Embedded Zero-Knowledge Authentication

Project submitted by:

Yosi Schlakman - 213661002

Yehoshua Gruenspecht – 332521103

Graphical user interface, website

Description automatically generated

Computer Science, JCT, Fall, 2021

Instructor: Prof. Arie Henel

# Table of Contents

Contents

[Table of Contents 2](#_Toc92411739)

[Abstract 3](#_Toc92411740)

[Introduction 4](#_Toc92411741)

[Authentication 4](#_Toc92411742)

[The Problem With Simple Authentication 4](#_Toc92411743)

[Zero-Knowledge Authentication 5](#_Toc92411744)

[Feige-Fiat-Shamir Identification Scheme 6](#_Toc92411745)

[Implementation 8](#_Toc92411746)

[Overall 8](#_Toc92411747)

[Security 8](#_Toc92411748)

[Future Improvements 9](#_Toc92411749)

[Conclusion 10](#_Toc92411750)

[Acknowledgments 10](#_Toc92411751)

[Appendix 11](#_Toc92411752)

# Abstract

As we all know, authentication is a key component in software security. The most common way to authenticate a person or machine is via password that only the one being authenticated should know. However, if there were to be an eavesdropper, this password can easily be heard and anyone can now be falsely authenticated. We wish to solve this problem by authenticating without revealing the password. This is called “Zero-knowledge authentication”. In the following project we researched the topic of Zero-knowledge authentication and created an embedded system that implements the Feige-Fiat-Shamir method for Zero-knowledge authentication. We will prove that our system works as an authentication system and that it fulfils the criteria for being Zero-knowledge.

# Introduction

## Authentication

In software security, authentication is when we ask the question: “Are you who you say you are?” to be sure we are communicating with the right person or machine. Peggy can authenticate herself in a number of ways.

1. Something Peggy knows

This can for example be a password that only Peggy knows therefore if Peggy tells Victor the password, Victor will know for sure that he’s talking to Peggy. Of course it can’t be this simple as we will see later.

1. Something Peggy has such as a physical key or magnetic chip.
2. Something Peggy is. For example, biometrics, fingerprint, face ID and even handwriting.

Authentication can be man-to-machine or machine-to-machine. In our project we will be dealing with machine-to-machine authentication. We will connect a Tiva Launchpad to a PC via serial and run an authentication protocol where the Tiva Launchpad proves it is part of a group that can communicate with the PC by knowing something only a member of the group will know.

## The Problem With Simple Authentication

As we already stated, Peggy can authenticate herself by proving she knows something only a member of the group with access would know. The simplest way to do so would be to simply tell said secret to Victor and Victor will know if it is correct or not.

This method is problematic, because if Eve were to eavesdrop while Peggy revealed her secret to Victor, or if Eve got a glimpse victors list of secrets (got access to the database) - Eve now knows Peggy’s secret and will be able to convince Victor that she is Peggy and therefore has all of Peggy’s capability’s that Eve may not have otherwise.[[1]](#footnote-1)

One possible solution is to use a One Time Password. In the case of an OTP eavesdropping is useless because the password will change next time it needs to be used. The problem with OTP is it isn’t easy and cheap for the client. There are other solutions such as the many challenge-response authentication protocols however, in these protocols, the password stays the same every time and can be deciphered and therefore still vulnerable. Another disadvantage is that all the encrypting and decrypting takes time.

## Zero-Knowledge Authentication

In the following project, we will use Zero-knowledge proof to solve the problem with simple authentication stated above.

Zero-knowledge authentication is where Peggy authenticates herself by proving to Victor that she knows a secret only she should know without revealing the secret to Victor. This can be done by Victor asking Peggy a series of questions, that by answering correctly consistently, prove that Peggy knows the secret. For example, the Two Balls and Color Blind Friend: Imagine your friend is red-green [colour-blind](https://en.wikipedia.org/wiki/Color_blindness) (while you are not) and you have two balls: one red and one green, but otherwise identical. To your friend they seem completely identical and he is sceptical that they are actually distinguishable. You want to prove to him they are in fact differently-coloured, but nothing else; in particular, you do not want to reveal which one is the red and which is the green ball.

Here is the proof system. You give the two balls to your friend and he puts them behind his back. Next, he takes one of the balls and brings it out from behind his back and displays it. He then places it behind his back again and then chooses to reveal just one of the two balls, picking one of the two at random with equal probability. He will ask you, "Did I switch the ball?" This whole procedure is then repeated as often as necessary.[[2]](#footnote-2)

A Zero-knowledge technique must satisfy three conditions:

1. Completeness: If this statement is true, the honest verifier will always be convinced of this fact by an honest claimant.
2. Soundness: If the statement provided by the claimant is false, no cheating claimant will be able to convince the honest verifier that it is actually true.
3. Zero-knowledge: If the statement is true, no cheating verifier can learn anything other than this property. This is proved by showing that every cheating verifier has some simulator that can give a fake transcript of interaction between the claimant and the verifier when it has to prove about the relationship.[[3]](#footnote-3)

Let’s consider the Two Ball example.

Completeness: If the balls really are two different colors, the friend will be convinced when you tell him that he switched the balls (assuming you are honest).

Soundness: If someone who doesn’t actually know if the balls were switched or not would try to convince the color blind friend that he does know, he would have a 50% chance of guessing correctly. Now let’s assume that the process was repeated 10 times. That means that the cheater now has a chance of tricking the friend. Since the probability of tricking the verifier is so low, it’s safe to say the no cheater will be able to convince the verifier that he is an honest prover.

Zero-knowledge: The friend is still color blind and since you never tell him which ball is which color, only that they are different colors, he still won’t know.

## Feige-Fiat-Shamir Identification Scheme

For our project we decided to use the Feige-Fiat-Shamir identification scheme (FFS) to implement our Zero-knowledge authentication system. Here’s how it works:

1. Peggy choses two prime numbers p and q where p
2. N = p\*q
3. S1,S2,…,Sk are co-prime to N and are Peggy’s secrets.
4. V1,V2,…,Vk are sent to Victor while -> Vi = Si2 mod N.
5. Peggy chooses random r ∈ R and s ∈ {1,-1} and calculates x s\*r2 mod N
6. Peggy sends x to Victor.
7. Victor chooses a1a2…ak | ai ∈ {0,1} and sends to Peggy
8. Peggy calculates and sends y to Victor
9. Victor verifies that y2 x(V1a1V2a2…Vkak) mod N
10. Run multiple times until Victor is satisfied.[[4]](#footnote-4)

Now let’s check that FFS satisfies our 3 conditions for Zero-knowledge proof.

Completeness: If Peggy really knows the secrets, Victor will know after receiving both the x and the y values because

Only if meaning, Peggy knows every Si value. Therefore, Victor will be convinced by a true statement.

Soundness: Eve will be able to learn the Vi values by eavesdropping. However, this will not be enough to be authenticated. Eve will now still have to guess all of Victors ai values and then pick a random y to calculate  and then send x to Victor. Now when Victor sends a1a2…ak Eve will simply return y and because of the x Eve sent Victor:

Victor will be falsely satisfied!

However, FFS is still sound because the probability of Eve guessing all ai values correctly is and the probability of Eve guessing correctly for all t rounds of the protocol is . For example, if we choose k = 10 and t = 5 we will get

In other words, the probability of Eve identifying as Peggy is zero (extremely low).

Zero-knowledge: Even if Eve learns the Vi values, she still won’t be able to know the secrets because it’s extremely hard to determine a modular square root when the modulus' factorization is unknown (like the discrete log problem).

# Implementation

## Overall

To implement a Zero-knowledge authentication system we used an embedded device (Tiva Launchpad) as Peggy trying to authenticate itself to Victor our PC.

The order of events goes like this:

1. Run the program.
2. Connect the Tiva to the PC using the USB cable.
3. The PC will now initiate a two-way handshake with the Tiva.
4. Now that the initial connection has been made, we can run the FFS identification scheme to authenticate the Tiva while all of the communication is being done through serial.
5. Run the FFS identification process t times. We chose t = 5 because we decided it was enough to make the probability of an eavesdropper guessing extremely low but not to many times that the whole process feels long and tedious for the user.

We also created a GUI to show the user every step that is going on the software level and the progress being made.

Our program fulfils the criteria for Zero-knowledge proof for the same reasons that FFS does.

## Security

* We chose to use a secret vector of length k = 4, and to run the FFS identification scheme t = 5 times. The reason we chose these numbers are as follows:
  + We experimented with various t values and found that t = 5 is the “sweet spot” for the identification (in our program) to not take too long, but also not be so short that the user can’t see what’s going on.
  + Once we found our t value, we knew we wanted the probability of successful cheating to be over one in a million, so we chose k = 4 - giving a cheater a probability of for success, but not increasing the probability (and therefore the execution time) more than necessary.

Of course if our program was to be modified and used for applications that require (almost) instantaneous authentication – such as smart card authentication, the t and k values would have to be reconsidered for dealing with the time constraints and possibly lowering the probability of malicious success if the application allows it (since a probability of one in a million is not needed for many application – such as a simple key card access program).

* We chose to use a prime number of 2048-bit length for N in the FFS scheme. The reason for our choice is that we found (via online research) this was the recommended length for RSA primes, in order to prevent factorization of N using modern technology and near future technology. Additionally, we found that all the official “RSA numbers”[[5]](#footnote-5) of 2048-bit length have not yet been factorized, which gave us confidence in our choice.
* While our program isn’t vulnerable to Eve eavesdropping on the communications, it is vulnerable to man in the middle attacks in which Eve would not just be a listener, but rather she would be in the middle of Peggy and Victors communication.

In this way, Eve could relay all of Peggy’s communications to Victor and all of Victors communications to Peggy – causing Victor to verify Eve when he thinks he’s verifying Peggy. While this is of concern, this still doesn’t allow Eve to know Peggy’s secret, only to “prove” to Victor that she does. Additionally, this wouldn’t allow Eve to authenticate whenever she wants – she can only authenticate when Peggy is actively trying to connect.

* After the authentication, it is assumed that the device connected to the PC serial port is the Tiva. This allows for a theoretical attack in which right after the Tiva authenticates, an attacker connects to the serial port and essentially “hijacks” the Tiva’s session. We tested for this attack, and the serial connection used automatically closes when the Tiva is disconnected and doesn’t allow a newly connected device to use the same port. However, there probably is a way to exploit this (that we don’t know of) and therefore this is a vulnerability.

## Future Improvements

* As mentioned in the security section, our program is vulnerable to the man-in-the-middle attack. This can be improved upon by adding some symmetric key encryption scheme to the communication between the PC and the Tiva (such as RSA). If the scheme is too slow, it can be used to initially transfer shared keys for symmetric-key cryptography which is faster.
* also mentioned in the security section, is the “session hijacking” problem. This can be improved upon by implementing the previous point and implementing sessions (similar to the ones used in most browsers) – in which the Tiva gets a session key (over an encrypted line) that he gives the PC when he wants to communicate with it. This way the PC knows that it is really the Tiva that is connected and that the session hasn’t been hijacked.
* We chose to load an image from the Tiva to our program on the PC in order to show that the Tiva is now trusted. However, more elaborate and abundant functionality can be thought of and implemented. We chose not to focus too much on this area of the project, since the requirement was to make an authentication system – but it could make the program more practical and elevate its “cool” factor.

# Conclusion

* We described an embedded Zero-knowledge authentication system using a Tiva Launchpad and the Feige-Fiat-Shamir identification scheme.
* It’s more secure than regular passwords.
* It’s easier to use than OTP.
* It’s quicker and more secure than a Challenge-response technique or an encrypted password.
* Just like everything else in software security, there is still room for improvement.

# Acknowledgments

We would like to thank Prof. Arie Henel for his helpful remarks and for guiding us through this past semester and this project in particular.

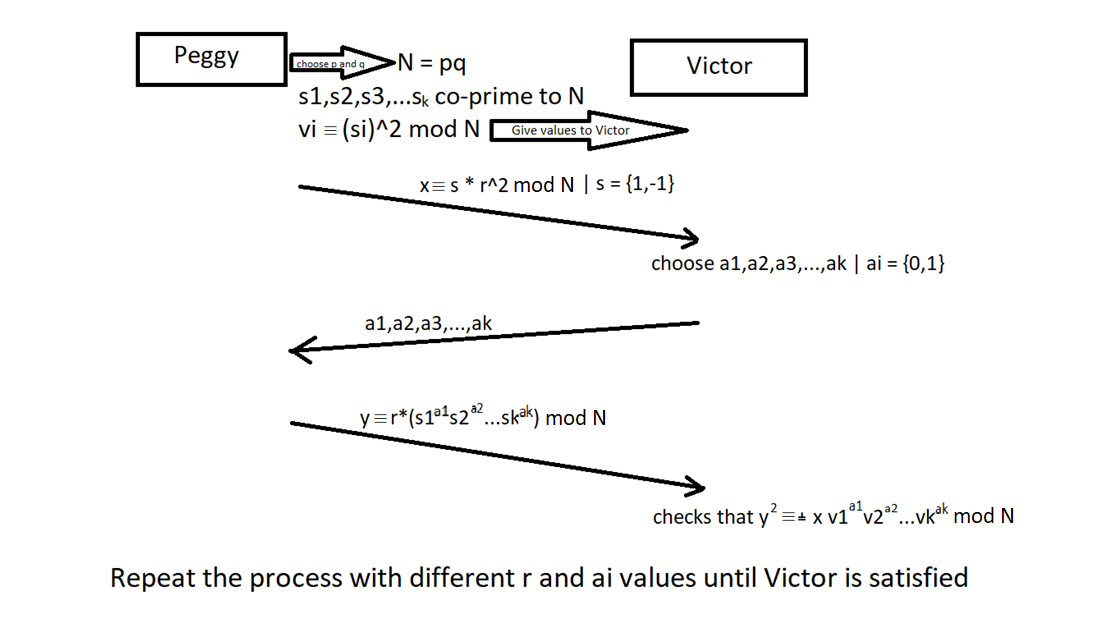
# Appendix

1.

Diagram, schematic

Description automatically generated

2.



1. See appendix 1 [↑](#footnote-ref-1)
2. Taken from Wikipedia [ZK\_Two\_Balls](https://en.wikipedia.org/wiki/Zero-knowledge_proof#:~:text=of%20her%20knowledge.-,Two%20balls%20and%20the%20colour%2Dblind%20friend,-%5Bedit%5D) [↑](#footnote-ref-2)
3. Taken from <http://ethesis.nitrkl.ac.in/5755/1/110CS0371-2.pdf> page 19 [↑](#footnote-ref-3)
4. See appendix 2 [↑](#footnote-ref-4)
5. https://en.wikipedia.org/wiki/RSA\_numbers [↑](#footnote-ref-5)