Password Managers

- Password Manager on a device with an external database -

Master Thesis

des1010f16

Aalborg University
Electronics and IT

AALBORG UNIVERSITY
STUDENT REPORT

Electronics and IT
Aalborg University
http://www.aau.dk

Title:
Password Managers

Theme:
Master thesis

Project Period:
Summer Semester 2016

Project Group:
des1010f16

Participant(s):
Horatiu-Laurentiu Vultur

Supervisor(s):
Rene Rydhof Hansen
Mads Christian Olesen

Copies: 4

Page Numbers: 41

Date of Completion:
May 29, 2016

Abstract:

In these days there are lots password managers like: Dashlane, 1Password, LastPass. All of these are web-based password managers. It is known that these password managers suffer different web attacks, therefore we will present a design for a password manager which is not web-based. In the first part we will present the decisions for choosing this approach and in the last part we will present the design and the implementation of this design.

Contents

Preface

This paper represents the master thesis for the group des1010f16, Embedded Software Systems.

Aalborg University, May 29, 2016

Horatiu Laurentiu Vultur
<hvultu14@student.aau.dk>

Summary

In the current days there are lots of attacks against different websites. In the year 2014, there was made public a list of 5 million passwords and some of them were available. The problem with these passwords is that they are easy to crack, because the users tend to choose simple, easy to remember passwords. To help the users to remember all the passwords there are password managers, these are like databases which stores all the passwords and the user needs only one password to access them. The problem with these password managers is that they are vulnerable to different attacks and some of them already leak private information.

This paper will present first some of the existing attacks against password managers. It will present attacks more specific to hardware based password managers, like: Side channel attacks, Read Memory, Key logging attacks, Man in the middle, Malwares, Exhaustive search and Social Engineering attacks. Then it will propose a password manager which uses a hardware device to store the master password and a phone to store the encrypted passwords. The paper will try to compare the proposed solution with other possible solution and will argue why each feature is selected or not. After that it presents the implementation of the design, this design has some features like: store the passwords encrypted with the AES 128 bit key, separate master password from database, the hardware device emulates a keyboard, has button to improve the security, the communication between the phone and the device is encrypted and made through Bluetooth, it contains a password generator, the master password is pass through a MD5 function and after that it is stored in the memory.

In the end it presents existing attacks that can be used against the proposed solution and what is their effect. Therefore this solution can be used by any kind of user and it can store any kind of passwords, from different websites, folders, documents etc, but the user has to know that this information is not 100% secure. Even though the possible attacks required many resources and knowledge.

Chapter 1

Introduction

The most common way of authentication is textual based. Even if it is the most used one it is not the safest one [12]. In the year 2012 the websites LinkedIn and eHarmony were attacked and some private information was stolen. Also in the year 2014 on the Russian BitCoin forum, a list of 5 million credentials showed up [1]. The majority of the passwords were in clear text. These accounts were Gmail accounts, but some of them weren't use or the passwords for these were old. Even though between the accounts that were valid, there were accounts that belong to people from important companies like: PayPal, Twitter, Microsoft, Uber, BBC, Facebook, Yahoo, Google [2]. The median craking time of the passwords was 22 seconds, so we can see that the users tend to use simple passwords that easy to remember. Not like strong password which are harder to remember. Another bad habit of the user is that, if they have a strong password for an account, he tends to use the password for all the other accounts.

To help the user with these problems, there are password managers. Password managers can be a software or a hardware device that helps the user to store and organize the passwords. They have multiple advantages, but the main one is that the user needs to remember only a master password to access all the other passwords. In the previous semester report we presented different existing solution like LastPass, Dashlane, 1Password and different attacks like: Brute force attacks, KeyLoging, Sql Injection, Man in the middle, Social engineering attacks [22].

The first part of this report contains an introduction to general attacks that can be used against password manager (Chapter 2). We will focus on hardware attacks, because the proposed solution will be hardware based. For example we will not take in consideration attacks like: SQL Injection, Cross site scripting and Cross site request forgery. We will take in consideration attacks like: KeyLogging attacks, Man in the middle, Social engineering attacks, Side channel attacks. Also this chapter presents the steps the attacker needs to take to succeed the attack. In the next chapter we will present our design and present which part can be used against which attack. In Chapter 4 presents the implementation of the design and the tests for the implementation. In the last part (Chapter 5) we will present some extra feature that can be added in the future and in the end we will present the conclusion (Chapter 6).

1.1 Problem definition

Is it possible for some web based password managers to leak private information. Already LastPass, one of the existing password managers leaked credentials to unauthorized parties [13] and according to David Silver at. there are known attacks against other password managers [11]. Another type of the password managers is software based. The problem with these password managers is that they can be used only on one computer. This paper presents the design and the implementation of the solution proposed by us [22]. The advantage of this hardware based password manager is that it can be used also for other applications (games, files, folders, etc) and can be used on multiple computers. This solution will prevent all internet-based attacks.

Chapter 2

Attacks

This chapter presents attacks against password managers in general. For each attack we will present the steps the attacker needs to take to succeed and mention if the proposed solution is designed to counter the attack.

2.1 Side channel attacks

The interaction between the software and hardware generates different information which can be monitored by an attacker. This information is known as side channel information. For example side channel information can be the amount of power the hardware is consuming, the sound it makes, or the time it needs to process different operations. The attack which is based on this information is known as side channel attack [24].

According to YongBin Zhou the side channel attacks can be classified in three categories [24]:

- control over computation
- ways of accessing the module
- methods used in the analysis process

Furthermore the control over computation can be divided in two:

- the attacker extracts information without interfering with the system
- the attacker generates some changes and based on these the system can react or not

Similarly the ways of accessing the module can be divided in:

- invasive attacks - attacker accesses directly the internal components of the system

- semi-invasive attacks - attacker access the system without damaging outside layer
- non-invasive attacks - attacker just observes the information leak from the system

Similarly the methods used in the analysis process can be divided in two:

- simple side channel attack - exploits the output that depends mainly on performed operation
- differential side channel attack - exploits the correlation between the data instantaneous of side channel leakage of the devices

The side channel attacks are already known to be successful against different cryptographic algorithms like: AES, DES, Misty1, RSA [21]. There are the following known attacks. Timing attacks are a way of obtaining useful information by measuring the time it takes the user to carry out cryptographic operations. This type of attack was used against RSA or servers running OpenSSL [21]. Fault attack presumes that the system can leak information if the hardware is not operated reliably. For example hardware faults and errors occured during the operations of cryptography can seriously affects the security. Power analysis attacks are based on the power consumption of the device. This power consumption of the device can offer lots of information regarding the operation that takes place and the parameters that are used. These attacks are successful against smart cards or other dedicated embedded systems storing secret key. [24]

For this type of the attack, the attacker can have multiple options to fulfill the attack. In case he has access to the device, he can do an invasive attack by reading the internals of the device. The attack is possible but the attacker needs to have special tools to fulfill this. In case the attacker doesn't have access to the device, he can do a non-invasive attack in which he can try to read the power consumption or the time needed for each kind of operation. Therefore this type of attack is possible but requires resources and good knowledge.

2.2 Read memory - EEPROM

The Figure 2.1 shows a transistor. These transistors compose the EEPROM memory.

It has the following elements:

- **gate** - terminal on which voltage is applied
- **drain** - terminal on which voltage is drained out
- **source** - terminal on which input voltage is applied

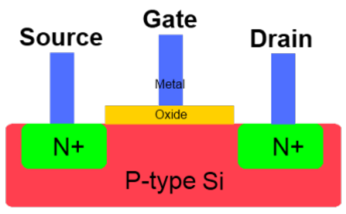


Figure 2.1: Transistor [15]

When high voltage is applied to the gate, the electrons flow in the P layer therefore the voltage from source can go to the drain. When a low voltage is applied, the electrons from the gate can't go in the P layer and therefore the voltage can't go from source to the drain. Usually the EEPROM can store the data in memory for only 40 years, but also they can loose the information if the memory is heat up to 125 C. An attacker can change the content of the memory by heating it up with a laser-diode and thus making the device unusable for the user.

For this type of the attack, the attacker needs to have access to the device. After he has access he needs a special device to read or to change the EEPROM memory. The solution can't defend against this type of attack, the attacker will manage to read the EEPRM memory. But the attacker requires some special resources to succeed it.

2.3 Key Logging attacks

These attacks are also known as keylogging or keyboard capturing. The attack consists in recording or logging keyboard's key strokes in such a way that the user who is using it doesn't know that his actions are monitored [3].

2.3.1 Software-based keyloggers

The main purpose of these programs is for troubleshooting, but the attackers can use them to obtain private information. These keyloggers can be divide in:

- **hypervisor-based**: is a malware which runs under the OS
- **kernel-based**: if it has root access, they can't be detected
- **API-based**: it hooks the keyboard API in an application therefore it will always receive events each time the keyboard is pressed

2.3.2 Hardware-based keyloggers

The main advantage of these hardware keyloggers is that they can't be detected by any software. These keyloggers can be divided in:

- **keyboard hardware** - represents a device which is situated between the computer and the used keyboard. This device can have memory which stores the key strokes and can be accessed using a key combination
- **keyboard overlays** - they are used against ATMs, by putting them over the real keyboard
- **smarthphones sensors** - based on the accelerometer of a smarthphone can detect if multiple keystrokes are pressed together

For an attack to managed to succeed this type of attack, he needs root access to the computer on which he adds the keylogger. So he can insert a malware which will read the internal buffers of the keyboard or the buffers of the USB ports in case a keyboard is attached to the computer. Another way to succeed is to have access to the desktop so he can attach a keylogger between the keyboard and the desktop. The proposed solution is design to prevent only the hardware-based keylogging attacks.

2.4 Malware

These programs can be added on the user's computer and try to steal the information. These malwares have many purposes. Therefore the attacker can listen to the USB ports and monitor data transfer or he can listen to the network interface and see all the incoming or outgoing connections. These malwares can have serious consequences against the security. To improve the efficiency of the attack, the attacker needs root access to the computer or to trick the victim to run as an administrator the malware. If not, the attacker still can add the malwares but they will not be so efficient. According to Barbara there are different types of malwares [9].

- **USB Hacksaw** - was designed to be used by any configurable flash drive that can be customized with a compact disk, read-only memory partition. This malware will copy the information from external disk on a remote location.
- **USB Switchblade** - the purpose of this malware is to get information about the Windows system or the network in which it relies. The malware can contain different tools like: Internet Protocol Configuration Utility, Dynamic Host Configuration Protocol statistics, GNU, Wget, Product key Recovery, Messenger Password recovery, Network password recovery, Mail password viewer, Firefox password recovery. Each of these tools requires administrator level.

- **USB Device Overflow** - a buffer overflow represents a buffer in which was written to much data or after the end of the buffer. These attacks are used more against the operating system application.
- **USB-Based Virus Malicious Code Launch** - there are multiple types of malicious programs:
  - virus - is a program that propagates by contaminating other files or programs on a single host
  - worms - is a type of malware that has the ability to propagate itself without user interaction
  - trojans - deploy different viruses and worms
  - rootkits - main activities are backdoor admitance

For this attack to succeed, the attacker needs a way to add the malware on the victim computer and to run it there. Using these malware the attacker can do a lot of damages by reading private information. This attack will not be prevented by the design.

2.5 Man in the middle

In this attack, the attacker positions himself between the two parties that are communicating. In this way the attack listens to the messages between the two parties or he can initiate the communication between them. The Figure 2.2 presents an example in which Mallory, the attacker, listens to the channel and the Figure 2.3 presents an example in which Mallory initiates the communication.

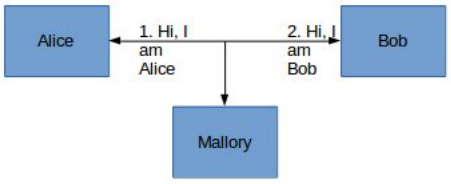


Figure 2.2: Mallory listen to connection

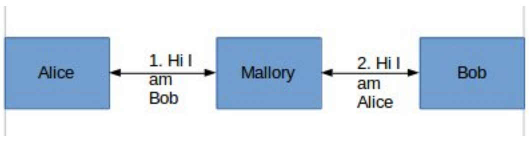


Figure 2.3: Mallory initiates communication

If the communication between them is encrypted then the attack can only read the encrypted information. For the attacker to succeed he needs to insert a malware on the victim's computer. This malware like in case of the Key Logging attacks it would read the internal buffers. In this way the attacker can read private information. In case the attacker reads the internals of the PC, then the design can't prevent this, but in case the attacker tries to read the Bluetooth connection, the design will prevent this.

2.6 Exhaustive search

This attack is also known as **brute force attack**. The attack tries all the possible solutions for a password, which it makes efficient in case the search space is small, but is inefficient in case there is a big search space. To improve the time the attacker can try using only words from a dictionary. The longer the password is the more time it takes to find it. The time growth exponentially with the password length. For example if the password has N bits then the proportional time is 2^N.

According to Landauer the energy required to perform a computation of bit flip equals to [18]:

$$E = k * T * ln2 \tag{2.1}$$

- k - represents the Boltzmann constant
- T - is the temperature of the computation device

To use the brute force attack against a password on 128 bits, it will require 2^{128} bits flips so the total energy required it would be 10^{18}. To improve the efficiency of the attack, the attacker can use multiple processors to find the password. For example devices which have multiple processors are GPU and FPGA. The GPU has hundreds of processing units and the FPGA consumes the same power as a computer but performs like 2.500 computers.

Another type of this attack is **reverse brute-force attack** in which the attacker tries multiple accounts with the same password. Therefore the attacker doesn't target specific user.

To manage to use this attack, the attacker needs access to the encrypted database. The success of this attack depends on the encryption algorithm. The proposed solution is design to defend against this type of attack because is using an encryption algorithm.

2.7 Social Engineering attacks

Social engineering attacks are the attacks in which the attacker tricks the victim to reveal private information. These attacks are against the humans and not against applications, therefore these attacks can't be prevented by any software. Different attacks are described by the Katharina Krombholz et al. [17]:

- physical approaches - they are also known as dumpster diving, because the attack presumes to find some notes or piece of paper with some private information in a dumpster. Another type of physical approach is the extortion.
- social approaches - the attacker tries to manipulate the victim to give him private information. The attacker can try to get friendly with the victim, so he can improve the chances of attacks.
- reverse social engineering - the attacker tries to make the victim to ask for attack's help and then he tries to get private information.
- social technical approaches - an example of this attack is known as baiting attacks. Where the attack leaves infected devices in public areas so the victim can pick it and insert it in the computer.

For this attack to succeed, the attacker can get friendly with the victim but this case can take a long time, a simple way is to send multiple email with some malwares to the victims and hopefully one of them will reveal some private information. The proposed solution can't defend against this because there is no technical defense, therefore the user has to be inform how this attack works and how to prevent it.

Chapter 3

Design

In this part we will present the design by arguing why we use or not use different features. First we will present an overview of the entire architecture and after that continue with specific decisions regarding the design. In the end we finish with the presentation of the final design.

3.1 Architecture

The previous semester report [22] and David Silver at al. [11] present multiple web based password managers from which most of them have problems. They leaked information and they are vulnerable to different attacks. Therefore the solution is to create a password manager which will store the encrypted passwords locally, so it would not require any internet connection to use them. This approach prevents any attack that can be used against web based password managers. This approach can be implemented in two different ways:

- Store the passwords on the computer.
- Store the passwords on a external device.

In both cases the passwords will be encrypted before they are stored in memory. The section Encryption 3.5 will present different encryption algorithms.

One scenario is to store the passwords on the computer. The advantage of this approach is that the password doesn't leave the computer. But if an attacker installs a malware then he may be able to read the information. The disadvantage of this approach is that the user can only use the password manager only on his computer. Therefore the user can't login from different computers.

In the second scenario, which will be used in the design, the passwords are stored encrypted on a mobile phone and the encrypted algorithm is implemented on a hardware device which is attached to the computer. To receive the passwords on the computer, the mobile application will send the encrypted passwords to the hardware device. The hardware device decrypts the password and after that it sends it to the computer. In this case there can be different approaches to display the information (username and website) on the mobile phone. We decided to display the information in clear text. The advantage of this approach is that is easy to use it, because the user can see the information he needs. The disadvantage of this case is that if attacker steals the phone, he can see user's information. The other approach is to display the encrypted information but then the user can't know which websites and usernames have been stored in the database. The advantage of this case is that it would be more secure than the other case. The final design will display the information in clear text because is easy to use, this improves the usability.

The purpose of the hardware device is to emulate a keyboard and to encrypt/decrypt the information. The main advantage of this approach is the usage of the device as a keyboard, in this case the password manager can be used on every computer. Another advantage of this approach is that we can prevent physical keyloggers attached to the keyboard.

Another factor to take in consideration is where the master password is stored. In the approach in which all the other passwords are stored on the computer then, also the master password needs to be stored on computer. By doing this, this approach will put together all the information. Another approach, which is used by the proposed solution, is to store the master password on an external device, so when the user wants to retrieve a password, he has to insert the device in the computer. This approach has the following advantages:

- if attacker steals the device, he will only steal the master password and he can not do anything without the encrypted passwords.

- if attacker steals the phone with the encrypted passwords, again he can't do anything with this information because the passwords are encrypted.

Therefore it seems a better choice to separate the master password from the passwords. One approach is to store the master key in clear text in EEPROM memory. The problem with this approach is that it would be easy for an attacker to read the memory and find it. Another approach is to pass the key through a key derivation function [4] and store that result. A key derivation function is a function which derives a password using a pseudo random generator. Cryptographic hash functions are popular examples of pseudo random functions used as key derivation functions. One example is MD5 [5]. The approach has the advantage that it would be harder for attacker to read the password because it would not be in clear text. One disadvantage of the MD5 functions is that, according to Honbo Yu this function is vulnerable to collisions attacks [23]. In our case is not a problem, because we use this function to derivate the master password and not to encrypt it. Another advantage is that the user will have strong passwords in memory, so the encryption algorithm will face better a brute force attack. For example if the user selects as a master password a word like 'table' then if the attacker steals the database with the encrypted passwords, he can try to do a brute force attack based on a dictionary. In this way the attacker will have bigger changes to decrypt the passwords. But if the password is passed through a key derivation function then the password stored in memory will be the MD5 value. In this case there are multiple key derivation functions like Aargon2 [8], MD5. The proposed solution uses MD5. Another approach is to use an algorithm to spread the password through the memory so the attacker can't find it at a fix address.

3.2 Phone-device communication

Section 3.1 mentioned that the proposed solution will have a phone and an external device. So this part will present different approaches on how to make the communication between the device and the phone. There are the following options:

- Bluetooth
- wireless
- cable

The advantage of first approach is that it can be secure, but this depends on the Bluetooth version [6]. If someone will read the connection it would not be a problem because the information is encrypted. Another advantage is that the attacker has to be close to the user to make the attack.

The second possibility is to use a wireless connection. The advantage of this approach is that the transfer speed is faster than the Bluetooth and it can be used at a longer distance. But this can be also a disadvantage because the attacker can attack from a bigger distance.

The last approach is using a cable. This approach can be the safest one, because no one from exterior can read it. The disadvantage of this approach is that the device needs two USB ports and also the phone application needs to read the data from the USB. This approach is hard to be implement and it would not be so comfortable for the user.

Therefore the solution will use the first approach in which the communication between the phone and the device is through Bluetooth.

3.3 Devices

The section Architecture mentioned that the solution uses a hardware device. Therefore there are the following options:

- Arduino
- FPGA
- Druino

All of them have different advantages and disadvantages. Our initial goal was to create a prototype, therefore we need more resources for our device. In this case all these devices can be used, all of them have enough computation power and size to support our implementation. Another criteria to choose the device is the knowledge required to program it. In this case we have knowledge regarding the Arduino and FPGA, we didn't have knowledge regarding Druino board. Also we took in consideration how easy is to implement the required application. In this case the Arduino Micro is the easier one. The advantage of Arduino Micro over the Arduino Uno is that it can emulate a keyboard, which is one of the key features of this design, therefore we use an Arduino Micro board.

3.4 Add, remove passwords

In case of adding or removing the passwords from password manager, there are the following approaches:

- create an application on computer to add, remove passwords
- extend phone application to add, remove passwords

The first approach presumes to create a GUI application on the computer and the user to use this one to add and delete passwords. Therefore to add a password the user has to send the password in clear text over the USB port to the hardware device which will encrypt it. From the device, the data is sent to the phone to store it in the database. To remove a password the message has to follow the same path only that the password will not be sent. The advantage of this approach over the second one is that is easy to use but it has also disadvantages: it requires another application to managed to add, remove passwords. From point of view of portability this approach can have problems with this but if the application is implemented in C++ language with the Qt framework, then this is not be a problem. Even if this approach has some disadvantages, this application will not be used so often, because we assume that the user wants more to retrieve passwords than to add or remove them.

To improve usability, the application will contain also a button to retrieve the entries from the database. Figure 3.1 shows the flow of messages from computer to the phone and back.

For example if the user wants to retrieve a password from the database, the information takes the following steps:

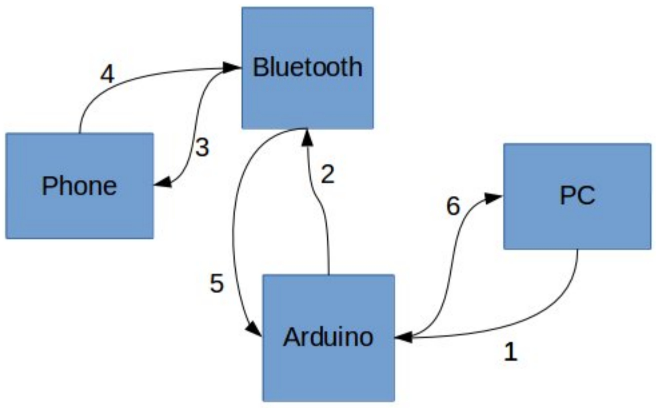


Figure 3.1: Message Path

- edge 1 contains the username and website, both of them are in clear text
- edge 2 contains the username, website in clear text
- edge 3 contains the username, website in clear text
- edge 4 contains the encrypted password
- edge 5 contains the encrypted password
- edge 6 contains the clear text password

In case the user tries to add a new entry in the database then the information will follow the following steps:

- edge 1 contains the username and website and password, all of them are in clear text
- edge 2 contains the username, website in clear text and the encrypted password
- edge 3 contains the username, website in clear text and the encrypted password
- edge 4 contains the reply is one of the following: OK or Failed.
- edge 5 contains the reply
- edge 6 contains the reply

This approach requires the device to have a button, so only when the button is pressed the operation takes place. The purpose of the button is to stop different attacks. One possible attack can be a malware to request all the entries. In case the hardware device doesn't have a button, then the attacker will managed to get all the passwords. But in case the device has a button then the attacker has to press the button to retrieve the entry. This is the same for the other operations: add, delete.

An extra feature of the GUI application is a random password generator. Therefore the user allows the system to generate the passwords for him. The user can configure the length of the password and which characters (letters, numbers, symbols) to be used. This feature is added because is a useful one, even though it can have flaws if the password generator is not a truly random number generator. This password generator can be placed in different parts of the password manager. There are three choices, in the GUI application, on the Arduino board or on the mobile application. The problem with the first choice and the last one is that they can't generate truly random numbers. Therefore the Arduino board will generate the passwords and the interface for the generator is on the GUI application. This approach will prevent also software keylogging attacks when the user adds a new password, because the user doesn't insert any password.

The other approach, regarding the add, remove passwords, presumes to extend the mobile application. The main advantage over the other approach is that the user doesn't need another application. The main disadvantage of this approach is that the user has to copy manually information from the computer to the mobile application, which in some cases can't be so easy.

In both cases, the password is sent in clear text over a channel. Therefore the attacker can try to read it. In case there is another GUI application, a malware can read the USB buffers before the application sends the password or in case the mobile application is extended, there can be someone that reads the connection. If the user wants to be 100% sure that no one will read his password in clear text then, he can reinstall the OS, so he make sure that the computer is clean.

Considering the advantages and disadvantages of each option the proposed solution uses a separate application to add, remove and retrieve passwords.

3.5 Encryption

Using an encryption algorithm, it will prevent attacks like men in the middle. If someone tries to listen the Bluetooth connection when the user sends the password to the PC, the attacker will read only the encrypted password. With this information the attacker can't do anything without the master password. The attacker will be in same situation if he tries to read the database from the phone, he will read only the encrypted passwords.

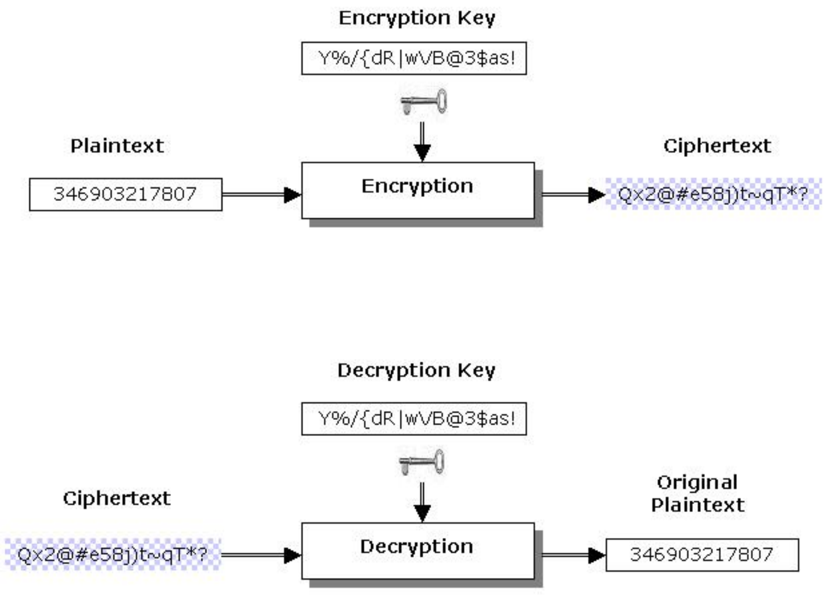


Figure 3.2: Encryption / Decryption Asymmetric [14]

In this case there are multiple encryption algorithms:

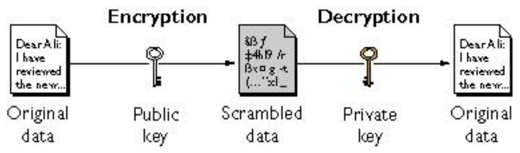
3.5.1 RSA

The Figure 3.2 shows how RSA algorithm is working. This encryption algorithm is based on asymmetric key, there is one key for encryption and one for decryption. The problem with this approach is that is not fast and easy to implement on a small device. The advantage of this algorithm is that the private key which is used for decryption will not be public. A good approach can be that when the connection is establish between the hardware device and the computer the device, the device will publish his public key. So when the user tries to add a new password he will encrypt it with public key. When the user tries to get the password he will send the encrypted password from the phone to the hardware device which then with the private key will decrypt the information. If this approach is used then also the GUI application has to be extended to encrypt the information.

3.5.2 AES

The Figure 3.3 shows how AES algorithm is working. This encryption algorithm is based on a single key. Based on the key there are 3 versions of this algorithm, when

Figure 3.3: Encryption / Decryption Symmetric [10]



the key is 128, 196, 256 bit. The advantage of this algorithm is that it is fast, easy to implement and doesn't require lot of memory. The fastest version is when the key is 128 bits because it would take less iterations to encrypt/decrypt the information than the longer keys. Another advantage of the algorithm is that it doesn't require lot of resources to be implemented, therefore it is a perfect fit to be implemented on a small device. One disadvantage of this is that when the key is on 128 bits is not the most secure one.

3.5.3 Hash functions

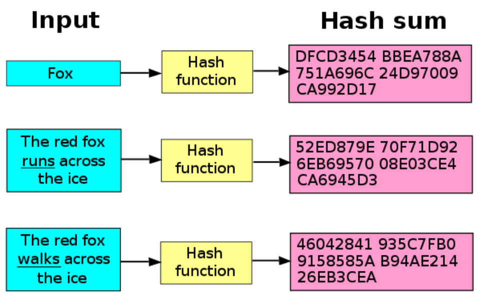

Input
Hash sum
Fox
Hash function
DFCD3454 BBEA788A 751A696C 24D97009 CA992D17
The red fox runs across the ice
Hash function
52ED879E 70F71D92 6EB69570 08E03CE4 CA6945D3
The red fox walks across the ice
Hash function
46042841 935C7FB0 9158585A B94AE214 26EB3CEA

Figure 3.4: Hash Functions [19]

The Figure 3.4 shows how a hash function is working. A hash function is a function which can be used to map data of arbitrary size to a fix sized data. An example of a hash function is MD5. The problem with this approach is that these hash functions can give collisions therefore for two passwords there can be the same hashed password. The main purpose of the MD5 is to verify data integrity, therefore it is preferred not to be used in encryption. Also another problem is that if the hash generates the hash of a password it would be very hard to find the password in clear text from the hash password.

Based on the fact that this algorithm should be used on a small device, then the encryption algorithm will be AES on 128 bits.

To improve the security of this password manager, the proposed solution will have a new defense layer. Given that the solution has two devices (phone and hardware device), we decided to add a feature to see if the devices are not used in pair. This helps to detect if someone tried to get the passwords from the phone or tried to use the hardware device with another phone. To do that the password manager has to store an identifier on both devices and compare them when the application starts. Therefore every time when the phone connects to the hardware device it tries to compare the identifiers. These identifiers can be counters, dates, hash values, etc. Also the GUI application could store an identifier on the computer, but then the problem would be that the GUI application is not used in all operations, like the phone application. Therefore it is better to store the hash values on the phone and on the hardware device. In our case every time when the phone connects to Bluetooth, it is asking for the stored value and compare it with the value which is stored in phone. Before the connection between the Bluetooth and the phone is closed, the phone calculates the new hash value and sends it to hardware device to store it in memory. In this way the user can detect if any of the devices are used with another device. The hash value has to change every time when the devices are used, for example when he adds, removes or retrieves a password.

3.6 Attacks that can not be prevented

The previous section presented which attacks can be prevented by the proposed solution. This section presents which attacks can not be prevented by the design and the effects of these attacks.

3.6.1 Read Memory

The current design stores the master password on the hardware device. In this case the password must be in an non-volatile memory, because the user needs it after the device is turn off. Non-volatile memories are: EEPROM and Flash. By storing the memory in EEPROM an attacker can try to read it using a fotodiode or he can try to change it. The attack can have the following effects:

- by modifying the password, the attacker can make the device unusable. To fix this problem user has to write again the master password in the memory.
- by reading the memory, the attacker can have access to the master password. But the attacker can not do anything with this, he will need also the encrypted passwords.

In the future the password can be spread in the memory so it would not be at a specific address.

3.6.2 Social Engineering

This kind of attack can not be prevented by any technical approach. If the user is tricked by the attacker to share his password, then the design is vulnerable to this attack.

3.6.3 Side channel attacks

It is known that this attack managed to break multiple algorithms and one of them was AES [24]. This attack presumes to get information generated by software-hardware interaction and based on this information it tries to find passwords. This attack can read the power or sound or the time between operations.

To prevent this type of attack, is possible to use a secure cryptoprocessor [20]. The cryptoprocessor will take the same time for any two different operations and also it will add a physical protective layer so the attacker can not try to put any needles on the data-bus to read the information. The disadvantage of this approach is that it is costly and needs a special hardware for this. The Arduino board cannot be used for such purpose.

The following table presents which attacks are possible and which one are not.

Attack	Possibility
Side channel attacks	Yes
Read Memory	Yes
KeyLogging	Yes
Man in the middle	No
Malwares	Yes
Brute force attacks	No
Social Engineering	Yes

The side channel attacks and the read memory attacks are not prevented by the proposed solution but with the information that the attacker gets it from these attacks, the attacker can't do anything. Only the hardware keylogger attacks are prevented by emulating a keyboard, the software attacks are still possible. If the user uses the password generator to add the passwords then in this case the attack will not be possible. The malwares are difficult to prevent and the proposed solution can't defend against these attacks, the same is for the social engineering attacks. The user has to know how to prevent these attacks.

3.7 Final Design

In the previous sections we presented the proposed design and presented also other possibilities for the design. In this section we will present the final design which will be implemented. This will have the following features:

- use an Arduino Micro to store the master password and the encryption/decryption algorithm. The encryption algorithm is AES 128 bit. Also the hardware device contains a password generator, a Bluetooth module and the main feature of this is that will emulate a keyboard. To improve the security the

Arduino has a button, which needs to be press for each operation the user wants to complete.

- use a mobile phone to store the encrypted passwords. From the phone is possible to send encrypted passwords to the hardware device. The communication between the phone and the Arduino is done through the Bluetooth.
- to add, remove passwords - there is GUI application, this one contains also the interface for the password generator. The communication between the GUI and the phone is made through the hardware device.

The Figure 3.5 shows a top level diagram of the proposed system. This password manager can be used by any kind of users from simple users to high tech users. The user has the possibility to store passwords from different websites or different application accounts. But the user has to know that there are possible attacks against it and these attack are able to steel or destroy user's private information. The possible attacks are not so easy to realize and they require more resources than the ones that the design prevents. These attacks can be governmental attacks or made by special hack groups because they have the required resources and the correct knowledge to attack the proposed solution.

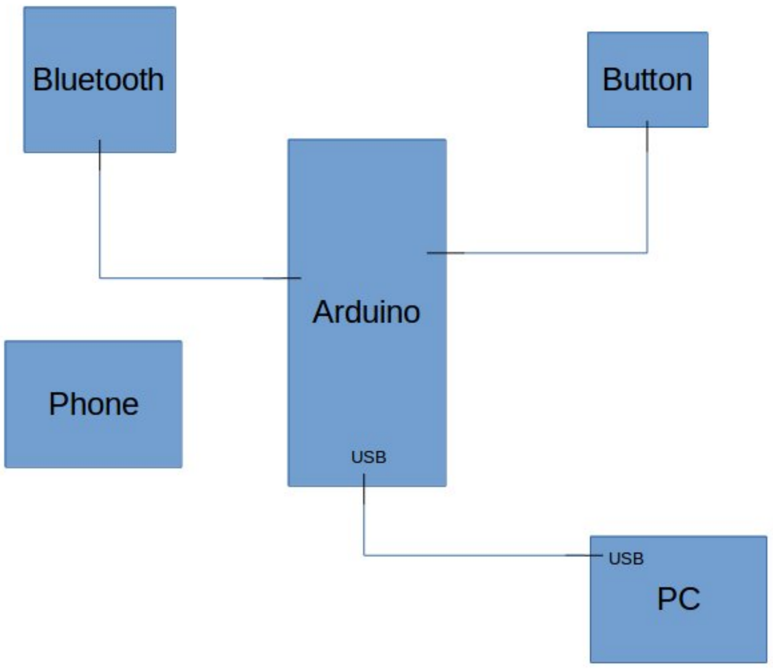


Figure 3.5: Design

Chapter 4

Implementation

The previous chapter presented the design of the proposed password manager and presented the purpose of each feature. This chapter presents the implementation of these features.

4.1 Architecture

Based on the devices needed for the project, the implementation can be divided in three different parts:

- Phone application- runs on a mobile phone
- Device application - runs on a hardware device, in our case Arduino Micro
- Computer application - runs on the computer that is connected to the hardware device

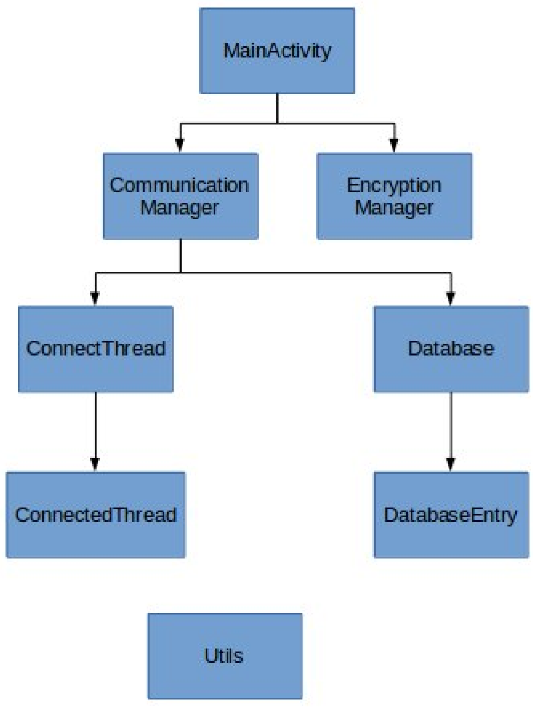
4.2 Phone application

This application can be implemented for multiple OS, for example Windows, iOS, Android. For this project the application is created on Android for the following reasons, we had already the device and also we had better knowledge in Android than Objective-C, Swift or Visual C++ which are needed for the other operating systems.

The application has two purposes:

- store the information in database - passwords are encrypted and username and website are in clear text
- send encrypted passwords to the hardware device

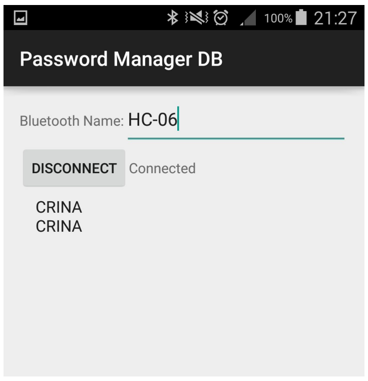
Figure 4.1: Android UML



The application was written in Android SDK 24.0.2 using Android Studio IDE. The Figure 4.1 shows the UML diagram, the class MainActivity represents the interface between the user and the back-end of the application. The class has the following GUI elements: an edit text to enter the name of the Bluetooth device, a button to establish or close the connection and a label which shows the connection's status. The label can have one of the two values: Connected or Disconnected. Also the MainActivity class contains a list with all the websites and usernames from the database. If the user wants to send a password he has to click on one of the rows. All these can be seen in the Figure 4.2. To use this application, first the user has to connect to the Bluetooth device, by entering the name of the device and press the connect button. When he can see that the connection status is Connected then he can send encrypted passwords to the hardware device.

The purpose of the class CommunicationManager is to establish and maintain the connection between the phone and the database and between the phone and the Bluetooth device. This class stores a Database and a ConnectedThread instance. The purpose of class Database is to add and remove data from database. The purpose of the class ConnectedThread is to connect to the Bluetooth device, to read and write information. Each message sent between the phone and the Bluetooth device has a type. The type of the message is represented by the first byte of

Figure 4.2: Android Interface



Bluetooth Name: HC-06

Connected

CRINA
CRINA

the message. The section Computer application 4.4 presents the types of these messages. Based on this type the CommunicationManager knows which actions has to take.

In the application, there is also an EncryptionManager, the purpose of this class is to calculate the new hash value, to read and write the hash value in memory. Every time when the application is used, a new value is added to the local variable. When the connection between the phone and the hardware device is closed, this class calculates the new hash value by applying the MD5 on the local variable and the class CommunicationManager sends the value to the hardware device. Also when the connection is closed, it stores the hash value in the memory so it can be reused when the application is reopen. The hash value is read from memory when the user tries to establish the connection between the phone and the hardware device. This application behaves like a database because it stores the encrypted passwords and let the external user to request or add passwords.

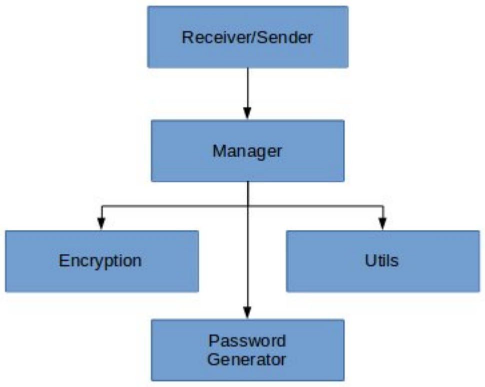
An attacker can not read the passwords from the phone because they are encrypted and the database is private to each application. If the attacker tries to emulate this application and ask the hardware device for different information, the device can't offer any information. Another possibility is to read the internal buffers of the Bluetooth, but also in this case, the information is encrypted. Therefore this application can be consider safe because it can face different attacks.

4.3 Device application

The device application has the following purposes:

- encrypt and decrypt password using AES encryption algorithm
- emulate a keyboard
- generate random passwords

Figure 4.3: Arduino UML



The Figure 4.3 presents the UML diagram of the application. The application was developed in Arduino 1.6.7 using Arduino IDE. The main component of this application is the Manager, this processes all the commands received from computer or from Bluetooth. The Sender/Receiver tries to read the commands from Bluetooth and computer. To read the data from the Bluetooth it uses an external library named SoftwareSerial. The purpose of the Encryption part is to encrypt or to decrypt the passwords. The component Utils is used in the manager, to extract data from messages. For example to get the username or the website or the password. Each message finishes with the char '\n'. Therefore this could be a problem if the users tries to send an encrypted password to the phone, because also the encrypted password could contain this char. Therefore the application sends the encrypted password to the phone in hex. The advantage of this is that the problem is eliminated but the disadvantage of this approach is that the size of the message will be bigger. For example if the encrypted password contains the symbol '\n', then the application will send to the phone two bytes '0C' and if the encrypted password contains the byte with value 60 then it will send the bytes '3C' which is the value in hex of 60.

The UML diagram contains also a password generator. This class is used to generate random passwords. This class use an external library called Entropy [7] to generate the passwords. When the user wants to generate a password for an entry, the GUI application sends the length of the password and which types of characters the password should contain. This class will generate random passwords until all the requirements are meet. So if a user tries to generate a short password with all the symbols, then he must wait more, until the requirements are meet. This has the following pseudo-code:

```
int length = getLength(message)
byte types = getAllowedTypes(message)
```

```
char password[16]
do {
        password = generatePassword(types, length)
} while( !isValid(password, types) )
```

So with this algorithm is possible for the following scenario: user requests to generate a password with symbols, numbers and letters and to have length 6. In this case the function generatePassword() will try to generate a password which contains these elements, but is possible to generate a password like "123abc" (without symbols). In this case the function isValid() will check this scenario. So it would try to generate a new password. Another case is in which the user wants to generate a password with only letters. In this case all the time the function isValid will generate true. Therefore it is advised to choose two or three types of characters (so the password will be stronger) and longer passwords (so the probability of the function isValid() to return true to be higher). One disadvantage of this library is that is slow to generate numbers. But the library "Entropy" mention that they generate truly random numbers which is a big advantage over the other parts of the application. A way of doing is to generate a random number of each type of characters in the password and after that to shuffle randomly the characters in the password. The problem with this approach is that to generate a random number it takes a lot of time, therefore this solution is ignored.

Before it can be used first time, the device has to be configured. It needs to be set the master password and the tables required for the encryption. All this information will be stored in the EEPROM memory. To managed to write the master password and the required tables the user has to use another program. In this program the user is asked to type the master password. After the password was inserted, this is added in the key derivation function which is a MD5 function. The result of this function is a string on 16 bytes, which will be stored in the EEPROM memory.

The Attacks chapter presented different attacks against hardware devices and the steps to be take by the attacker. The chapter Design mention exactly which attacks can be used against the proposed solution and what is their effect.

4.4 Computer application

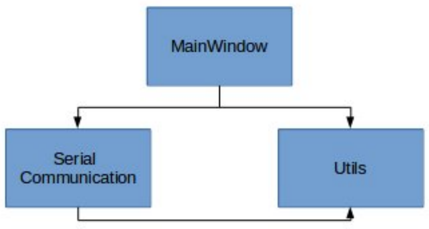
The purpose of this application is to add, retrieve or remove entries from the database. To managed these operations the device has to be connected so it can encrypt or decrypt the message and send it to phone application. The connection between the device and the computer is seen as a serial port. The user has to configure the serial port. He has to configure the following parameters:

- port name

- baud rate
- data bits
- parity
- stop bits
- flow control

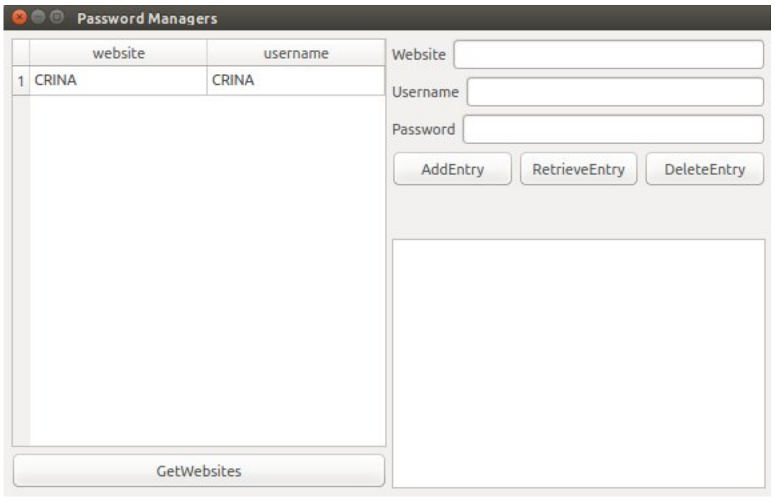
The user has to configure these parameters in the tab: "Configuration" and after that to open the port by pressing the button: "Open Port". These fields are read every time the user tries to open a port. To see on which port the device is connected, the user can use the command: "dmesg | grep ACM". The command is valid on Linux. After the user presses the button to open the port, there will appear a message box which says if the port was open successfully or not.

Figure 4.4: GUI UML



This application can be implemented in one of the languages: Java, C++, Python. We decided to use C++11 and Qt using QtCreator for IDE, because of the previous knowledge in this language and Qt has good support for serial communication. The Figure 4.4 presents the UML diagram of this application. The interface between the user and the backend is through the class MainWindow, this represents the window which have different buttons and edit lines so the user can add, get or remove passwords. Also the GUI has a list to display all the passwords from the database. To see the entries from the database first the user has press the button *GetWebsites*. Other GUI elements are the buttons and the check box used to add a new password in the database but the password has to be generated by the hardware device. The previous section mention that only the hardware device can generate a reliable random password generator. To add a password with a generated password, the user needs to select at least one check box and press the button: AddEntry and GeneratePassword. There are some restriction regarding the passwords generated, the passwords generated can be only between 6 and 12. On the main window there is also a log edit, this stores a log of the actions that were taken with this application, from when it was started. All these can be seen in the Figure 4.5.

Figure 4.5: Gui Interface



The communication between the USB port and the application is made by the class SerialCommunication. This class reads the configuration file. Using the Qt framework the data is read in another thread and the function readBytes() receives it. Also this function tries to decode the information based on the first byte of the message. There are the following message types:

- AddEntry = 1
- RetriveEntry = 2
- DeleteEntry = 3
- ObtainWebsites = 4
- AddGeneratedPassword = 5

Based on these values each message is processed in a different way. Each message has to terminate with the value '\n', if the application can't read these, it means that it got a wrong message and it will not process it. If the message is composed from multiple items then they are separated by '\r'. This character is used as a separator between the data in the information because is not a printable character. For example if the user tries to delete an entry then the command will have the following format: '3\rwebsite\rname\n'.

The file Utils.h contains only the declaration of different functions used to parse the messages or to generate request messages.

4.5 Tests

This section presents possible methods to be used to test the application. There are different types of tests:

- white box
- black box

The white box tests represent tests in which the tester can see the source code and based on that and on the requirements he can write the tests. Black box tests represents the tests in which the tester can see only the interface.

For example some white box tests can be used to test the core of the application, in our case to check the mobile application if it parses correctly the message from the computer, or on the computer to check if it parses correctly the message from phone and on device if the encryption or decryption work. Usually these tests are run every day or when there are changes in the code. Black box tests are the tests which checks if the phone managed to connect to the hardware device, or the computer application managed to open the USB port. Usually all these tests are run automatically.

In the current implementation there are no explicit white box tests, the application is tested manually. Another reason for not adding the white box tests is that the source code is stored in repository. This helped us to see all the previous changes and it is easy to find the bugs. If the application will growth then it is necessary to add some white tests and to run black tests automatically. Also we should add some tests to check the vulnerability of the password manager. In chapter Attacks, we saw general attacks that can be also used against the proposed solution. For example we could create a script to try to decode the database using the brute force, or another example is to create a malware which reads the internal buffers. These vulnerability tests can be used in future when the application is develop to stand against these attacks.

The project contains only black box tests that are run every time when there is added new functionality or when something is changed. The project doesn't contain white box tests because the effort to add them was to big because there has to be different tests frameworks, one for phone and one for computer.

4.6 How to use it

This section presents the required steps to start to use the password manager and what the user should do in each of them. In case the user needs to install the phone application he just has to install the apk file. In case that the user install the computer application he has to the install the application by copying the exe file on the computer. This file also requires the Qt framework or will copy the exe which is compiled statically with Qt. Currently the exe file works only for Linux OS. To use the hardware device he has to do the following:

- first run the program SetKey which sets the md5 of the master key and the tables for the encryption.
- run the program PortTest which sets the main application.
- connect the phone application to the hardware device

After following these steps the user can start to use the application.

Chapter 5

Future work

This chapter presents some extra features that can be added to the project.

5.1 Phone application

The phone application can be ported on multiple OS. For example it can be written for iOS and for Windows phones. In this way there can be multiple users. This will not be so hard because this application is straight forward.

5.2 GUI application

Like in the phone application, this application can be ported on different OS, like Windows and iOS. It would be straight forward to port the application because is written in C++ with Qt framework which are supported by all major OS.

Another feature that can be added to the password manager, is to have the possibility to share all the passwords with another user. To do that, the GUI application needs to be extended. The password manager is not web-based so the user needs to plug in the both devices in the PC, because on the new phone he has to store the encrypted passwords with the new master password. Another possible feature is to replicate the encrypted password manager on a different phone, but this feature can offer opportunities for the attacker to steal the encrypted passwords. The attacker can take the victim's phone and replicate the victim's database on his phone. This feature needs more investigation to see if is useful or not.

5.3 Arduino board

The current implementation uses an Arduino Micro board with a breadboard, in the future this can be changed with a smaller board, which has only enough power and resources to run the application. The new hardware will need also to incorporate the Bluetooth module and the button. The device should be also smaller than the current one, because it would be more easy to use.

Currently when the master password is rewritten, only the password is written everything else is set to '0' except the tables need it for encryption. In this case it is easy for the attacker to find at which address the password is stored. One possible solution for this problem is that every time when the the master password is reset, the full memory is written with garbage data and after that the required data is written. In this way, it would be harder for attacker to known at which address the password is stored. Another possibility is for the password to be spread around the memory so it would not be at a fix address.

5.4 Prevent different attacks

This part proposed different approaches to prevent some attacks that can't be defeated with the current implementation.

5.4.1 Side channel attacks

The Attack chapter presents how these attacks work. Therefore to prevent these kind of attacks the hardware should be extended with a cryptoprocessor, in this way it can detect if someone tries to tamper the device. If someone will try to read the memory, then it can erase the memory, this process is called zeroization. Also by using a cryptoprocessor, it can prevent the timing attacks in which the attacker tries to measure the time needed for different operations.

5.4.2 Read memory

Also in this case, by using a cryptoprocessor these attacks can be prevented. If someone will try to tamper the exterior layer, then it will clear the EEPROM memory, therefore the attacker can't read the memory.

5.4.3 Malwares

In case the computer is infected with a malware, the attacker can read the password in clear text. There are some basic defends in this case, like reinstall the OS on the computer then the user is sure that the computer is not infected. There is another approach, to change the encryption algorithm from AES to RSA. So when the hardware device is connected to the computer or when the phone connects to the device it would publish his public key. Now when the user will try to add a new entry, first the GUI application will encrypt the password and send it to the hardware device which will send it unmodified to the phone. When the user will try to receive the password, he will send the encrypted password from the phone to the hardware device which will decrypt it and will emulate the keyboard. By using this approach the password manager prevents someone to read the password in clear text when the passwords are added in the database. The problem with this approach is that the hardware device needs more power to compute the private and public key.

Chapter 6

Conclusion

In these days, it is known that authentication based on passwords is insecure. Already different websites like LinkedIn and eHarmony were attacked and private information was stolen. In the year 2014 a list of Gmail accounts was made public. One of the problems is that the user tries to select passwords which are easy to remember. Another problem is that, if he has a strong password, then he is using it for all the applications and websites where he has an account. These approaches can increase the chances of successful attacks.

To resolve these problems there are password managers. A password managers is a software or a device which stores all the passwords. By using this password manager, the user needs to remember only one password, which is used to access all the other ones. There are multiple password managers with different features like: encryption, two-factor authentication, generate passwords, etc. The are some web based password managers like: LastPass, FinalKey, Dashlane and some of them can store the passwords locally.

These password managers prevent different attacks like, SQL injection, Key logging, Man in the middle but they are still vulnerable. David Silver at al. and Karthikeyan Bhargavan and Antoine Delignat-Lavaud presented some attacks against existing password managers [11] [16]. Another problem is that one of the famous password managers LastPass leaked some private information [13]. Therefore this paper proposed a solution which is not web based. The approach is to store all the passwords encrypted with AES on 128 bits on the phone and have a device which is used to emulate a keyboard. In this way the user can use the application on every computer. This approach will prevent attacks which are based on the internet. The down side is that the design makes possible other attacks like: Read Memory, Side channel attacks.

Even if there are other possible attacks, this approach has different defenses against them. For example, it emulates a keyboard, the passwords sent between the hardware device and the phone are encrypted, the master password is separated from the encrypted passwords. One of the requirements is to be easy to use therefore it contains a GUI application to add other entries in the database. The username and the website are not encrypted on the phone, so the user can select easily which password to send to the PC. Being easy to use let any kind of users to use it, from IT professional to simple computer users. Therefore this password manager can be used by everyone but they have to know that there are possible attacks against the it. The chapter Attacks presents some attacks that can't be defended by this design but they are not easy to make and some of them were taken in consideration for the future work. Some of the attacks required lots of resources, for example attacks which read directly the memory. Therefore these attacks can be done by hacker groups or governmental attacker, which have all the required resources and knowledge. Therefore the user needs to take in consideration that the encrypted passwords are not 100% safe. The chapter Future Work presented some features that can be added to improve the usability and security of the password manager.

In the end we can say that we managed to create a prototype, that can be used in every day and has a high security by having multiple layers and is easy to use.

Bibliography

[1] URL: http://wpengine.com/unmasked/.

[2] URL: http://lifehacker.com/5-million-gmail-passwords-leaked-check-yours-now-1632983265.

[3] URL: https://en.wikipedia.org/wiki/Keystroke_logging.

[4] URL: https://en.wikipedia.org/wiki/Key_derivation_function.

[5] URL: https://en.wikipedia.org/wiki/MD5.

[6] URL: http://electronics.howstuffworks.com/bluetooth4.htm.

[7] URL: https://sites.google.com/site/astudyofentropy/home.

[8] Daniel Dinu Alex Biryukov and Dmitry Khovratovich. *Argon2: the memory-hard function for password hashing and other applications*. 2015.

[9] Brian Anderson Barbara Anderson. *Seven Deadliest USB Attacks*. Syngress.

[10] Peter Bright. *Microsoft .Net*. URL: http://archive.arstechnica.com/paedia/n/net/signing.html.

[11] Eric Chen Collin Jackson Dan Boneh David Silver Suman Jana. *Password Managers: Attacks and Defenses*. 2014.

[12] Ping Wang Ding Wang. *The Emperor's New Password Creation Policies*.

[13] Tom Gijselinck. *LastPass: password manage*. 2014.

[14] Srdjan Holovac. *Securing Data at Rest:Database Encryption Solution using Empress RDBMS*. URL: http://www.empress.com/marketing/encryption/EmpressSecurityWhitePaper.v11.htm.

[15] Jfkfhhfj. *Another Type of Transistor: The Mosfet*. 2010. URL: http://mcuplace.com/mcu/blog5.php/another-type-of-transistor-the-mosfet.

[16] Antoine Delignat-Lavaud Karthikeyan Bhargavan. *Web-based Attacks on Host-Proof Encrypted Storage*. 2012.

[17] Markus Huber Edgar Weippl Katharina Krombholz Heidelinde Hobel. *Advance Social Engineering Attacks*. SBA Research.

[18] V.Vitell M.B. Plenio. *The physics of forgetting, Landauer's erasure principle and information theory*. Contemporary Physics, 2001.

[19] pradhikshaa. *Hash Functions*.

[20] Ingrid Verbauwhede Santosh Ghosh. *BLAKE-512 Based 128-bit CCA2 Secure Timing Attack Resistant McEliece Cryptoprocessor*. 2014.

[21] XiaoFeng Wang Kehuan Zhang Shuo Chen Rui Wang. IEEE Computer Society.

[22] Horatiu Vultur. *Password Managers*. 2015.

[23] Xiaoyun Wang and Hongbo Yu. *How to Break MD5 and Other Hash Functions*. Shandong University, Jinan 250100, China.

[24] DengGuo Feng YongBin Zhou. *Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing*. State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Science Beijing.

Appendix A

Preliminaries

In this paper we will use the following terms:

- AES - Advanced Encryption Standard, is a symmetric block cipher.
- Clear text - represents the information which is not encrypted.
- Decryption - is the process of decoding the information.
- EEPROM - is the acronym for Electrically Erasable Programmable Read-Only Memory and is a non-volatile memory used in computers.
- Encrypted text - represents the information which is encrypted.
- Encryption - is the process of encoding a message in such a way that only authorized parties can read it.
- FPGA - Field Programmable Gate Array.
- GUI - Graphical User Interface.
- IDE - Integration Development Environment.
- Key - represents a word which is used to transform the plain text in encrypted information.
- Malware - represents a malicious software.
- OS - Operating System.
- RSA - is a public key crypto-system. Stands for Ron Rivest Adi Shamir and Leonard Adleman.