research regarding security in smart door locks.

1.2 Thesis objective

To address the need from the previous section, this thesis will investigate smart door locks from a security perspective. The work aims to compile a list of common attacks on smart locks and similar IoT devices, and then apply them to a set of smart locks. This should answer the research question: *Do previous discovered vulnerabilities in smart door locks exist in smart door locks today?* By doing so, we can promote secure development of IoT products and highlight the problems that exist today.

1.3 Delimitations

The number of smart locks that should be tested has been limited to two different devices, based on the time scope of the degree project. The smart locks have been limited to the following criterion’s:

- They should be able to order to Sweden.
- Since doors have different physical design in different countries, and the thesis work is carried out in Sweden, the locks should be adapted for Scandinavian doors.
- The locks should be able to be remotely accessed from the internet.

The thesis work does not intend to investigate other wireless management techniques on the door locks, such as Bluetooth or NFC. It will not investigate vulnerabilities in any of these technologies, but instead focus on the part of the technology that is available over the internet. This choice is based on the fact that the technology of remote controlling smart locks is new and has less available research in comparison to NFC and Bluetooth.

1.4 Choice of methodology

The first stage of the project uses a state-of-the-art evaluation of current research within known attacks on smart locks and IoT devices. We use a threat model to identify sensitive assets, attack surfaces, and threats in the smart locks. Combined with the preliminary study of previous attacks, attacks are conducted on the selected locks. An iterative approach is used to apply the attacks on each and every of the selected smart locks. To ensure scientific validity, the work is evaluated by using a quantitative evaluation method to determine whether the attacks were successfully executed or not.

The process used for threat modelling is created by Clinton and Marisetty [6] as an recommendation for developers of IoT products. It's based on the well-known STRIDE[7] model and it is specifically made for IoT products. Compared to other well-known threat models, such as P.A.S.T.A[8] and attack trees[9], the model used in this thesis is less business driven than P.A.S.T.A and handles complexity better than attack trees.

1.5 Ethics and laws

When researching vulnerabilities in smart locks, there are mainly three Swedish laws that needs to be considered. The first law, The Swedish Penal Code chapter 4 9c §[10], states that anyone who unlawfully tries to gain access to a resource intended for automated processing or unlawfully modifies, obliterates, blocks or registers such a resource is sentenced for data breach of fines or imprisonment for a maximum of two years. The same applies to those who unlawfully interfere with any other similar action or hinder the use of such a resource. It also states that if the act has caused serious damage or intended for a large number of tasks or has otherwise been of a particularly dangerous nature, the sentence will be imprisonment for a minimum of six months and a maximum of six years.

The second law, the Act on Trade Secrets(2018:558)[11], contains regulations on indemnity, prohibition of penalty, and penalties for unauthorized attacks on business secrets. It states that the law applies only to unauthorized attacks on business secrets. As an unauthorized attack, it is never considered that anyone is attacking a business secret to disclose or reveal to an authority or other appropriate body something which is

- reasonably suspected to be a crime with a prison sentence, or
- can be considered to be any other anomaly where the disclosure is to protect the public interest.

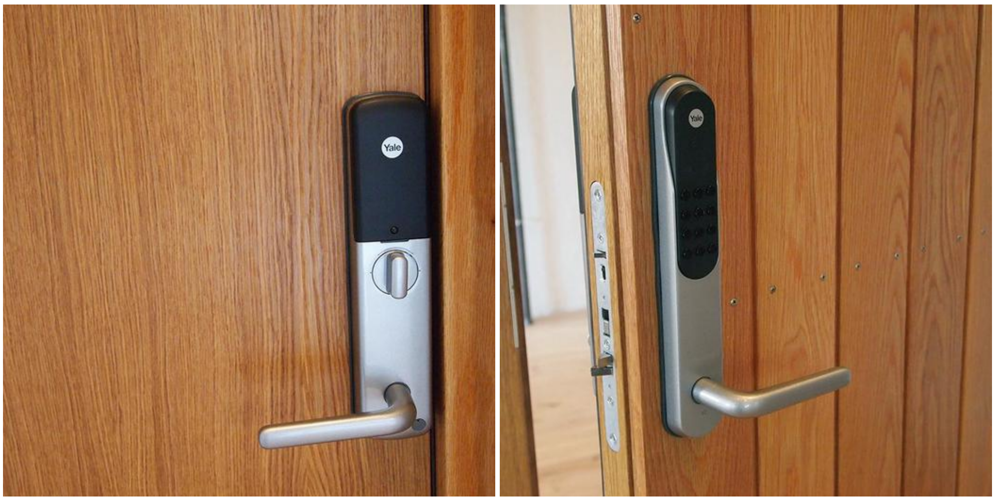
The third law, the law on copyright for literary and artistic works(1960:729)[12], states that copyright includes, with the limitations provided hereinafter, the exclusive right to dispose the work by making copies of it and by making it available to the public, in its original or altered state, in translation or processing, in any other literature or art form, or in other technologies. However, anyone who has the right to use a computer program has the right to observe, investigate or test the program's function to determine the ideas and principles that underlie the program's various details. This applies provided that this occurs during such loading, display on screen, run time, transfer or storage of the program that he is entitled to perform.

Chapter 2

Background

2.1 Physical design of smart locks

The design of smart locks varies, but there are two different design types. The first design type replaces the lock deadbolt with it's own instance, as seen in figure 2.1. The second is mounted on an existing lock deadbolt, as seen in figure 2.2. With the second approach, it is still possible to use the original keys to open the door.



(a) The inside of the door. (b) The outside of the door.

Figure 2.1: An installed lock with replaced lock deadbolt[13].

Figure 2.2: An installed lock on existing lock deadbolt[14].

2.2 Architecture of smart locks

Basic smart locks use a simple connectivity model. A smartphone with Bluetooth pairs with the smart lock which enables it to control the lock using an app, see figure 2.3. The range of controlling whether the lock is locked or not is limited to the range of Bluetooth, causing the lock to only be controllable when an authorized user is nearby the lock. To grant access for other users, the smartphone communicates with a remote cloud server, which sends a grant to the new user's smartphone[15].

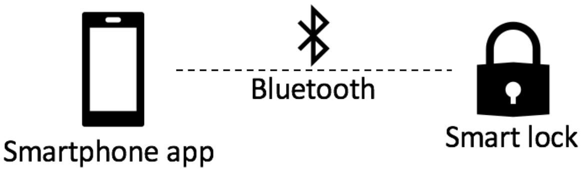


Figure 2.3: Connecting to the lock with Bluetooth.

Advanced locks use an approach where the lock connects to the internet via a separate hub. The hub connects to the lock using Bluetooth or a radio communication standards such as Z-Wave[16]. The hub then connects to an existing network, for instance a WiFi network, to access the internet, see figure 2.4. This architecture is called Device-Gateway-Cloud (DGC) [15]. The type of gateway varies, in modern locks it's often an extra hub than can be purchased in combination with the lock. It can also be an already existing device in the same ecosystem, for instance an Apple TV for Apple Homekit[17], or an hub for Z-Wave[18].

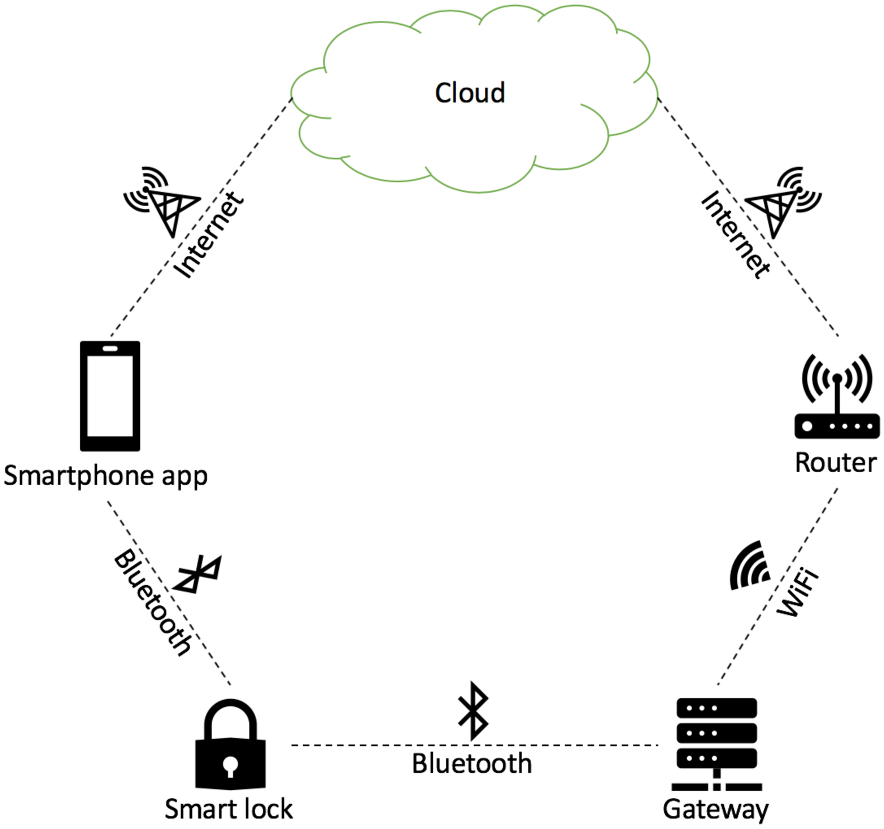


Figure 2.4: Connecting to the lock with remote access.

2.3 Features of smart locks

The main feature of smart locks is to lock and unlock doors, this section aims to introduce common features of smart locks.

2.3.1 User levels

The main benefit with smart locks is the ability to give other users access to the lock. There are mainly two types of user levels. The owner role is the owner of the lock and manages user levels. The guest user can be both time limited and not time limited. If time limited, the user will only be able to control the lock during a specified timespan. The non-time limited guest user can control the lock at all times. Common for both time and non-time limited user is that they can't manage other users, they can only control the lock.

2.3.2 Logging

When a user interacts with the lock, the actions are logged. As an example, it enables parents to know when their kids arrive at home after school when the parents aren't home. It's also a security feature to know if the door has been locked or not, and to know if someone has opened the lock when they were not supposed to.

2.3.3 Vacation mode

Some smart locks which replaces the deadbolt offer vacation mode. Vacation mode is a feature which prevents unauthorized access when users are away from the lock during a longer period of time. If a door has a glass window it is possible for a burglar to break the glass and turn the deadbolt on the inside and unlock the door. If a smart lock is set in vacation mode, it is not possible to turn the deadbolt on the inside[19, 18].

2.4 Vulnerabilities in smart locks

This section covers previous research about security and attacks on smart locks and IoT devices with similar architecture.

2.4.1 State consistency attack

In a study done by Ho et al. [15], four smart locks using the simple version of the DGC architecture were found vulnerable to state consistency attacks. The locks were connected to a cloud server by pairing to an smartphone via Bluetooth and then using the network of the phone to access the cloud server, see figure 2.5

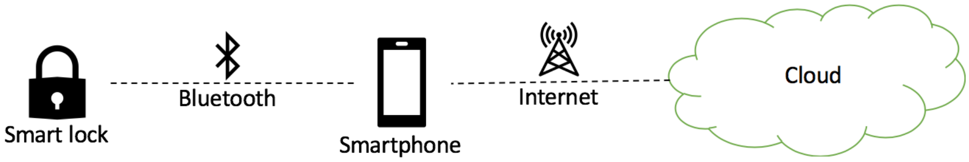


Figure 2.5: Using a smartphone as a gateway to access the remote server.

This made the locks vulnerable to a state consistency attack, because the access list were stored on the smartphones and not in the lock itself. The attack is executed as follows.

1. Alice allows Bob to have temporary access the smart lock. Bob can unlock and lock the smart lock from his smartphone.

2. Bob disables the network of his phone, leaving it unable to connect to the cloud server.

3. Alice revokes Bobs access and gets an message that it has been successfully revoked.

4. The cloud server tries to contact Bobs phone and revoke his access, but fails since Bobs phone is offline.

5. Bob can still use his smartphone to control the lock via Bluetooth, even though the access should have been revoked.

The attack is successfully executed because the lock itself doesn't contain any information regarding who is allowed to access the lock or not. Instead, the information is stored on the smartphone. When Alice revokes Bobs access, the cloud server sends a request to Bobs phone with the message that he no longer has access, but Bobs phone is offline, leaving his state unchanged. Hence, Bob can still control the state of the lock using Bluetooth. This also means that any logging done on Bobs phone isn't sent to the cloud server, making him evade the logging feature. The study also showed that the user interface of Alice's phone showed a message claiming that Bobs access had been successfully revoked.

2.4.2 Overprivilege attacks

Some cloud servers that serves the smart locks have an open API. The API enables third parties to develop apps and scripts to control the lock. In research done by Fernandes, Jung, and Prakash [20], three different overprivilege attacks were successfully executed due to various security design flaws and developer bugs in an open API. The attacks were executed on the Samsung SmartThings platform, which works with a variety of smart locks[21].

The first attack investigated the source code of a third party app which exposed the client ID and secret to obtain an OAuth token. By obtaining an OAuth token, they were able to inject a new PIN code for the lock into the web service which controls the smart lock. The attack leverages overprivilege by using the coarse-binding between the app and the smart device, the use of unsanitized strings in method invocation, and a non-secure implementation of the OAuth protocol.

The second attack used a flaw in privilege isolation. They created an battery monitoring app that only had the privilege to show to battery status of a smart lock. However, due to an overprivilege flaw, the battery monitor app was able to snoop up whenever a new pin was set on the smart lock. This was possible because the lock device handler supports other capabilities besides battery status and didn't isolate them from each other. Hence, the battery app could ask to get reports every time a setting changed in the lock, and then filter the reports for added PINs. The PINs were shown in plaintext.

The third attack exploited the sendLocationEvent API in SmartThings which didn't have any authorization. By writing a small app that raised a false mode change, Fernandes, Jung, and Prakash [20] were able to disable vacation mode for the device. Recalling section 2.3.3, disabling vacation poses a security threat, making it possible to open the door from the inside.

Overprivilege is overall a common problem within IoT products. Recent research of health and fitness apps in IoT programming frameworks showed that 30% of the investigated devices had overprivilege issues[22]. As a countermeasure, researchers are stressing the importance of using least privilege principle to overcome the problem with overprivilege[23, 24, 25].

2.4.3 Storage attacks

Smart locks often encrypt their communication channels using symmetric keys in cryptographic functions such as AES. Manufacturers may fail to securely store these encryption keys on a the smartphone. Ye et al. [26] showed that if a smartphone is rooted (Android) or jailbreaked (iPhone), it's possible to extract the handshake key of August Smart Lock from the file system of the smartphone. The handshake key is used to encrypt the traffic between the application and the lock, meaning that an attacker with the key could communicate and control the lock. This attack also evades the logging feature, leaving the owner unaware if an attacker is locking or unlocking the lock. According to the researchers, the attack would only take 20 seconds to execute. This was also shown by security blogger Jmaxxz at the security conference Defcon in 2016. By activating debug mode on the August smart lock app, he could extract the encryption key which was shown in plain text of the debug logs[27].

The research of Ye et al. [26] also found that sensitive data of the owner was accessible in the files of the rooted/jailbroken devices. Databases and shared preferences were stored in plaintext XML-files. This allowed an attacker to copy the files from the victims phone to their own phone. The attack was executed by creating and signing into a new account on the attackers phone, copy the files from the victims phone to the attackers phone, and then relaunching the app. This made the attacker able to login in as the victim and control their lock.

Since files discovered by Ye et al. [26] where readable in plain text, they also leaked personal information about the user. Information such as the username and the profile photo was available without any encryption.

Another approach to a storage attack is to decompile the source code for the app itself and see if it contains any valuable information. Rose et al. [28] decompiled the Android application package (APK) that used to be stored on a smartphone for the Danalock app. Once they had decompiled the source code, they parsed it with keywords and found a plain text password stored in a table. As an attacker, they were able to gain access to the lock by using the hard-coded password.

2.4.4 Command fuzzing

Although implemented on the Bluetooth Low Energy communication standard, command fuzzing attacks on smart locks are interesting. Rose et al. [28] used a Bluetooth sniffer to collect a valid command to the Okidokeys smart lock. The first two bytes of a command is the operation code (opcode), which specifies which command the device should execute. Rose et al. [28] changed the third byte of the valid command to 0x00, see figure 2.6, and then sent the modified command to the lock. The modified command made the lock crash and enter a error state. As a feature, the lock automatically unlocks when entering an error state.

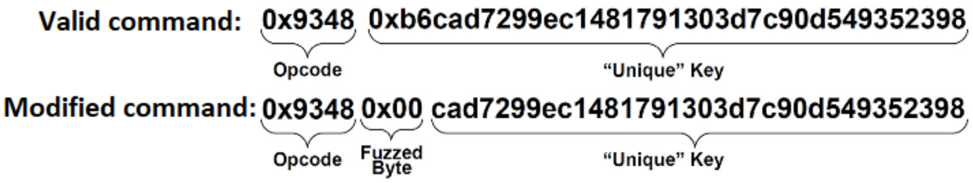


Figure 2.6: Inserting a fuzzed byte into a valid command[28].

2.4.5 Brute forcing

To repeatably try to guess the password of an account is called brute force[28]. With modern computing power, it can be applied if the length of the password is short and the amount of allowed characters is small. If a password is allowed to contain numbers 0-9 and have a maximum length of eight numbers, it would take an attacker approximately twelve days to brute force the password for the account via Bluetooth. This assumes a speed of 6 000 attempts/min[28], which is limited by the transmission rate of the sending device and the speed of which the receiving device translates the data. Table 2.1 shows the estimated completion time to brute force a password depending on the amount of characters and the length of the password.

Available characters	**Password length**	**Password possibilities (millions)**	**Expected completion time (years)**
10	8	10^2	0.03
10	16	10^{10}	3.17
128	8	$7.2 * 10^{10}$	22.83
128	16	$5.1 * 10^{27}$	$1.61 * 10^{18}$
256	8	$1.8 * 10^{13}$	5475
256	16	$3.4 * 10^{32}$	$1.06 * 10^{23}$

Table 2.1: Time expected of brute forcing passwords[28].

Another approach to brute force character based passwords is a dictionary attack[29]. The dictionary attack uses a predefined list of possible passwords that are common and often reused by users. All the passwords in the list are tried as login credentials and checked if they were successful. If successful, the attacker could sign into to the users account and control the lock.

Discovered by security blogger Jmaxxz, the August Smart Lock had a serious password reset flaw in 2015[30]. If a user forgets their password, August would send a six digit verification code. The code would be a number between 000000 to 999999, meaning there is 1 000 000 unique combinations. The problem was that the reset code wasn't time limited, and an attacker could try the code as many times as they wanted. Due to modern computing it was fairly easy to brute force the verification code. An attacker would do a password reset for a known account, and then try every combination of verification codes between 000000 and 999999. By doing so, they were able to reset the password and log in to the victims account with a password of their choosing.

2.4.6 Traffic analysis

During research on a undisclosed smart lock, computer science students Gustafsson and Janstad [31] showed that it is possible for outsiders to extract information from the lock's communication. They analyzed the data which was sent from the lock to the remote cloud server via radio communication. Approximately 70% of the door's communication is encrypted with what they claim to be AES-128. The information that can be extracted is the metadata of the communication in the form of message length (number of packets per message), type of packets, and length of data in the packets. This information was successfully used to categorize the interaction with the door in six different categories: interaction on distance, closing of door, unlocking with code/key tray, opening of door, locking via physical button and input of incorrect code. Besides the radio communication protocol, other researchers[32] has showed that the network traffic of IoT devices can expose user interactions of potentially sensitive data. This is possible even though the data is encrypted.

2.4.7 Man-In-The-Middle attacks

During Defcon 24, the security blogger Jmaxxz showed that it was possible to bypass the certificate pinning between the app and the cloud server on the August smart lock[27]. *Certificate pinning*[33] is a method to prevent man-in-the-middle attacks when using certificates in a public key infrastructure (PKI). This is done by a client attaching a certificate or a public key to a server or such entity. At the next interaction with the server, the client compares its stored validation data to the current certificate without contact with a certificate authority (CA) and can then see if anyone has affected the server or its certificate. Jmaxxz bypassed the certificate pinning by gaining access to a debug console in the app and then changing the API URL which the app connects to. By routing the traffic from the app to a man-in-the-middle proxy instead of the production server, Jmaxxz was able to see all traffic that was going to the production server. He could also fake the responses to the app, pretending to be the cloud API server, see figure 2.7. This was used in two different attacks. He was able to upgrade a guest user to a owner role by changing the userType from 'user' to 'superuser' in the response from the MITM proxy to the app. He could also evade the logging feature of the lock by telling the app that the log was successfully sent to the server, even though it wasn't.

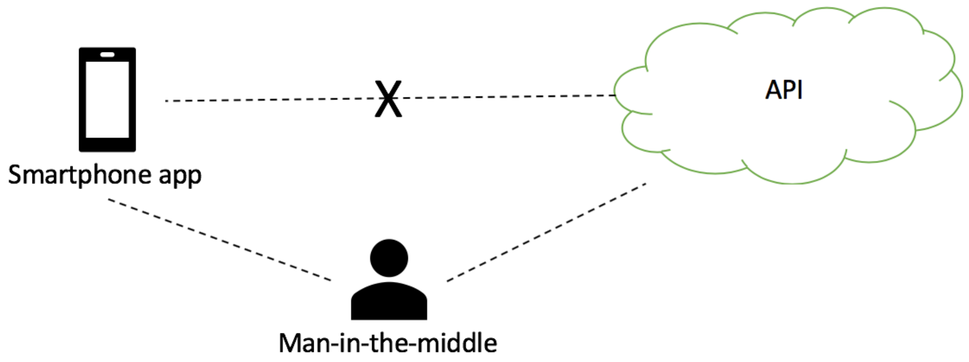


Figure 2.7: Man-in-the-middle attack intercepting traffic between the app and API.

2.4.8 Firmware upgrade attack

By using the same man-in-the-middle technique as in previous section 2.4.7, security blogger Jmaxxz discovered that the firmware on the August Smart Lock wasn't signed[27]. This made it possible to do a firmware update on the lock with customized firmware, possibly with malicious code.

HP investigated 10 of the most popular smart devices available in 2014[34]. They discovered that 60% of the devices did not encrypt the traffic when downloading software updates. In some cases, they found that it was possible to intercept, extract and mount the downloads of the software. By mounting the software, they could view and modify the software, making it possible to inject malicious code before it was downloaded to the device.

2.4.9 Other attacks

There are other attacks mainly focusing on the Bluetooth communication between the lock and gateway. For instance unintentional unlocking[15], relay attack[15], replay attack[28] and Denial-of-Service(DoS) attack[26]. However, since these attacks use Bluetooth as communication protocol, they are out of scope for this thesis.

Chapter 3

Threat model

The following chapter describes the threat model used for the thesis. The chapter begins by introducing the threat model and continues by applying it to smart locks.

3.1 Threat modeling smart locks

The first step when conducting a security evaluation of smart locks is to understand the ecosystem in which the smart lock devices operates, the target of evaluation (ToE)[6]. In order to do this, assets, potential adversaries and attack surfaces are identified. After doing so, it is possible to model the ToE and apply the threat model. The process of creating the threat model can be divided in to four steps:

1. Identify the most valuable assets. Analyse use cases and the target of evaluation to gain knowledge about assets that should be protected.
2. Identify potential adversaries. Who can and would try to attack the device?
3. Identify attack surface. Which components of the device are exposed?
4. Model ToE and identify threats. Use the STRIDE model to identify and classify potential threats.

The STRIDE model

The STRIDE model[7] is a well-known threat model developed by Microsoft and it's used to identify security threats in electronic devices. There are other well-known threat models, such as P.A.S.T.A[8] and attack trees[9], but in comparison to STRIDE the P.A.S.T.A model is more business driven and attack trees handles complexity more poorly. There is a tool for creating threat models with STRIDE, called the Microsoft Threat Modeling Tool[35]. It's free to download and use. The tool is used to create a data flow chart of the target of evaluation and then generates a report of potential threats. The threats are categorized in the STRIDE categories, which are:

- Spoofing identity - Accessing and using another users authentication details.
- Tampering with data - Modifying data in a malicious way.
- Repudiation - Unable to prove the integrity and origin of data.
- Information disclosure - Leakage of personal and sensitive data.
- Denial of service - Make a service unavailable and/or unusable.
- Elevation of privilege - Gain elevated access to resources that are protected from the user.

3.2 Applying the threat model

Applying the process of threat modeling described in previous section, the following subsections are the outcome of the four steps described in section 3.1.

3.2.1 Assets

The following assets has been identified as assets that needs to be protected.

Firmware

During the analyze of the ToE, four different firmware assets has been identified to protect against attacks.

- The firmware of the lock - controls the physical device whether the lock is open or closed, which makes it an exposed asset. Altering the firmware of the lock could lead to unintentional denial of service attacks or an attacker gaining access to control the lock, similar to the firmware upgrade attack by Jmaxxz in section 2.4.8.

- The firmware of the hub - communicates the data from the cloud API to the smart lock. Changing the firmware of the hub might lead to unintentional denial of service where communication to the lock gets blocked. It might also act as an man-in-the-middle in order to read and/or modify communication between the cloud API and the smart lock, similar to the man-in-the-middle attacks described in 2.4.7.
- The firmware of the smartphone app - lets the user control the lock both locally and remotely. It might contain sensitive information about the user, as discovered by Ye et al. [26] in 2.4.3. It might also be possible to elevate privilege, as described in section 2.4.2.
- The firmware of the cloud API - handles communication between the smartphone app and the hub. Altering the firmware could lead to unauthorized access, escalation of privilege and possibly denial of service.

Certificates and encryption keys

Certificates and encryption keys are assets that are used to encrypt the traffic between the devices in the ecosystem. If the certificates and encryption keys would be compromised, an attacker might gain access to sensitive data or escalate privilege to gain access to the lock, as described by Jmaxxz [27] in section 2.4.7 or by Ye et al. [26] in section 2.4.3.

Credentials

The credentials of the users in the ecosystem are used to grant authorization to the lock. Compromised credentials might lead to an attacker gaining access to the lock, as Ye et al. [26] used in the storage attack in section 2.4.3. Poor implementation of authentication for credentials might also make it easy for an attacker to apply brute force techniques as described in section 2.4.5.

Event logs

The event logs contain information regarding the state of the lock, who used it, and at what time. The logs serves transparency to the users, keeping them updated on activity of the lock. By evading the log feature an attacker might open the lock without logging, meaning that users wont be notified about the state of the lock, as described by Jmaxxz [27] in section 2.4.7.

Communication channels

The traffic consists of the network communication between the smartphone app and the cloud API, and the network communication between the hub and the cloud API. It also consists of the Bluetooth communication between the lock and the hub, and the Bluetooth communication between the smartphone app and the lock. As proven by Gustafsson and Janstad [31] in section 2.4.6 it might be possible to gain knowledge about the state of the lock and user interaction by using traffic analysis. This would make it possible for an attacker to gain knowledge about the state of the lock, even when the data is encrypted.

3.2.2 Potential adversaries

The following adversaries have been identified when analyzing smart locks. They're based on the generic adversary model by ARM[6].

Remote software attacker

An external adversary that wants to gain access to the lock. The external adversary does not have access to any internal resources, meaning that he/she can only try to gain access to the lock from the outside. The remote attacker might attack one or a network of devices in order to find software vulnerabilities.

Network attacker

A network attacker has access to the same ecosystem. It can be divided into two parts, one user that already has access to the lock but tries to elevate it's privilege, The second user has the same lock model and tries to gain access to a second lock which it is not authorized to do. The network attacker might intercept the network traffic by using techniques such as the man-in-the-middle attack.

Malicious insider attacker

A bad malicious inside attacker may be someone employed at the manufacturer of the lock with malicious intentions. For instances implementing a backdoor in the software, or fetching personal data about the user without their consent. A malicious inside attacker is accounted as someone who has malicious intention during the manufacturing process. This also includes the owner of the cloud where the cloud API is located. This category also includes an inexperienced user, a user that might misconfigure services due to lack of technical experience.

Hardware attacker

A hardware attacker can be dived in to two sub categories, simple hardware attacker and advanced hardware attacker. The simple hardware attacker might USB dongles, debug ports, etc. to gain access to the hardware. The advanced hardware attacker has unlimited resources and might use advanced equipment to deploy advanced attacks. Due to the scope of the thesis, the hardware attacker is not considered.

3.2.3 Attack surfaces

The attack surfaces can be dived into four main categories, communication, lifecycle, software and physical[6].

Communication

The communication attack surface includes communication between all of the components in the smart lock ecosystem. Four different communications have been identified:

- The network (HTTPS) communication between the smartphone app and the cloud API.
- The network (HTTPS) communication between the smart lock hub and the cloud API.
- The Bluetooth communication between the smart lock and the smart lock hub.
- The Bluetooth communication between the smart lock and the smartphone app.

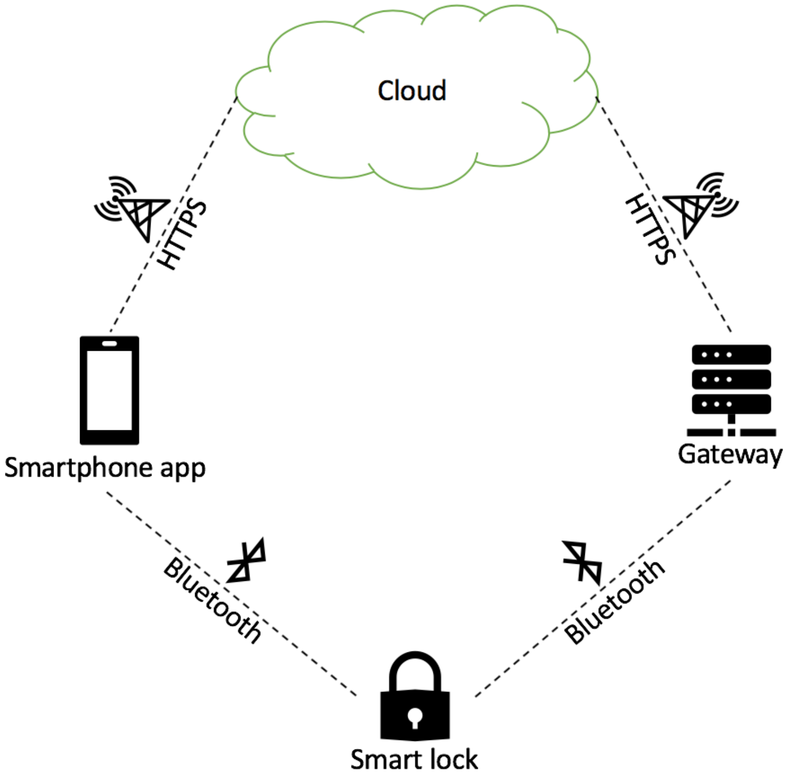


Figure 3.1: The communication attack surfaces.

Lifecycle

The lifecycle attack surface can be divided into two part, user lifecycle and product lifecycle. The user lifecycle, when the lock is transferred to another user, is an attack surface as the ToE might contain data from the previous owner. The product lifecycle, when the lock manufacturer has declared end of life of the product, is an attack surface if a user continues to use an discontinued product.

Software

The software of the ToE handles the interactions between the user, the lock, the hub and the cloud API. It consists of a login interface, management of users, logging, and local storage.

Hardware

The hardware of the ToE consists of the lock itself and its hub. The hardware may contain debug ports, USB dongles, emit electromagnetic fields etc, which can be used to attack the hardware. Due to the scope of the thesis, the hardware attack surface is not considered.

3.2.4 Threat model and STRIDE

The threat model is conducted using the Microsoft Threat Modeling Tool 2018. The threat model diagram is based on the architecture of the smart locks, see figure 3.2. Microsoft Threat Modeling Tool automatically generates a report of threats based on the diagram. The generated report contained a total of total 71 threats which are categorized in the STRIDE model. The threats have been summarized and accounted for in tables 3.1, 3.2, 3.3 and 3.4 below.

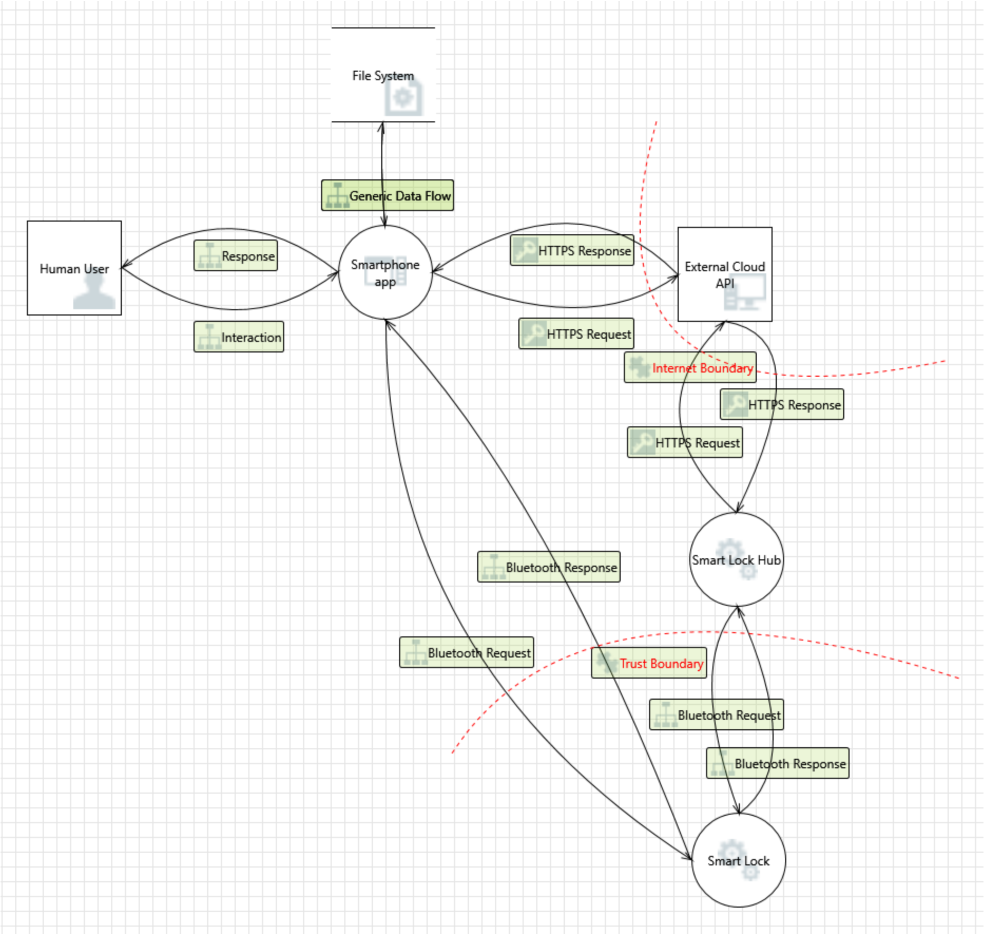


Figure 3.2: Threat model of smart locks created in Microsoft Threat Modeling Tool.

The interaction between the User and the Smartphone App has two threats, which are described in table 3.1.

Table 3.1: Threats in interaction between User and Smartphone App

IDs	Threat	Category	Description
70	Spoofing the Human User External Entity	Spoofing	Human User may be spoofed by an attacker and this may lead to unauthorized access to Smartphone app. Consider using a standard authentication mechanism to identify the external entity.

Continuation of Table 3.1			
71	Elevation Using Impersonation	Elevation Of Privilege	Smartphone app may be able to impersonate the context of Human User in order to gain additional privilege.
End of Table			

The file system storage of the smartphone is an attack surface with four potential threats. The threats are described in table 3.2 below.

Table 3.2: Threats in interaction between the Smartphone App and the Smartphone File System.

IDs	Threat	Category	Description
46	Potential Excessive Resource Consumption for Smartphone app or File System	Denial Of Service	Does Smartphone app or File System take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.
47	Spoofing of Destination Data Store File System	Spoofing	File System may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of File System. Consider using a standard authentication mechanism to identify the destination data store.
48	Weak Access Control for a Resource	Information Disclosure	Improper data protection of File System can allow an attacker to read information not intended for disclosure. Review authorization settings.
49	Spoofing of Source Data Store File System	Spoofing	File System may be spoofed by an attacker and this may lead to incorrect data delivered to Smartphone app. Consider using a standard authentication mechanism to identify the source data store.
End of Table			

The HTTPS communication between the Smartphone App and the Cloud API has 10 potential threat according to the report generated by Microsoft Threat Modeling Tool. The threats are described in table 3.3 below.

Table 3.3: Threats in HTTPS communication between the Smartphone App and the Cloud API.

IDs	Threat	Category	Description
53, 60	Data Flow HTTPS - Request/Response Is Potentially Interrupted	Denial Of Service	An external agent interrupts data flowing across a trust boundary in either direction.
54	External Entity External Web Service Potentially Denies Receiving Data	Repudiation	External Cloud API claims that it did not receive data from a process on the other side of the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
55, 57	Spoofing of the External Web Service External Destination Entity	Spoofing	External Cloud API may be spoofed by an attacker and this may lead to data being sent to the attacker's target instead of External Cloud API. External Cloud API may be spoofed by an attacker and this may lead to unauthorized access to Smartphone app. Consider using a standard authentication mechanism to identify the external entity.
56	Elevation Using Impersonation	Elevation Of Privilege	Smartphone app may be able to impersonate the context of External Cloud API in order to gain additional privilege.
58	Elevation by Changing the Execution Flow in Smartphone app	Elevation Of Privilege	An attacker may pass data into Smartphone app in order to change the flow of program execution within Smartphone app to the attacker's choosing.

Continuation of Table 3.3			
59	Smartphone app May be Subject to Elevation of Privilege Using Remote Code Execution	Elevation Of Privilege	External Cloud API may be able to remotely execute code for Smartphone app.
61	Potential Process Crash or Stop for Smartphone app	Denial Of Service	Smartphone app crashes, halts, stops or runs slowly; in all cases violating an availability metric.
62	Potential Data Repudiation by Smartphone app	Repudiation	Smartphone app claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
End of Table			

The attack surface consisting of the HTTPS communication between the Smart Lock Hub and the Cloud API, has 10 threats to consider according to the report generated by Microsoft Threat Modeling Tool. The threats are described in table 3.4 below.

Table 3.4: Threats in HTTPS communication between the Smart Lock Hub and the Cloud API.

IDs	Threat	Category	Description
50, 66	Data Flow HTTPS - Request/Response Is Potentially Interrupted	Denial Of Service	An external agent interrupts data flowing across a trust boundary in either direction.
51	External Entity External Web Service Potentially Denies Receiving Data	Repudiation	External Cloud API claims that it did not receive data from a process on the other side of the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.

Continuation of Table 3.4			
52, 64	Spoofing of the External Cloud API External Destination Entity	Spoofing	External Cloud API may be spoofed by an attacker and this may lead to data being sent to the attacker's target instead of External Cloud API. External Cloud API may be spoofed by an attacker and this may lead to unauthorized access to Smart Lock Hub. Consider using a standard authentication mechanism to identify the external entity.
63	Elevation by Changing the Execution Flow in Smart Lock Hub	Elevation Of Privilege	An attacker may pass data into Smart Lock Hub in order to change the flow of program execution within Smart Lock Hub to the attacker's choosing.
65	Smart Lock Hub May be Subject to Elevation of Privilege Using Remote Code Execution	Elevation Of Privilege	External Cloud API may be able to remotely execute code for Smart Lock Hub.
67	Potential Process Crash or Stop for Smart Lock Hub	Denial Of Service	Smart Lock Hub crashes, halts, stops or runs slowly; in all cases violating an availability metric.
68	Elevation Using Impersonation	Elevation Of Privilege	Smart Lock Hub may be able to impersonate the context of External Cloud API in order to gain additional privilege.
69	Potential Data Repudiation by Smart Lock Hub	Repudiation	Smart Lock Hub claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
End of Table			

The attack surface of the Bluetooth communication between the smart lock and the smart lock hub conducted 22 threats in the report. The Bluetooth communication between the smart lock and smartphone app conducted 23 threats according to the report from Microsoft Threat Modeling Tool. Due to the scope of the thesis, they're not explained further in this chapter.

Chapter 4

Methodology

This chapter begins by introducing the scope of attacks. It continues by describing the methodology of the executed attacks and the motivation when selecting locks. Lastly, the chapter explains the lab environment used when applying the some attacks to the smart locks.

4.1 Scope of attacks

It would be to extensive to cover all 71 threats in the threat model within the limits of the thesis. To motivate the choosing of threats, this thesis will focus on the threats that have been proven to have vulnerabilities based on the previous work in section 2.4. Due to the time limit of the project, it is not feasible to try all vulnerabilities in section 2.4. Hence, this report will focus on the following three attacks:

1. State consistency attack from section 2.4.1.
2. Brute force attacks from section 2.4.5.
3. Man-in-the-middle attack from section 2.4.7.

4.2 Attacks

The following section describes the attacks that has been conducted for the smart locks.

4.2.1 State consistency attack

The state consistency attack in section 2.4.1 is possible since the list of authorized list of users is saved on the smartphone rather than the lock. This has been captured by the threat model in threat 71 of table 3.1, *Smartphone app may be able to impersonate the context of Human User in order to gain additional privilege.* With the adding of the smart lock hub, the lock should be able to keep track of which users that are allowed to access the lock and not. As a malicious user, we want to investigate if it is possible to still apply the state consistency attack. If it is possible, a malicious user with guest access to the lock might be able to bypass the revocation of the access. This also applies to an attacker trying to add an already registered lock to its own account. Hence, the following will be considered in terms of the state consistency attack:

- Is it possible to add an already registered lock to a second account?
- Is it possible to evade the revocation of a time limited user?
- Is it possible to evade the revocation of a non-time limited user?
- Is it possible to evade logging?

4.2.2 Brute force attack

Recall threat with ID 70 in table 3.1, *Human User may be spoofed by an attacker and this may lead to unauthorized access to Smartphone app. Consider using a standard authentication mechanism to identify the external entity.* The Smartphone app uses login credentials as authentication mechanism. The credentials ensures that only an authenticated user is allowed to access the lock. The user credentials are not bound to a device, meaning that a user can login with its credentials on any device that supports the Smartphone app. Spoofing the login credentials would mean that an malicious user would gain unauthorized access to the lock. Hence, it is important that appropriate security measures are taken in order to protect the users credentials.

The brute force attack focuses on an external adversary trying to gain access to a users credentials. As proven in section 2.4.5, it is possible to brute force a password if it is limited in available characters and length. It is also possible to use dictionary attacks with common words and passwords in order to try and guess the correct password. To test the authentication mechanism, two procedures are examined: the account creation procedure and the password reset procedure. To test the account creation, a new account is created and the following criterion's are checked:

- What is the least amount of characters that can be conducted in the password? Are they sufficient in order to prevent password brute forcing described in section 2.4.5?
- Is the password policy using good measure as recommended by OWASP[36]:
  - Password must meet at least 3 out of the following 4 complexity rules:
    - 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 (such as #, / and -)
  - at least 10 characters
  - at most 128 characters
  - not more than 2 identical characters in a row (e.g., 111 not allowed)
- Are common passwords prohibited, such as *password* and *12345*, or would it be possible to use a dictionary attack, as described in section 2.4.5?
- Is multi-factor authentication available and mandatory[37]?
- After the account is created, is it possible to use username enumeration to gain knowledge about existing accounts[38]?
- Is there a limit of login tries in order to prevent brute forcing credentials[37]?

In order to test security flaws in password resets, an account is created on the Smartphone app. The account requests to reset the password in the Smartphone app. The architecture for resetting passwords is examined to find vulnerabilities, specifically previous password reset vulnerabilities explained in section 2.4.5. The following tasks are investigated, as recommended by OWASP[39]:

- Is it possible to order multiple password resets?
- If it is possible to order multiple password resets, does previous ordered resets get invalidated?
- How long is a password reset valid?
- Is it possible to gain knowledge about existing accounts in the password reset feature?

4.2.3 Man-in-the-middle attack

In section 2.4.7, security blogger Jmaxxz was successfully able to use man-in-the-middle techinques in order to spoof responses from the cloud API when the smartphone app did a request. This has been acknowledged as threats in the threat model: threat 55 and 57, *External Cloud API may be spoofed by an attacker and this may lead to data being sent to the attacker's target instead of External Cloud API. External Cloud API may be spoofed by an attacker and this may lead to unauthorized access to Smartphone app. Consider using a standard authentication mechanism to identify the external entity.* Spoofing responses has in previous work proved that a malicious user in the same ecosystem may elevate a guest user to owner, and evade logging. Elevating privilege from guest to owner would mean that the attacker gains access to resources that he/she does not have authorization to. This is found in threat 58 of the threat model, *An attacker may pass data into Smartphone app in order to change the flow of program execution within Smartphone app to the attacker's choosing.* The following will be considered if it is possible to spoof responses from the cloud API:

- Is it possible to elevate privilege of a guest user?
- Is it possible to spoof a successful login?
- Is it possible to add already registered locks to an second account without authorization?

4.3 Selection of smart locks

A list off all locks considered can be found in table 4.1. Two locks were selected from the list, one from a Swedish online retail shop, and one from AliExpress.com. The following delimits has been considered when selecting which locks to investigate.

- The locks should have a hub that makes the lock available for remote control.
- The hub should manufactured by the same company as the manufacturer of the lock, e.g. locks that use open protocols such as Z-Wave are not selected.
- The locks should be manageable from an smartphone app or web interface.

- They should be able to order to Sweden and should be adapted for Scandinavian doors.
- As a last delimiter, the cheapest locks were selected.

Table 4.1: Comparison of smart locks.

Name	**Has app?**	**Has web interface?**	**Retail price**	**Comment**
Yale Doorman with Aptus module[40]	Yes	Yes	20 000kr	Webb+server available from Aptus. Requires training from Aptus.
Yale Doorman[41]	Yes	No	5 500kr	None.
Net Smart Lock	Yes	No	1 900kr	None.
ID Lock 150[42]	Yes	Yes	4 400kr	No own hub, only external hub via Z-Wave.
Danalock V3[43]	Yes	Yes	1 500kr	No own hub, only external hub via Z-Wave.
Lockitron Bolt[44]	Yes	No	3 600kr	Due to recent acquisition not available for order.
Wattle Door Lock[45]	Yes	No	1 900kr	Not shipping to Sweden.
August Smart Lock Pro[46]	Yes	No	3 000kr	Not available in Swedish retail.
Candy House Sesame[47]	Yes	Yes	2 100kr	Still in beta-testing.
Nuki Smart Lock 2.0[48]	Yes	Yes	3 200kr	None.
Friday Lock[49]	Yes	No	3 300kr	Requires iPad or AppleTV as hub.
Electronic Door Lock ordered from AliExpress.com	Yes	No	1 100kr	None.

Continuation of Table 3.1
End of Table

The Net Smart Lock

The first lock, the Net Smart Lock, is created in Stockholm, Sweden. Due to responsible disclosure of the findings in this report, the name nor hyperlink of the lock will be provided in this version of the report. In the following part of the report it will be named the Net Smart Lock.

The Net Lock is mounted on a existing deadbolt. It has two available smartphone apps. The first is the main app which the user manages its lock. The app can add and remove locks and hubs, invite other users to be able to control the lock, and view logs of the lock. It has three different user roles, *Guest*, *Resident* and *Owner*. Guests and residents can control the lock and view their own access logs. They can control the lock via Bluetooth but not remotely over the internet. The guest user is a time limited user, whilst the resident user is not time limited. The owner role can lock and unlock both via Bluetooth and remote control, it can view all users activity, and it can also invite and remove users.

The purpose of the second app is that a user can order deliveries to Net and then schedule a time in the app when Net will delivery it inside of the users property. The person handling the delivery for Net will receive a one-time-use code in order to enter the users home. In order to prove a safe delivery, the whole delivery is filmed by the person doing the delivery. The user can then view the video confirmation that the delivery has been made.

The Ali Lock

The second lock is ordered form AliExpress.com and due to responsible disclosure of the findings in this report, the name nor hyperlink of the lock will be provided in this version of the report. In the following part of the report it will be named the Ali Lock. The lock replaces the deadbolt, meaning that it's not possible to use the original key in order to open the lock. The lock has one smartphone app that is recommended by the manufacturer. However, the app uses an platform that any manufacturer of a smart lock can use to develop their own smartphone app.

4.4 Lab environment

In order to examine network traffic, a lab environment is set up. The environment is used for analyzing the traffic between the smartphone app and the cloud API. The traffic from the smartphone is routed trough a proxy called *mitm-proxy*[50] which is installed on a computer, see figure 4.1. The computer can intercept HTTPS requests and responses between the smartphone and the API.

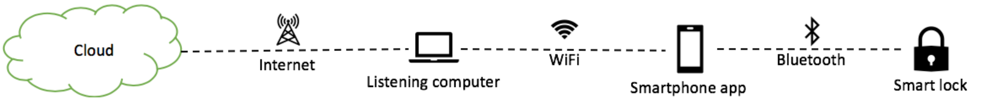


Figure 4.1: Lab environment for analyzing traffic between the API and the smartphone.

In order to test the smartphone app, an iPhone 6 16GB with iOS 12.0 is used. In the cases where a second account is needed, another iPhone 6 16GB running iOS 10.2.1 is used. This is to ensure that the two accounts can't interfere with each others storage.

Chapter 5

Results

The following chapter concludes the findings when applying the attacks from the previous chapter.

5.1 State consistency attack

Recalling the state consistency attack in section 4.2.1, the following aspects are tested:

- Is it possible to add an already registered lock to a second account?
- Is it possible to evade the revocation of a time limited user?
- Is it possible to evade the revocation of a non-time limited user?
- Is it possible to evade logging?

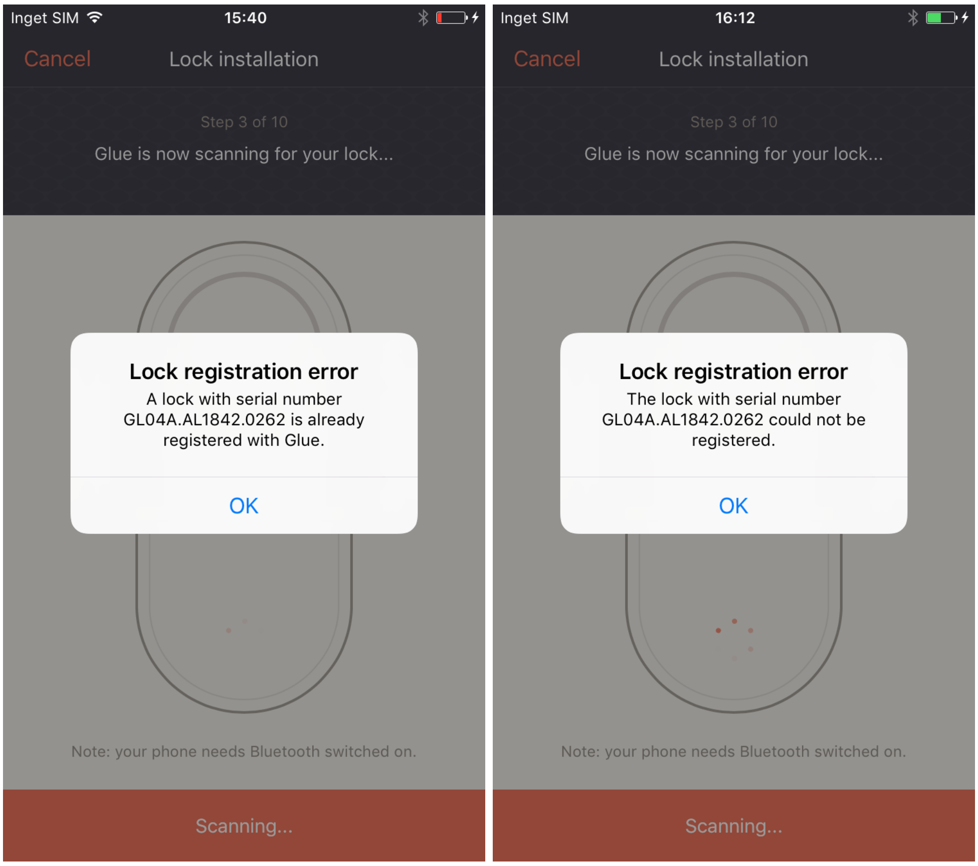
5.1.1 Adding an already registered lock to a second account

Net Smart Lock

The Net Smart Lock is installed by first downloading the smartphone app and create an account. Once the account is confirmed, the user mounts the lock on the door and inserts batteries to the lock. The user scans a QR code in the app that contains a serial number for the lock. It's also possible to manually enter a serial number. The app will then start scanning for the lock via Bluetooth. If found, it will start to calibrate the lock. Once the lock is calibrated the user is prompted to name the lock and add it to a property. Once the lock installation is completed, the user is prompted to install the hub.

An account was setup on the first smartphone and registered the lock. A new account was then setup at the second smartphone, which tried to add the same lock. This gave an error message that the lock had already been registered, see figure 5.1a. It was also not possible to register the lock when the second smartphone had disabled network communication, see figure 5.1b.

During the testing of this attack, it was noticed that the Net Smart Locks uses incremental serial numbers. As it's possible to manually enter serial numbers during the installation of a lock, it would be possible to add locks that hasn't been activated yet. This would in theory make it possible to create a denial of service attack for locks that hasn't been registered yet by activating them before the real owner does. If a user buys a lock that has already been activated by a malicious user, it's not possible to register that lock and it would be useless for the user until it's deactivated from the malicious users' account. The same attack applies to the hubs as they also use incremental serial numbers.



Glue is now scanning for your lock...

Lock registration error

A lock with serial number GL04A.AL1842.0262 is already registered with Glue.

Note: your phone needs Bluetooth switched on.

Glue is now scanning for your lock...

Lock registration error

The lock with serial number GL04A.AL1842.0262 could not be registered.

Note: your phone needs Bluetooth switched on.

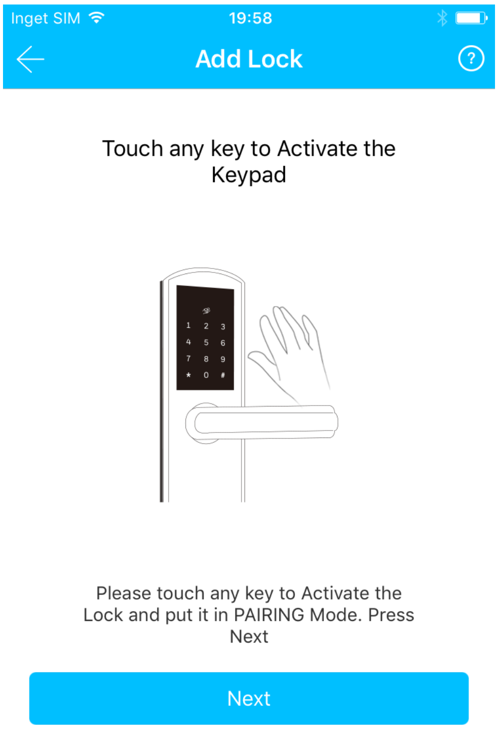
(a) Trying to add an already registered lock.

(b) Trying to add an already registered lock, without any network.

Figure 5.1: Trying to add already registered locks in the Net app.

The Ali Lock

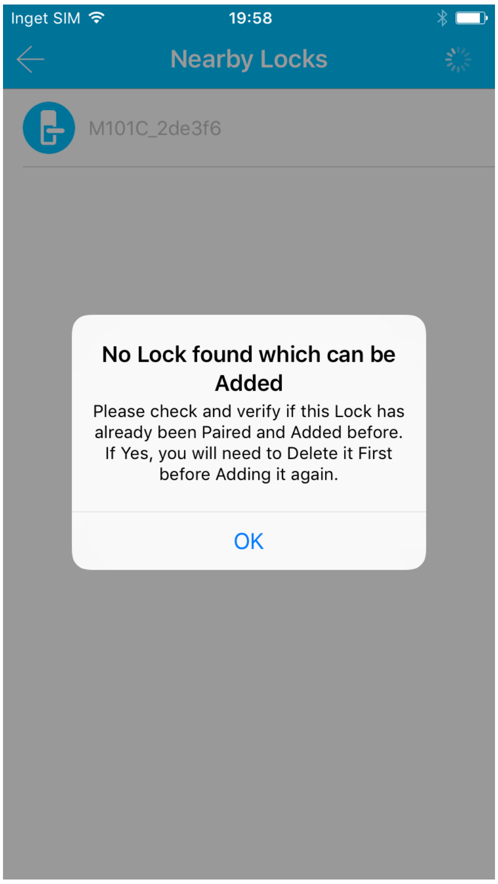
When registering a lock in the smartphone app for the Ali Lock, the owner first creates an accounts and logins in to it in the app. After doing so, the user taps the *Add new lock* button in the app. The lock needs to be activated by pressing any button on the keypad, see figure 5.2. After doing so, the lock will show up in the app with a "+"-sign next to it. Tapping the "+"-sign will prompt the user to name the lock. The lock is now associated with the user. When trying to add the lock to a second account, after it has been associated with the first account, the second user follows the same steps as the first user. However, when searching for nearby lock, there is no "+"-sign next to the locks name. After a short moment, the app will tell the second user that the nearby lock can't be associated, see figure 5.3. It was also not possible to add the lock to a second account with the network unavailable, with the same error message.



Touch any key to Activate the Keypad

Please touch any key to Activate the Lock and put it in PAIRING Mode. Press Next

Figure 5.2: Activating the lock in order to pair it with the smartphone app.



No Lock found which can be Added

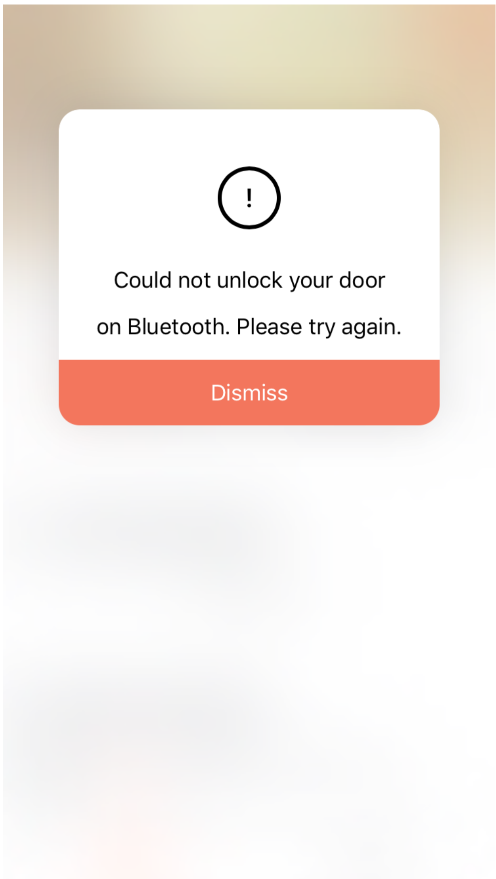
Please check and verify if this Lock has already been Paired and Added before. If Yes, you will need to Delete it First before Adding it again.

Figure 5.3: Trying to add an already registered lock.

5.1.2 Evade the revocation from a time limited account

Net Smart Lock

To test the guest account of the Net Smart lock, a time limited guest user was created and invited to share the lock. The guest user first tried locking the lock with network enabled and while the timeframe of the users access was valid. The guest user then disabled network, leaving only Bluetooth activated. The guest user then waited for the valid timeframe to expired, and then tried to unlock the lock. The lock was successfully unlocked, meaning that it was possible for the guest user to evade the time limits of its account. This approach only worked when the lock was not paired with the hub. If the lock was paired with the hub, the guest would get an error message when trying to unlock the lock without network, see figure 5.4.



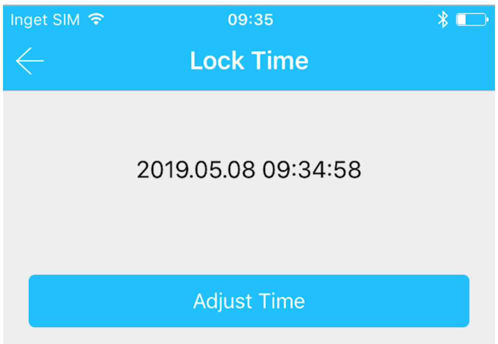
!

Could not unlock your door on Bluetooth. Please try again.

Figure 5.4: Error message when trying to control the lock without network.

The Ali Lock

For the Ali Lock, the time limited guest user was conducted in the same way as for the Net Lock. However, when time expired, it was not possible to unlock the lock. The app would display an error message of *Operation failed*. When looking into the settings of the lock, there seems to be a clock installed on the lock itself, see figure 5.5. The clock in the lock keeps track if the timed key has passed its time limit. It is possible to adjust the time on the clock, however the time is fetched from the cloud API, and not the smartphones current time. Therefore, it was not possible to spoof the time at this stage.



Inget SIM 09:35

Lock Time

2019.05.08 09:34:58

Figure 5.5: The time of the lock

5.1.3 Evade the revocation from a non-time limited account

Net Smart Lock

To test the guest account of the Net Smart lock, a guest user was created and invited to share the lock. The guest user first tried locking the lock with network enabled. The guest user then disabled network, leaving only Bluetooth activated. The owner of the lock revoked the access of the guest user. When the guest user had no network, he/she could still control the lock via Bluetooth, even though access should have been revoked. As soon as the guest user enabled network, the access was revoked. This approach only worked when the lock was not paired with the hub. If the lock was paired with the hub and trying to unlock the lock without network, the guest would get the same error message as with the time limited account, see figure 5.4.

The Ali Lock

The test of evading the guest account for the Ali Lock was conducted in the same way as the Net Smart Lock. A guest user was created and invited to share the lock. The guest user first tried locking the lock with network enabled. The guest user then disabled network, leaving only Bluetooth activated. The owner of the lock revoked the access of the guest user. When the guest user had no network, he/she could still control the lock via Bluetooth, even though access should have been revoked. In comparison to the Net Smart Lock, the Ali Lock app showed the access key as *"Deleting..."* in the owners app when the guest user had turned off network and it had been revoked. The next time the guest user connected to network, it was marked as deleted. Hence, the Ali Lock showed an indication that the access hadn't been completely removed when the guest user had turned of its network. This approach worked regardless if the lock was paired with the hub or not.

5.1.4 Log evading

Net Smart Lock

To test the log evading of the Net Smart lock, a guest user was created and invited to share the lock. The guest user first tried locking the lock with network enabled. This triggered an notification to the owner of the lock that the guest user had locked the lock. The guest user disabled network, but left Bluetooth on, and was still able control the lock. Since there was no network, no notification was sent to the owner of the lock. Hence, it was possible for the guest user to evade the logging feature. This approach was only successful when the lock wasn't paired with the hub. If the lock was paired with the hub, it was not possible to control the lock without network. The smartphone app would display the same error message as in figure 5.4, that it was not possible to control the lock via Bluetooth.

One thing that should be noted regarding log evasion is when a guest user revokes its access by itself, or gets removed by the lock owner, the logs from the guest user are removed as well. Hence, there will be no logs confirming that the guest user has ever existed. This behavior was present regardless if the lock was paired with the hub or not.

The Ali Lock

The test of log evading on the Ali Lock is executed similar to the Net Smart Lock, and had the same results as with evading the non-time limited account. The guest user tried to lock the lock with network enabled, which was successfully logged to the owner of the lock. The guest user then disabled network, leaving only Bluetooth activated. When the guest user had no network, they could still control the lock via Bluetooth. Since the smartphone app was unable to access any network, the logs of the guest user controlling the lock was never sent to the owner of the lock. Hence, the it was possible to evade logging. This approach worked regardless if the lock was paired with the lock or not.

5.2 Brute force attack

To test the brute force attack described in 4.2.2, two procedures are examined: the account creation procedure and the password reset procedure.

5.2.1 Account creation

To test the account creation, a new account is created and the following criterion's are checked:

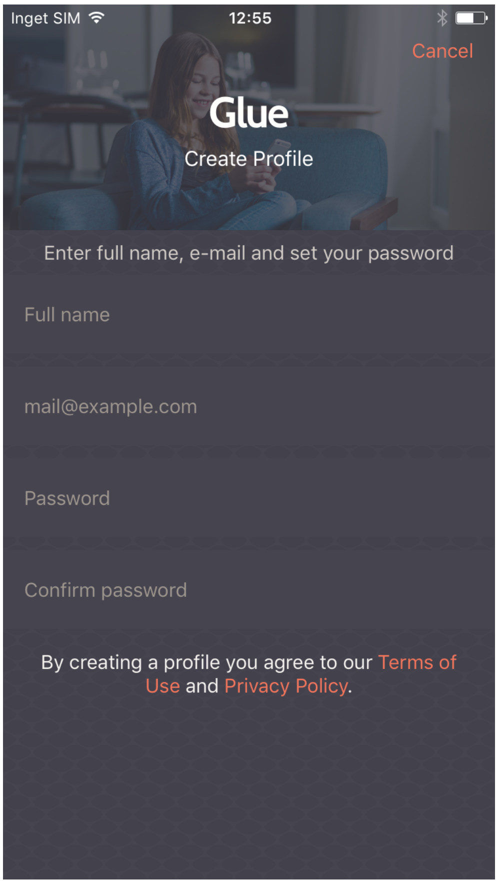
- What is the least amount of characters that can be conducted in the password? Are they sufficient in order to prevent password brute forcing described in section 2.4.5?
- Is the password policy using good measures as recommended by OWASP:
  - Password must meet at least 3 out of the following 4 complexity rules:
    - 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 (such as #, / and -)
  - at least 10 characters
  - at most 128 characters
  - not more than 2 identical characters in a row (e.g., 111 not allowed)
- Are common passwords prohibited, such as *password* and *12345*, or would it be possible to use a dictionary attack, as described in section 2.4.5?
- Is multi-factor authentication available and mandatory?
- After the account is created, is it possible to use username enumeration to gain knowledge about existing accounts?
- Is there a limit of login tries in order to prevent brute forcing credentials?

Net Smart Lock

When creating a new account in the Net Lock app, the first step is to enter a phone number. Upon clicking *Create Account*, a SMS is sent to the users phone with a six digit code. The code is used to confirm the account. Once the code has been entered, the user enter full name, e-mail and a password twice, see figure 5.6. If the user chooses a password shorter than 8 characters, the app will give an error message, informing that the password is to short. When entering the password, both upper- and loswercase charcters, digits, and special characters are allowed. However, the only password policy that is enforced is the minimum length of 8 characters. Hence it's possible to use passwords such as *password*, *11111111* and *12345678*. If the user changes the password after the account has been created in the app, the password policy restricts to 6 characters instead of 8 characters which was the minimum length during registration. As with the registration, there is no password policy other than the length of the password when changing the password.

There is no multi-factor authentication implemented. When logging in to a new smartphone, the user only needs its credentials to verify their identity. When trying to create an account with a phone number that is already registered with Net, the app will let the user know that the phone number has already been used, see figure 5.7.

To test if it there was a limit of login attempts, 20 login attempts with the wrong password were conducted during 30 seconds. All login attempts were successful and without stressing the server any further, it seemed to be no limit of login attempts.


Inget SIM 12:55

Cancel

Glue

Create Profile

Enter full name, e-mail and set your password

Full name

mail@example.com

Password

Confirm password

By creating a profile you agree to our Terms of Use and Privacy Policy.

Figure 5.6: Registering an account in the Net Lock app

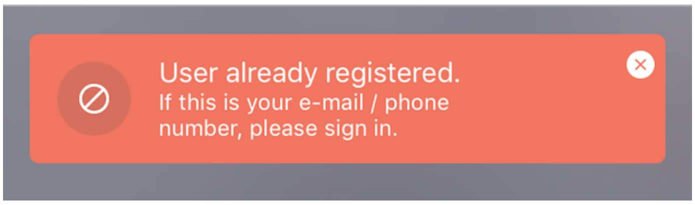
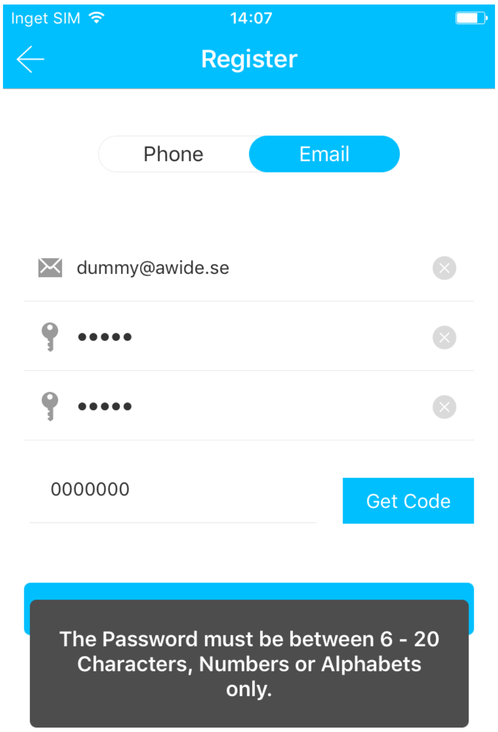

User already registered.
If this is your e-mail / phone number, please sign in.

Figure 5.7: Trying to register an already existing account in the Net Lock app.

The Ali Lock

To create a new account on the Ali Lock, the user can choose to register with a phone number or an e-mail. The user enters an e-mail or a phone number, confirms the password twice and then orders a verification code. The verification code is a six digit code that will be sent to the user via e-mail or SMS. Once received, the user can complete the registration.

The password when creating the account is limited to 6-20 characters. The password policy allows a user to use both upper- and lowercase characters and digits. It was not possible to use any special characters, the app wouldn't input any special character in the password field. This was confirmed when trying to enter a password shorter than 6 characters or greater than 20 characters, which would generate an error message, see figure 5.8.

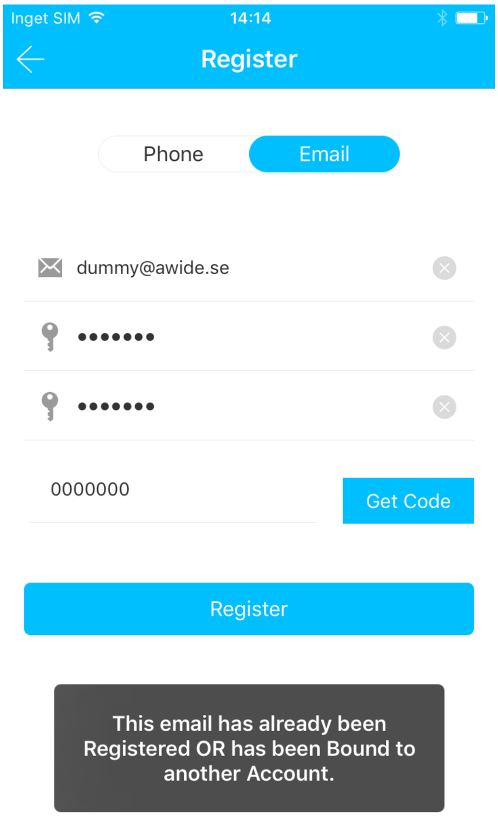


The Password must be between 6 - 20 Characters, Numbers or Alphabets only.

Figure 5.8: Password policy of the Ali Lock.

The password policy didn't force combinations of digits and/or upper- and lowercase letters. It was possible to use the common passwords *password* and *123456*, and use a password with identical characters, *111111*. When a user logins in to the app, there is no two-factor authentication. However, if the user tries to login in to a phone that he/she hasn't been logged in to before, a verification code will be sent to the e-mail/phone number. The code is required in order to login. When trying to log in with an e-mail that already exists the user will get notified that the e-mail is already associated with an account, see figure 5.9.

To test if it there was a limit of login attempts, 20 login attempts with a wrong password were conducted during 30 seconds. All login attempts were successful, concluding that it seemed to be no limit of login attempts.



dummy@awide.se

•••••••

•••••••

0000000

This email has already been Registered OR has been Bound to another Account.

Figure 5.9: Trying to register an already existing account in the Ali Lock app.

5.2.2 Password reset

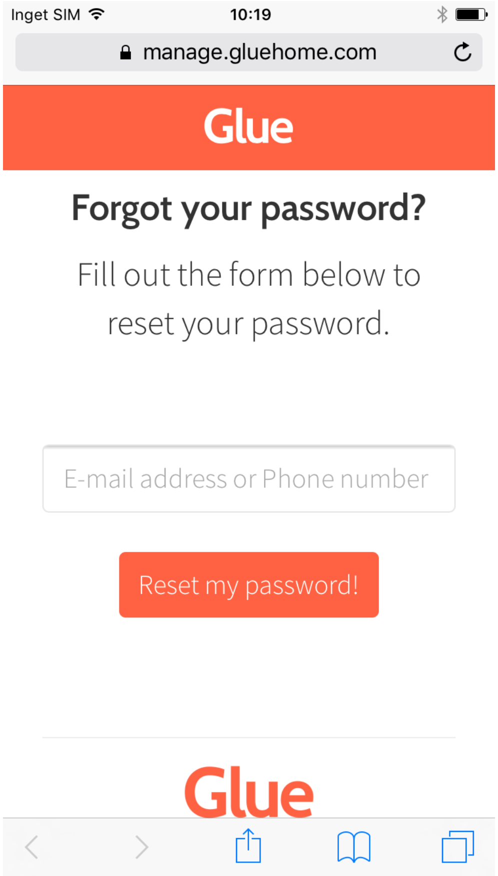
In order to test security flaws in password resets, the following tasks are investigated:

- Is it possible to order multiple password resets?
- If it is possible to order multiple password resets, does previous ordered resets get invalidated?
- How long is a password reset valid?

- Is it possible to gain knowledge about existing accounts in the password reset feature?

Net Smart Lock

When a user requests to reset the password in Net Lock app, the app will open up a web browser in which the user can enter its e-mail or phone number that is associated with the account, see figure 5.10. Once the user has requested a password reset, an e-mail will be sent to the e-mail of the account. The e-mail contains a link to a website in which the user can enter a new password. The link itself contains a parameter with a unique hashed value in order to validate the user. The hash has a length of 32 hexadecimal characters.



Forgot your password?

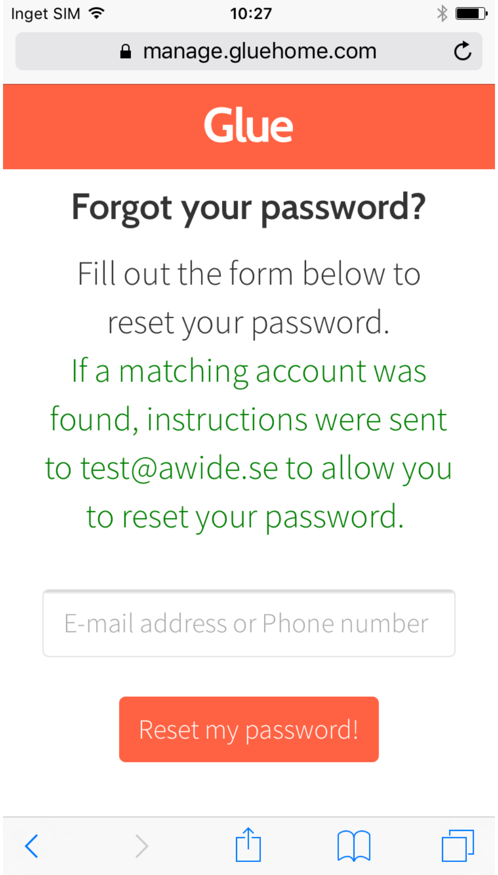
Fill out the form below to reset your password.

Figure 5.10: Password reset in the Net Lock app.

During the tests of this features, 10 password resets were requested after each other for the same account e-mail. All password resets successfully delivered e-mails with reset links to the user. This concludes that it is possible to order multiple password reset links. All of the links have a unique hash value. In order to try the invalidating of reset links, the reset link of the most recent password reset mail was used to reset the password of the user. When the user password had been reset, the second most recent link was used to try and reset the password again. It was possible to use the second password reset link to reset the password. The remaining 8 reset links were also used to try and reset the password, which were also successfully used to reset the password. The conclusion is that when a new password reset link is requested, previous links does not get invalidated. The links did however get invalidated upon use, meaning it was not possible to use the same link twice.

When requesting a new password reset, the e-mail doesn't state for how long the link is valid. By trying the reset links in intervals of 10 minutes, it was possible to narrow it down that a reset link is valid for 6 hours.

When requesting a password reset from Net, the application will tell the user that if it finds a matching account it will send instructions to that account, see figure 5.11. By doing so, it is not possible to know if a user with a specific e-mail or phone number exists or not.



Inget SIM 10:27

manage.gluehome.com

Glue

Forgot your password?

Fill out the form below to reset your password.

If a matching account was found, instructions were sent to test@awide.se to allow you to reset your password.

E-mail address or Phone number

Reset my password!

Figure 5.11: Password reset in the Net Lock app.

The Ali Lock

For the Ali Lock, the password reset is ordered from the app. The user launches the app and then presses the *Forgot Password* button. To reset the password, the user must provide an e-mail address or a phone number associated with the account. The user then provides a new password, two times to confirm it. The user then needs to order a verification code, which triggers an email sent to the users e-mail. The verification code is a six digit number and can only ordered each 60 seconds in the smartphone app. Once the verification code is entered, the user can request to reset the password, see figure 5.12.

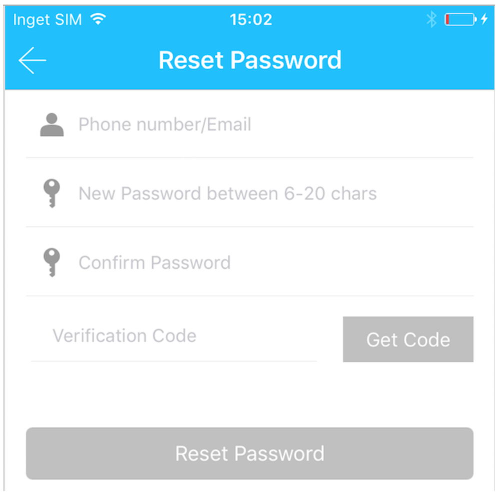


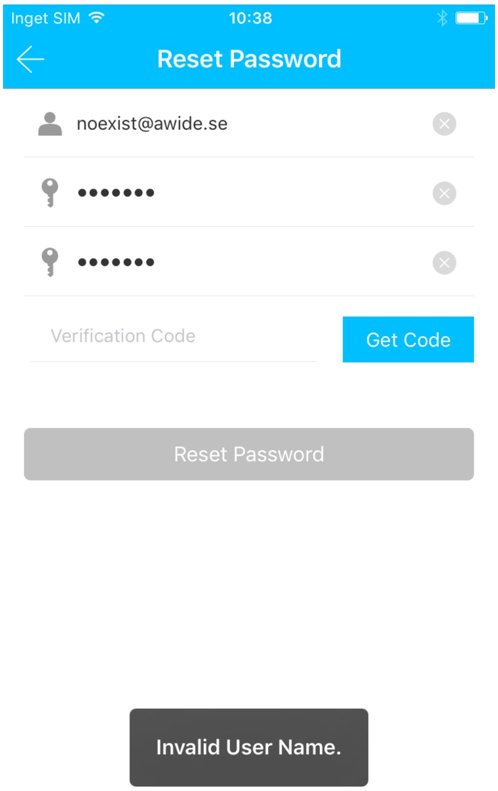
Figure 5.12: Password reset confirmation in the Ali Lock app.

Firstly a test was conducted in order to try and order multiple password resets. In the smartphone app it was only possible to do so each 60 seconds. Next step was to use the man-in-the-middle proxy lab environment described in section 4.4 to view the HTTPS traffic between the smartphone app and the cloud API. This showed that the request sent to the cloud API to request a verification code didn't require authorization. A small Pyhton script was written in order to try how many password resets could be requested at the same time. The script sent the same parameters in the request as the smartphone app did. The outcome of the test showed that it was possible to order nine verification codes before the server returned an error.

Nine password reset requests meant that nine e-mails with password reset codes was sent to the account owner. In order to try password reset code invalidation, the reset code of the most recent password reset mail was used to reset the password of the user. When the user password had been reset, the second most recent password reset code was used to try and reset the password again. It was possible to use the second password reset code to reset the password. The remaining 7 reset codes were also used to try and reset the password, which were also successfully used to reset the password. The conclusion is that when a new password reset code is requested, previous codes does not get invalidated. The codes also doesn't get invalidated upon re-use. The request sent when trying to do password reset with a password reset code didn't require authentication, since the user is requesting a forgotten password and doesn't have access to his/her account.

When requesting a password reset code, the e-mail which contains the code claims that the reset code is valid for one hour. By trying to reset the account with the code after certain time intervals, it was found that the code was in fact valid for one hour and 40 minutes. This was tried with multiple codes with the same result.

If the user enters an valid e-mail och phone number, the app will start a timer of 60 seconds until the user can order a new code. If a user enters an invalid e-mail or phone number, the app will tell the user that the account doesn't exists, see figure 5.13. This makes it possible to gain knowledge regarding if an account exists or not.



noexist@awide.se

•••••••

•••••••

Invalid User Name.

Figure 5.13: Entering a non-existing account in the Ali Lock app password reset.

The result of the password reset for the Ali lock can be summarized as follows:

- Password reset codes are six digits, meaning there is 1 000 000 different combinations for one reset code.

- Password reset codes can only be ordered every 1 minute in the app, but are not limited via the API, making it possible to order nine reset codes at the same time via the API.
- Password reset codes are valid for one hour and 40 minutes.
- When ordering a new password reset code, the previous ordered codes are not invalidated.
- There is no (obvious) limitation of sending requests to try if the password reset code is valid.
- It is possible to gain knowledge if an account exists or not.

In order to test if it was possible brute force the password reset code, a Python script was written that could send multiple password reset requests simultaneously. To not flood the server and cause a denial of service, a small test was conducted. Eight virtual machines was setup through Google Cloud in different locations around the world. Each and every machine used the Python script to do 40 requests to reset the password simultaneously. This was done in three iterations. In the slowest case, the machines had an average time of 10.7 requests per second per machine. That means 642 requests per minute and 38520 hourly requests for one machine. A reset code is active for 100 minutes, so one machine can make 64200 requests in 100 minutes. If there was only one recovery code, 16 machines would be neccessary to test all combinations within the range 0-999999 in 100 minutes. To avoid flooding the server, it is worth waiting three seconds every hundred request to avoid being blocked by the API server. In that case, the average time is 8 requests per second, which implies 48000 requests per 100 minutes, per machine. Then 21 machines would be needed to cover the entire range 0-999999 within the time frame of 100 minutes. However, it is possible to request 9 recovery codes instead of one, which significantly improves the ability of finding the right code faster, provided the codes are fairly evenly spread. The conclusion is that it is theoretically possible to brute force the password reset code. The attacker needs to know the e-mail or phone number associated with the account in order to request password reset codes and then brute force the password reset. By doing so, the attacker can gain access to the users account. This attack was not limited to the app itself but exists within the platform which multiple app developers use. This test was never conducted in full scale to not violate Swedish law, but instead a smaller theoretical test was conducted. The findings have been responsible disclosed to the owners of the platform, which have been given a 90 days grace period to patch the vulnerabilities.

It should be noted that the first time a user logins in to his/her account on a new smartphone, they need to enter a verification code very similar to the password reset code. This would prevent an attacker to login in to an account of which the attacker has reset the password and where the account hasn't been used on the attackers phone before. However, the procedure for the verification code is the same as with the password reset code, meaning that the attack for brute forcing the password reset code is also applicable for brute forcing the verification code. Hence, it is still possible for an attacker to gain access to the account, but the attacker needs to do the brute force two codes: one for the password reset and one for the sign in on a new device.

5.3 Man-in-the-middle attack

Recalling the man-in-the-middle attack in section 4.2.3, the following is tested and described in the following three subsections:

- Is it possible to elevate privilege of a guest user?
- Is it possible to spoof a successful login?
- Is it possible to add already registered locks to an second account without authorization?

5.3.1 Elevation of privilege

Net Smart Lock

When the user signs in to the app, it fetches the role of the current user from the cloud API endpoint */api/Accesses*. The endpoint returns a JSON object with the user ID and it's role, see listing 5.1.

Listing 5.1: Response of */api/Accesses*

```
{
    ...
    "Role": {
        "Id": "0da029b4-4b76-4c75-a16b-3f64619219da",
        "IsCompanyRole": false ,
        "ParentId": "d85de7e2-5582-4a1f-9fc7-ffb3e2008e85",
        "Privileges": [],
        "PrivilegesMaskSum": 0,
        "RoleName": "Standard User"
    },
    "RoleId": "0da029b4-4b76-4c75-a16b-3f64619219da",
    ...
    "UserId": "00c94089-1418-42c6-afb1-27269b78bf63"
}
```

Roles are fetched from the endpoint */api/Roles* when the app is launched. The response contains the id and description of all available roles. By using mitm-proxy, it's possible to change the response of */api/Accesses* to use the role id of the owner role instead of the guest role. To make the user a owner of a lock, the response of endpoint */api/<lockid>/Accesses* also needs to be modified with the owner role id. By doing so it's possible to, at least esthetically, elevate the privilege of an guest user. However, trying to execute tasks that are only available for owners, for instance invite a new user to use the lock will result in a 403 Forbidden response with the following body: *Unauthorized invite, wrong lock or user*. It was not possible to access any logs, and it was not possible to open the lock with only Bluetooth enabled on the smartphone. It seems that the lock makes a validation through the hub when an app tries to control the lock, preventing any elevation of privilege of unauthorized users.

The Ali Lock

After logging in to the app, the user role is fetched from the endpoint */check-/syncData*. The endpoint has an entry, called *userType* in the returning JSON object, which defines the role of the user. The admin user has userType 110301 and a guest user has userType 110302. By using mitm-proxy it is possible to change the response of */check/syncData* to return the userType of the admin role instead of the guets user role. This showed that it was possible to, at least esthetically, elevate the privilege of an guest user. However, when trying to use admin features, such as inviting new user to become admins, the smartphone app will check the privileges of the user against cloud API and return *Permission denied*. Another attempt was to set a new passcode for the lock, and spoof a successful response. When a new passcode is set, the smartphone app makes an API call to the */keyboardPwd/add* endpoint which, if successful, returns a *keyboardPwdId* for the code. The attack was executed by first changing the userType response from */check/syncData* to the owner role, and then change the response from */keyboardPwd/add* to be an integer incremented by one based on a legit request. The app gave an *Operation Successful* message when this was executed, but the code didn't work to open the lock. Hence, this attack was not successful. In similarity with the Net Smart Lock, it seems the Ali Lock makes a validation through the hub when an app tries to control the lock, preventing any elevation of privilege from unauthorized users.

5.3.2 Spoofing login

Net Smart Lock

The Net Home app uses the basic HTTP authorization scheme header[51], and encrypts the API call with SSL/TLS. The basic HTTP authorization scheme provides the user credentials in plain text with base64 encoding, Every API request made requires that the authorization details are specified in the header. When a user logs in to the app, the app makes an request to the */api/Users* endpoint with the basic authorization header. The response is a JSON object with details about the user. Spoofing the response would be possible, however, after a successful login the app fetches data about the lock, roles and accesses. All of these endpoints also require the HTTP authorization header, meaning that spoofing the */api/Users* endpoint would still not make the user authorized to access the other endpoints.

The Ali Lock

The Ali Lock login procedure makes a call to the */api/login* endpoint with the e-mail/phone number and the password as a URL Encoded form, where the password is hashed with MD5. The API call is encrypted with SSL/TLS. The response will return an access token which is used as an header with every call that the smartphone does to the API. The token is renewed every time a user re-authenticates in the smartphone app. In order to test the validity of the token, a spoofed token was specified in the header and a API called was made to access the */api/Users* endpoint. The result was a *Permission denied* response from the API.

5.3.3 Adding an already registered lock to a second account

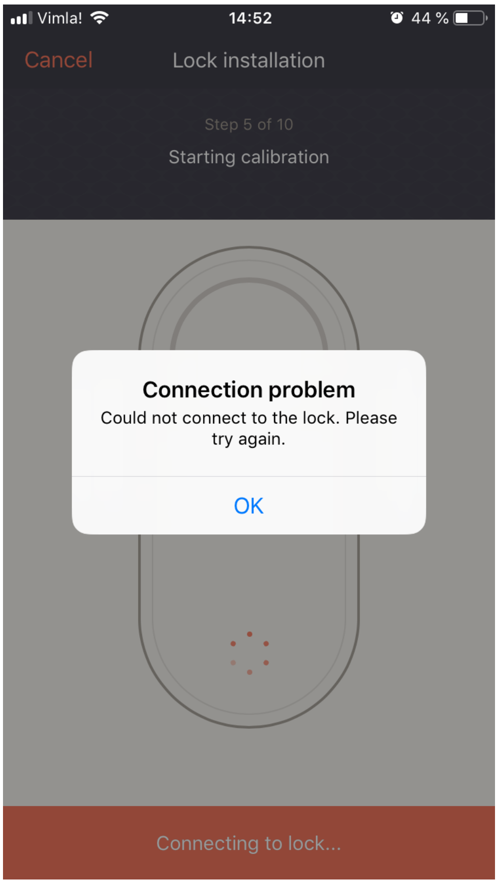
Net Smart Lock

When registering a new lock in the Net Home app, the app makes an API call to */api/Locks*. It specifies a JSON object with the locks serial number, description, firmware version and status. If the action was successful, the repsonse returns a 201 created status and a JSON object with general information about the lock, see listing 5.2. If the action wasn't successful, the API returns *Lock already registered*.

Listing 5.2: Successful response of */api/Locks*

```
{
    "AvailableFirmwareVersion": null,
    "BatteryStatus": null,
    "BatteryStatusAfter": null,
    "BatteryStatusBefore": null,
    "CommandlistVersion": 12,
    "Created": "2019-05-07T11:51:51.2920162Z",
    "Description": "GL04A.AL1751.0094",
    "FirmwareVersion": 32,
    "HardwareVersion": "3",
    "HubId": null,
    "Id": "07152a0e-1099-4926-813a-46d95b34d54b",
    "ImageId": null,
    "ImageUrl": null,
    "IsLogPublic": false,
    "IsNotificationsPublic": false,
    "PositionX": null,
    "PositionY": null,
    "SerialNumber": "GL04A.AL1751.0094",
    "Status": 1,
    "SystemConfig": null,
    "TimeZone": null
}
```

To test if it's possible to register an already registered lock to a second account, the failed response of */api/Locks* is spoofed by mitm-proxy with a successful response with the serial number of the lock. This made it possible to bypass the error that the lock is already registered (recall figure 5.1a). However, when trying to calibrate the lock the app froze since the API endpoint */api/Locks/<lockid>/CommandList* would return that there was no available commands for the user of the current lock. To expand the attack further the command API endpoint */api/Locks/<lockid>/CommandList* was spoofed with a response from a valid API call. The same applied to *api/Locks/<lockid>/AccessKeys*, *api/Locks/<lockid>/Accesses* and *api/Locks/<lockid>* which are all called upon during the registration of the lock. However, it was still not possible to get passed the calibration phase of the lock. The app would simply freeze for a few seconds and then return an error that there was a problem with connecting to the lock, see figure 5.14.



Vimla! 14:52 44 %

Step 5 of 10

Starting calibration

Connection problem

Could not connect to the lock. Please try again.

Figure 5.14: Error message when trying to add an already registered lock.

The Ali Lock

For the Ali Lock, there was no requests made to the cloud API when trying to add a second lock. Hence, it's most likely that the smartphone app tries to communicate with the lock using Bluetooth only. Since the Bluetooth communication is out of scope for this thesis, there is no more results to report.

Chapter 6

Discussion

In the following chapter, the method and results of the attacks are discussed.

6.1 Methodology

6.1.1 Related Work

The related work section aimed to be comprehensive in finding attacks on smart locks and similar IoT devices. Efforts were made to find recent research, but since the technology of smart locks is young, not many white papers are available. Instead security conferences and security blogs on the internet was used as sources. For the few white papers available, for instance the attack regarding traffic analysis in section 2.4.6, it was not possible to know which locks that had been tested. Since the attack was successful, the author chose not to disclose the manufacturer of the lock.

Due to the short timespan of the thesis, it's not feasible to aim to find and take every possible research into account. Hence, there might be some attacks that have been overlooked. It should however be noted that remarkable efforts was made to find relevant research.

6.1.2 Threat Model

The threat model process used in the thesis is based on the work of ARM[6]. The process is made specifically for IoT products and overall it was good to start by identifying assets, adversaries and attack surfaces. To create the STRIDE threat model, Microsoft Threat Modeling Tool was used. This created a foundation of how the ecosystem of smart locks is constructed. However, the Microsoft Threat Modeling Tool is generic and not always straight forward when adapted to IoT devices. Some of the threats generated was not applicable to the ToE, and some threats could be hard to interpret in the context of the ToE. Hence, it's recommended to use a threat modeling tool which is more focused on IoT devices in future work.

Due to the timespan of the thesis, it would not be feasible to investigate all 71 threats from the threat model. Instead a focus was made on the threats that had already been proven to have successful attacks in previous work. The small scope can be extended in future work by investigating further into the threats, develop new attacks based on the threats, and adapt the attacks to other locks. Previous work have found vulnerabilities in the Bluetooth communication of the smart locks, which has not been tested in this thesis. Neither has physical/hardware hacking and tampering been considered.

The selection of locks was limited to two locks, but as stated in table 4.1, there are many more locks to test. In order to draw a conclusion regarding the security of smart locks today, future work needs to be more comprehensive by applying conducting attacks to more locks.

6.2 Results

6.2.1 State consistency attack

Net Smart Lock

For the state consistency attack, the registration of the same lock twice wasn't successful. This means that it's not possible for an attacker to be nearby the lock and add it to it's own account.

The time limited and non-time limited accounts were able to successfully execute the state consistency attack. However, they were only successful without using the hub. The same applied for the log evasion. In the work of Ho et al. [15] where the state consistency attack was discovered, none of the smart locks had a separate hub. Since the attack was not successful with the hub, it can be concluded that the hub adds some protection regarding the state consistency attack on the Net Smart Lock.

As remarked in the result, logs are removed when a user revokes it's own access or gets revoked by the owner. This raises some serious security concerns regarding logs. The purpose of logs is to keep the owner updated regarding interactions with the lock, but this is not possible if the logs are deleted.

The Ali Lock

The Ali lock also couldn't add a lock to a second account. The Ali Lock differed in comparison with the Net Smart Lock in the time limited and non-time limited accounts. The non-time limited account and log evasion attack was successful regardless if the the hub was used or not, whereas the time-limited account could not be evaded at all. Compared to the Net Lock, when a malicious user was evading the revocation of the Ali Lock, it showed that the key had not yet been deleted. However, the hub didn't prevent the revocation attack. The built-in clock in the Ali Lock did prevent time-limited access revocation. Such feature could be used on the Net Smart Lock in order to prevent the revocation attack when not using the hub.

6.2.2 Brute force attack

Net Smart Lock

The Net Smart Lock password policy varied between the creation of the account and the change password view in the smartphone app. The only restrictions of the policies was the length, which makes it easy for users to select common passwords. There was also no multi-factor authentication. Allowing short passwords, not having limited login tries, and not having any multi-factor authentication makes it possible for an remote attacker to execute a password enumeration attack. To complement the attack it is possible gain knowledge about whether a user account exists or not when trying to add a new account using those details.

No apparent vulnerabilities could be found in the password reset of the Net Smart Lock. In order to brute force the password recovery link, the attacker needs to brute force 32 hexadecimal characters. With a speed of 6000 tries per minutes, it would take $3.83 * 10^{14}$ years to brute force the password link.

The Ali Lock

Because common passwords are not prohibited and can be as short as 6 characters and a maximum of 20 characters, and no special characters are allowed, the authentication mechanism for the Ali Lock might be exposed to password enumeration attacks. An attacker only needs to know the phone number or the e-mail of the victim to be able to execute the attack. Gaining knowledge about the phone number or the e-mail of the user is possible since the app will let the user know if such details exist when trying to create an account using those details.

Regarding the possible password reset attack on the Ali Lock, the developers of the platform were contacted. The first contact was made to a support e-mail which then was redirected to the CTO of the platform. The CTO answered the first e-mail which claimed that vulnerabilities had been found in the platform. However, when asked how the CTO would like the details of the vulnerability to be delivered, the CTO stopped responding to e-mails. After 4 weeks of reminders without any repsonse, on the 18th of April, both the CTO and the support of the platform was informed about the responsible disclosure policy of KTH. The e-mail was conducted as follows (the name of the platform has been masked):

```
To whom it may concern,
We have since March 15 attempted to report a vulnerability in the XXX Platform
(please see email thread below). The policy at the Royal Institute of Technology
is to grant vendors a 90 days grace period in which to patch the vulnerability.
After 90 days, however, we will disclose the vulnerability publicly regardless
of the availability of patches. During these 90 days, we are prepared to assist
you in remediating the vulnerability.
Please do let us know how you wish to proceed.
```

6.2.3 Man-in-the-middle-attack

Regarding elevation of privilege it was not possible to use the an attack like demonstrated by Jmaxxz [27] in order to elevate privilege by spoofing a response from the cloud API. This applied to both of the locks. It seems like the hub adds extra security when validating the spoofed credentials.

Spoofing login was not successful on any of the devices. The Net Smart Lock requires the HTTP authorization header to be set for each request, and the Ali Lock receives an token during login which could not be spoofed.

When trying to spoof the adding of the lock to a second account, it seemed like the Net Lock did some verification with the cloud API via the hub, as spoofing the responses from the cloud API to the smartphone API didn't work. As for the Ali Lock, there seems to be no requests made at all from the smartphone. Hence it was not possible to spoof. A suggestion for future work is to investigate if the locks validate the adding of a new lock via the hub, or via Bluetooth. If via Bluetooth, it might be possible to spoof or replay a successful Bluetooth command. If via the hub, it's hard to read the traffic between the hub and the cloud API because it's encrypted with TLS. It would be necessary to bypass the encryption.

Chapter 7

Conclusion

Since locks are a great part of protecting properties from unauthorized access, they need to be considered in as many aspects as possible before claiming to be secure. One example is the previous work of Rose et al. [28], where no vulnerabilities were found in the Kwikset smart lock, but the lock could be opened by a simple flat head screwdriver. As for the physical design of the smart locks, locks which replaces the deadbolt makes it easier for an attacker to know that the owner is using a smart lock compared to a lock that is mounted on an existing deadbolt.

It can be argued that smart locks doesn't add any security, but instead add new attack surfaces compared to traditional locks. That being said, they do add a lot of functionality and can ease the everyday life. On the other hand it can be argued that it's safer to use virtual keys rather than physical, since it's easier to revoke a digital key compared to replacing the lock if a physical key is lost. However, current research shows that flaws and vulnerabilities exists in smart locks.

In this report we've found several vulnerabilities and flaws that could pose as security threats in smart locks. Altough only applied on a small set of smart door locks, the research can conclude that previous discovered vulnerabilities in smart door locks do exists in todays locks. Future research is necessary in order to enlighten and show the importance of security in smart locks.

Bibliography

[1] A.J. Bernheim Brush et al. "Home Automation in the Wild: Challenges and Opportunities". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '11. Vancouver, BC, Canada: ACM, 2011, pp. 2115–2124. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979249. URL: http://doi.acm.org.focus.lib.kth.se/10.1145/1978942.1979249.

[2] M. R. Alam, M. B. I. Reaz, and M. A. M. Ali. "A Review of Smart Homes—Past, Present, and Future". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.6 (Nov. 2012), pp. 1190–1203. ISSN: 1094-6977. DOI: 10.1109/TSMCC.2012.2189204.

[3] Charlie Wilson, Tom Hargreaves, and Richard Hauxwell-Baldwin. "Smart Homes and Their Users: A Systematic Analysis and Key Challenges". In: *Personal Ubiquitous Comput.* 19.2 (Feb. 2015), pp. 463–476. ISSN: 1617-4909. DOI: 10.1007/s00779-014-0813-0. URL: http://dx.doi.org/10.1007/s00779-014-0813-0.

[4] Sarah Mennicken, Jo Vermeulen, and Elaine M. Huang. "From Today's Augmented Houses to Tomorrow's Smart Homes: New Directions for Home Automation Research". In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp '14. Seattle, Washington: ACM, 2014, pp. 105–115. ISBN: 978-1-4503-2968-2. DOI: 10.1145/2632048.2636076. URL: http://doi.acm.org.focus.lib.kth.se/10.1145/2632048.2636076.

[5] Nazmiye Balta-Ozkan et al. "Social barriers to the adoption of smart homes". In: *Energy Policy* 63 (2013), pp. 363–374. ISSN: 0301-4215. DOI: https://doi.org/10.1016/j.enpol.2013.08.043. URL: http://www.sciencedirect.com/science/article/pii/S0301421513008471.

[6] Brian Clinton and Suresh Marisetty. *How Threat Modeling Can Secure Your Next IoT Product in 5 Steps*. [Online; accessed March 04, 2019]. Oct. 2018. URL: https://pages.arm.com/threat-models-webinar-recording-typ.html.

[7] Michael Howard and David LeBlanc. *Writing secure code*. Pearson Education, 2003.

[8] Tony UcedaVelez and Marco M Morana. *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis*. John Wiley & Sons, 2015.

[9] Sjouke Mauw and Martijn Oostdijk. "Foundations of Attack Trees". In: *Information Security and Cryptology - ICISC 2005*. Ed. by Dong Ho Won and Seungjoo Kim. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 186–198. ISBN: 978-3-540-33355-5.

[10] Justitiedepartementet L5. *Brottsbalk (1962:700)*. [Online; accessed March 08, 2019]. 2014. URL: https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/brottsbalk-1962700_sfs-1962-700.

[11] Justitiedepartementet L1. *Lag (2018:558) om företagshemligheter*. [Online; accessed March 08, 2019]. URL: https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/lag-2018558-om-foretagshemligheter_sfs-2018-558.

[12] Justitiedepartementet L3. *Lag (1960:729) om upphovsrätt till litterära och konstnärliga verk*. [Online; accessed March 08, 2019]. 2018. URL: https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/lag-1960729-om-upphovsratt-till-litterara-och_sfs-1960-729.

[13] Bovalls Dörrbyggeri. *Yale Doorman Lock*. [Online; accessed January 24, 2019]. 2019. URL: https://bovalls.com/produkt/tillbehor/dorrtillbehor/yale-doorman-elektroniskt-dorrlas/.

[14] IF Säkerhetsbutik. *Glue Smart Lock*. [Online; accessed January 24, 2019]. 2019. URL: https://www.if-sakerhet.se/media/catalog/product/cache/2c29cda20e50098bc593ba7672cf800e/g/l/glue-smart-kit-12.jpg.

[15] Grant Ho et al. “Smart Locks: Lessons for Securing Commodity Internet of Things Devices”. In: *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ASIA CCS ’16. Xi’an, China: ACM, 2016, pp. 461–472. ISBN: 978-1-4503-4233-9. DOI: 10.1145/2897845.2897886. URL: http://doi.acm.org.focus.lib.kth.se/10.1145/2897845.2897886.

[16] Z-Wave Alliance. *About Z-Wave Technology*. [Online; accessed March 06, 2019]. Z-Wave Alliance. URL: https://z-wavealliance.org/about_z-wave_technology/.

[17] Friday Labs. *Seamless integration with Apple HomeKit*. [Online; accessed February 08, 2019]. 2019. URL: https://www.fridaylabs.net/views/homekit.html.

[18] ID Lock. *User Manual - ID Lock 150*. [Online; accessed February 28, 2019]. ID Lock AS. URL: https://idlock.no/wp-content/uploads/2018/07/20180516-IDL150-manual-v1-2.pdf.

[19] ASSA Abloy. *User Manual*. [Online; accessed February 28, 2019]. Yale Doorman. URL: http://www.lasmontage.se/content/uploads/products/manual/Doorman_User_Manual.pdf.

[20] E. Fernandes, J. Jung, and A. Prakash. “Security Analysis of Emerging Smart Home Applications”. In: *2016 IEEE Symposium on Security and Privacy (SP)*. May 2016, pp. 636–654. DOI: 10.1109/SP.2016.44.

[21] Samsung SmartThings. *Door Locks*. [Online; accessed February 20, 2019]. 2019. URL: https://www.smartthings.com/products/-/filter/categories/door-locks.

[22] M. Nobakht et al. “Permission Analysis of Health and Fitness Apps in IoT Programming Frameworks”. In: *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. Aug. 2018, pp. 533–538. DOI: 10.1109/TrustCom/BigDataSE.2018.00081.

[23] S. Gusmeroli, S. Piccione, and D. Rotondi. “IoT Access Control Issues: A Capability Based Approach”. In: *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. July 2012, pp. 787–792. DOI: 10.1109/IMIS.2012.38.

[24] Yunhan Jack Jia et al. “ContexloT: Towards Providing Contextual Integrity to Appified IoT Platforms.” In: *NDSS*. 2017.

[25] Elisa Bertino et al. “Internet of Things (IoT): Smart and Secure Service Delivery”. In: *ACM Trans. Internet Technol.* 16.4 (Dec. 2016), 22:1–22:7. ISSN: 1533-5399. DOI: 10.1145/3013520. URL: http://doi.acm.org.focus.lib.kth.se/10.1145/3013520.

[26] M. Ye et al. “Security analysis of Internet-of-Things: A case study of august smart lock”. In: *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. Apr. 2017, pp. 499–504. DOI: 10.1109/INFCOMW.2017.8116427.

[27] Jmaxxz. *DEF CON 24 - Jmaxxz - Backdooring the Frontdoor*. Youtube. 2016. URL: https://www.youtube.com/watch?v=MMB1CkZi6t4.

[28] Anthony Rose et al. “SECURING BLUETOOTH LOW ENERGY LOCKS FROM UNAUTHORIZED ACCESS AND SURVEILLANCE”. In: *Critical Infrastructure Protection XI*. Ed. by Mason Rice and Sujeet Shenoi. Cham: Springer International Publishing, 2017, pp. 319–338.

[29] M. Dell’ Amico, P. Michiardi, and Y. Roudier. “Password Strength: An Empirical Analysis”. In: *2010 Proceedings IEEE INFOCOM*. Mar. 2010, pp. 1–9. DOI: 10.1109/INFCOM.2010.5461951.

[30] Jmaxxz. *The August Smart Lock’s not so smart password reset (Part 2)*. [Online; accessed February 27, 2019]. Mar. 2015. URL: https://jmaxxz.com/blog/?p=498.

[31] Robin Gustafsson and Linus Janstad. “Säkerhetsutvärdering av smarta dörrlås utifrån ett kommunikationsperspektiv”. In: (2015).

[32] Noah Apthorpe, Dillon Reisman, and Nick Feamster. “A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic”. In: *arXiv preprint arXiv:1705.06805* (2017).

[33] Jeffrey Walton et al. *Certificate and Public Key Pinning*. [Online; accessed February 21, 2019]. Oct. 2018. URL: https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning.

[34] Kristi Rawlinson. *HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack*. [Online; accessed March 06, 2019]. HP Development Company, L.P. July 2014. URL: https://www8.hp.com/us/en/hp-news/press-release.html?id=1744676.

[35] Microsoft. *Getting started with the Threat Modeling Tool*. [Online; accessed April 16, 2019]. 2018. URL: https://docs.microsoft.com/en-us/azure/security/azure-security-threat-modeling-tool-getting-started.

[36] The Open Web Application Security Project. *Implement Proper Password Strength Controls*. [Online; accessed April 24, 2019]. URL: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Authentication_Cheat_Sheet.md#password-complexity.

[37] The Open Web Application Security Project. *Implement Proper Password Strength Controls*. [Online; accessed May 03, 2019]. URL: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Authentication_Cheat_Sheet.md#prevent-brute-force-attacks.

[38] The Open Web Application Security Project. *Implement Proper Password Strength Controls*. [Online; accessed May 03, 2019]. URL: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Authentication_Cheat_Sheet.md#authentication-and-error-messages.

[39] The Open Web Application Security Project. *Implement Proper Password Strength Controls*. [Online; accessed May 03, 2019]. URL: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Forgot_Password_Cheat_Sheet.md.

[40] ASSA ABLOY. *Yale Doorman and Aptus System*. [Online; accessed April 18, 2019]. URL: https://www.aptus.se/en/site/aptus/products/yale-doorman-and-aptus-system/.

[41] ASSA ABLOY. *Doorman*. [Online; accessed April 18, 2019]. URL: https://www.yalelock.com/en/yale/com/smart-living/products/smart-door-locks/yale-doorman/.

[42] ID Lock AS. *Nyhet ID Lock 150*. [Online; accessed April 18, 2019]. 2018. URL: https://idlock.se/id-lock-150/.

[43] Danalock International ApS. *Danalock V3 Smart Lock*. [Online; accessed April 18, 2019]. 2019. URL: https://danalock.com/products/danalock-v3-smart-lock/.

[44] Lockitron. *Bolt*. [Online; accessed April 18, 2019]. URL: https://store.lockitron.com/products/bolt-bridge.

[45] Wattle. *DOOR LOCK*. [Online; accessed April 18, 2019]. URL: https://wattle.com/product/door-lock/.

[46] August Home. *Smart Lock Pro + Connect*. [Online; accessed April 18, 2019]. 2019. URL: https://august.com/products/august-smart-lock-pro-connect/.

[47] Inc. CANDY HOUSE. *SESAME*. [Online; accessed April 18, 2019]. 2018. URL: https://candyhouse.co/.

[48] Nuki Home Solutions. *NUKI Smart Lock 2.0*. [Online; accessed April 18, 2019]. 2019. URL: https://nuki.io/en/smart-lock/.

[49] Friday Labs. *Meet Friday*. [Online; accessed April 18, 2019]. URL: https://www.fridaylabs.net/views/home-page.html.

[50] Aldo Cortesi, Maximilian Hils, and Thomas Kriechbaumer. *mitmproxy is a free and open source interactive HTTPS proxy*. [Online; accessed May 03, 2019]. URL: https://mitmproxy.org/.

[51] J. Reschke. *The 'Basic' HTTP Authentication Scheme*. [Online; accessed May 06, 2019]. Internet Engineering Task Force (IETF), Oct. 2015. URL: https://tools.ietf.org/html/rfc7617.

TRITA EECS-EX-2019:300

www.kth.se