POLITECNICO DI TORINO

Department of Electronics and
Telecommunications

Master of Science Thesis

Authentication in Remote Controls

Academic Tutors:
Prof. Michele Elia
Prof. Carmelo Interlando

Candidate:
Andres Yesid Diaz Pinto

October 2014

to my family

Acknowledgement

I would like to thank all the people who have made significant contributions to my work:

- I would like to express my deep gratitude to *Professor Carmelo Interlando* and *Professor Michele Elia*, my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work. Also for giving me the great opportunity to develop this project in USA.
- Quisiera agradecer de una manera muy especial a mi padrino y tio Nahin; que gracias a su gran e incondicional apoyo he podido llegar a donde estoy. Estudiar en la Pontificia Universidad Javeriana y en el Politecnico di Torino ha sido gracias a su increible contribución. Sin su ayuda, nada de esto hubiese sido posible. Gracias a mi tio por hacer este sueño posible en el cual aprendí muchísimas cosas.
- Quisiera tambien agradecer enormemente a toda mi familia. En especial a mi madre, mi padre, mis hermanas y los niños. Quienes han sido el motor de mi pasión por seguir adelante y desarrollarme como persona. Quienes tambien han estado muy pendientes de mi y de las cosas que me sucedian.
- Quisiera agradecer a Andrea quien ha estado conmigo durante este tiempo apoyandome e impulsandome a trabajar y ser mejor cada día.
- My special thanks are extended to my friends who I met in Bogota, Turin and San Diego. Thanks for all you guys have done for me.

Contents

Introduction

Nowadays, it is more common for people use Internet to buy everything they need or they want. It is also true that every day the amount of people using credit or debit card is increasing in an extraordinary way. Every day systems come out that make easy payments or money transfers. Therefore, it is really important that information transmitted in different ways will be safe.An important factor to take into account is the devices used to communicate between two points. This means a transmitter and a receiver device. Who develops systems that make safe communication possible between two or more points, must consider important that all the devices that people use will be authentic. In other words, developers invest a lot of money finding devices, and in general, systems are difficult to clone.

Suppose a small company dedicated to develop crypto-systems that allow low-cost communication between two, or more, points or devices, and a person asks for a service from the company. This service must satisfy these conditions:

- The company must develop remote controls and its respective base stations to whatever application. In those remote controls, there must be an algorithm that makes defined, step-by-step procedures and safe communication with any base station fabricated for the company.
- The company must make a system difficult to replicate or clone, making it hard to clone the function of the remote controls and/or base station.
- And the company offers different possibilities to this problem depending on energy consumption, processing capacity, and remote control complexity.

The primary motivation of this project is to do the function of the company in the case previously described. Based on different possibilities offered by cryptography and anti-tamper technologies like a Smart Card, this system must make any attempt to clone or copy any part of the system too difficult. In this project several solutions to this problem are described, depending on important requirements like energy consumption, difficulty to clone, processing capacity, functionalities in the remote control, and cost.

Other problems this thesis addresses are system copy and feasibility of remote controls. System copy implies make the system immune to any possible attack or intend to replicate it. Feasibility implies that it is viable to implement. Some cryptography algorithms hold the promise of solution to these problems. Basically, cryptography is the science of secret writing. It provides a means to communicate in such a way that a third party is unable to reveal transmitted information between first and second party.

Cryptography indirectly provides authentication because only the first and second party know how to encrypt and decipher each other's messages. Another useful tool used is Anti-tamper technologies. That keeps and protects information from any attempt to modify or read it, by either the normal users of the system or system, or a third party.

Given the context, later this paper will describe authentication schemes in the application and use of remote controls, in particular, domestic use or applications where large amount of remote controls are needed. To do that, it will be described different cryptography primitives applied in Smart Cards, different types of authentication and Remote control's functionalities. Based on a couple cryptography primitives was developed a particular signature system, different to the classic one. That system was based on clock controlled LFSR's and Elliptic Curve cryptography. And it is different to the classic signature because in this case the equation that represents the Elliptic Curve is part of the proprietary secret, which is different to the classic system where this information is public.

This thesis work was developed to describe and simulate a cryptographic channel working in ideal conditions and make a software prototype of several schemes solutions. These schemes solutions are composed by signature schemes based on practical implementation of Elliptic curves in $GF(2^m)$. Later, the process used to obtain an Elliptic Curve and how it works will be described.

The widespread use of Smart Cards in many different sectors of everyday life makes them cheap and well documented. The reason that Smart Card technology is very diffused is because of their simplicity, memory capacity and processing capacity in a small and tiny card. However, the most important fact behind the use of Smart Cards is their anti-tamper property. In simple words, this means that it is not possible to access the information saved in it from people who do not know how the algorithm works.

All of this software was made using C language with the purpose of testing feasibility, estimated computation time and computational power needed. Also, a tool that identifies possible weaknesses and simulates possible attacks like store-and-reply as well as Known Cypher-text attacks was created.

In Chapter 1 of this work, a brief explanation of Linear Feedback Shift Registers (LFSR) is presented. The first thing examined is a general description of what they are and why are important in Cryptography is discussed. Secondly, some types of LFSRs, differences between them and different practical constructions are introduced. How it works, principal aspects about Clock Controlled LFSRs, and finally some examples of typical configurations used for Cryptography purposes will be discussed.

In Chapter 2, a brief introduction of Elliptic Curves is given including some mathematical details and discussion of a simple algorithm that generates them, using MAGMA software. This chapter represents the core of this project because, based on Elliptic Curves, a signature system used by basically all proposed schemes will be developed. Therefore, several examples of Elliptic curves in $GF(2^m)$ will be presented including the number of elements in this field and what convention is used to represent different points over the field in binary numbers. As the core of the project, this chapter will have more mathematic details and will show why Elliptic curves Cryptography is the most powerful, but least understood type of cryptography in wide use today.

In Chapter 3, different authentications protocols will be presented, showing their main characteristics, working conditions, computational power needed for each one, when practical implementation is required, and mathematical models that represent them. Also, their weaknesses will be scrutinized and attacks that might compromise their security.

In Chapter 4, attacks to different Crypto-systems will be elucidated. In general, this chapter is dedicated to show different attack models for Cryptanalysis. Basically Store-and-reply and Chosen Cypher-text attacks will be discussed and how they can compromise the different schemes propose in this project.

In chapter 5, anti-tamper systems like smart cards will be considered. General information will be presented, followed by how this project is benefited by the wide diffusion of the technology, especially the idea of low-cost implementation importance. Secondly, the variety of microprocessor smart sard, memory smart card, and touchless smart card will described, showing their main characteristics, working conditions and cost. At the end of this chapter, smart card readers and the software used to program them will be showed.

Chapter 6 is the main part of this work. Some authentication schemes will be presented along with their different characteristics, working conditions, limitations, and components. All the schemes presented in this chapter were simulated in C Language. A combination of clock controlled LFSRs and elliptic-curve signature algorithms produced a robust authentication scheme to work under restricted conditions, such as low power consumption, memory, and processing capacity.

In the last chapter of this work, some conclusions of the results obtained via simulation are presented, possible future work that can be developed. Finally, implementation details and viability using a certain type of Smart Card, and a chip that makes the stream sequence are showed.

Mathematical Theory Part

Chapter 1

Overview of LFSRs

1.1 Introduction

The primary goal of writing about shift registers in this study comes from the fact that every algorithm or authentication scheme proposed in this thesis work make use of Linear Feedback Shift Registers. For that reason, shift registers is the first chapter and the basic but important part in the signature process.

1.2 Shift Registers

A shift register is an interconnected set of Flip Flops sharing the same clock. Each Flip Flop has inputs and outputs. The output of each Flip Flop is connected to the 'data' input of the next Flip Flop in the set. That configuration would represent an array where each Flip Flop is a position of it. That configuration of these Flip Flops connected to the clock results in a circuit that shifts by one position the information stored in it. This information will be a bit saved in each Flip Flop.

There are several types of shift registers; depending on its inputs or outputs they can be classified as Parallel or Serial shift register. They can also be classified in a combination of these two configurations. This means that shift registers can has a serial input and serial output called (SISO) or serial input and parallel output called (SIPO). As well as when its input is parallel and its output is in parallel also, it is called (PIPO) or parallel input and serial output (PISO) or vice versa.

1.2.1 Serial-in and serial-out (SISO)

These are the simplest type of shift register. The data string is presented at 'Data In' and is shifted to the right each time the clock is enabled. As the result it generates the called 'Data Advance'. This will be the right shift version of the previous data. Basically, at each clock's period the bit on far left is shifted into the first Flip Flop's output and the bit on the far right is shifted out and lost. This configuration will be equivalent of a queue, like it is possible to see in the graphic:

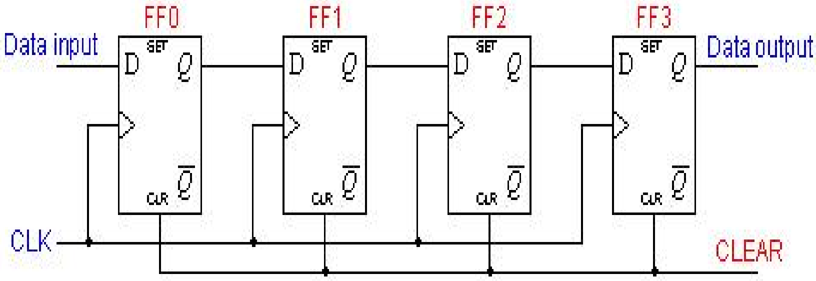


Figure 1.1. **Shift Register, SISO [16]**

1.2.2 Serial-in, parallel-out (SIPO)

This is a very common configuration called Serial-to-parallel converter also. It allows conversion from serial to parallel format. The data is input serially as described in SISO before. Once the clock is activated, it can be either shift out and replaced, or it may be read off at each output simultaneously. There are cases where parallel outputs do not change having a serial input, it would be a Buffer, being another application of this type of shift register.

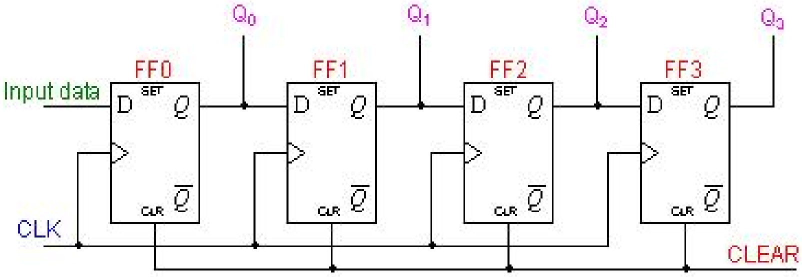


Figure 1.2. **Shift Register, SIPO [16]**

1.2.3 Parallel-in, Serial-out (PISO)

This configuration is also called Parallel-to-Serial converter. It allows converts parallel input to parallel output. As the previous configurations, to shift each bit it needs the clock to be activated. It can also work like a Buffer. It is easy to make it Buffer, is just disable the clock each time it needs. Is interesting to know that one of the first example of a shift registers knowing in the history was used in the first Electronic Digital Computer that was all programmable. This computer called Colossus was developed for British codebreakers during Worlds War II to help the Cryptanalysis of the Lorenz cipher. Basically this computer was created to help decrypt radio teleprinter messages that had been encrypted using the electromechanical Lorenz SZ40/42. It is easy to understand why this thesis work starts with types of shift register. The reason is because its relationship with Cryptography and Cryptanalysis. This will be a basic but not unimportant part of any LFSR cryptography system.

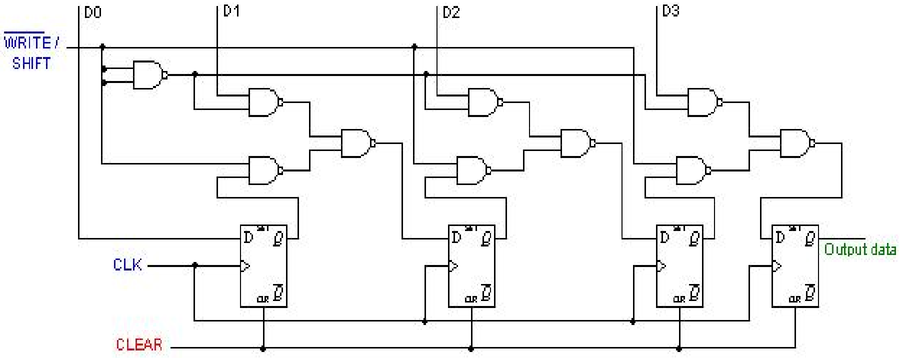


Figure 1.3. **Shift Register, PISO [16]**

1.3 Linear Feedback Shift Registers

An LFSR is basically a shift register. It allows the signal advances through the register from one bit to the next most-significant bit when it is clocked. There is a particularity on the LFSR and it is that some outputs are combined in exclusive-OR configuration to form a feedback mechanism. It has an initial value called 'seed', and because of how it works (in a deterministic way), the stream of values produced by the register is completely determined by its current or previous state. Therefore, the register has a finite number of possible states; it eventually enters in a repeating cycle. A linear Feedback Shift Register can be constructed by at least two Flip Flops. Essentially a LFSR will be made performing and exclusive-OR on the outputs of these Flip Flops, and feeding those outputs back into the input of one of the Flip Flops like it is possible to see in the figure: A Linear Feedback Shift Registers are described entirely by their characteristic Polynomial. For example,

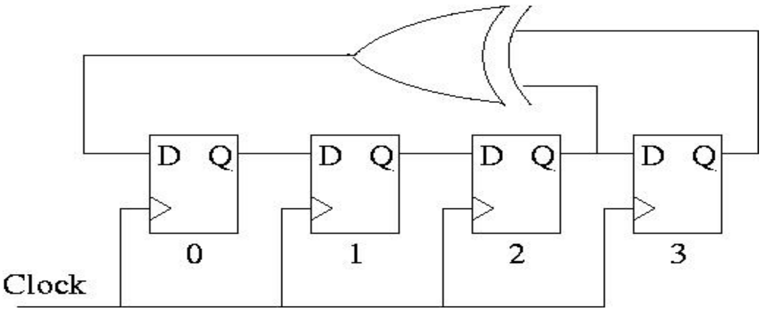


Figure 1.4. **Basic Linear Feedback Shift Register [19]**

a 5th-degree polynomial where all possible terms are present is represented with the equation $x^5 + x^4 + x^3 + x^3 + x^2 + x + 1$. For each degree, there can be many different primitive polynomials. These polynomials must satisfy the conditions of the primitive polynomials such as generation of all elements of an extension field from a base field and the irreducibility condition. Irreducibility condition means that the polynomial cannot be factored into nontrivial polynomials over the same field. For example, in the field of rational polynomials $Q[x]$ (i.e. polynomials $f(x)$ with rational coefficients), $f(x)$ is said to be irreducible if there do not exist two nonconstant polynomials $g(x)$ and $h(x)$ in x with rational coefficients such that

$$\mathrm{f(x)=g(x)h(x)} \tag{1.1}$$

For any prime or prime power q and any positive integer n, there exist a primitive polynomial of degree m over $GF(q)$. For example, x^2+x+1 has order 3 since [44]:

$$\frac{x+1}{x^2+x+1} = \frac{x+1}{x^2+x+1} \pmod 2$$

$$\frac{x^2+1}{x^2+x+1} = 1 + \frac{x+1}{x^2+x+1} \pmod 2$$

$$\frac{x^3+1}{x^2+x+1} = x+1 \pmod 2$$

1.3.1 Fibonacci Linear Feedback Shift Registers

Just to be agreed with vocabulary, Tap is a line that runs from the output of one register within the LFSR into an exclusive-OR, and determines the input to another

Flip Flop within the LFSR. Fibonacci LFSR is a type of LFSR where the outputs from some of the Flip Flops are connected with exclusive-OR or exclusive-NOR gates with each other and feed back to the input of the shift register.

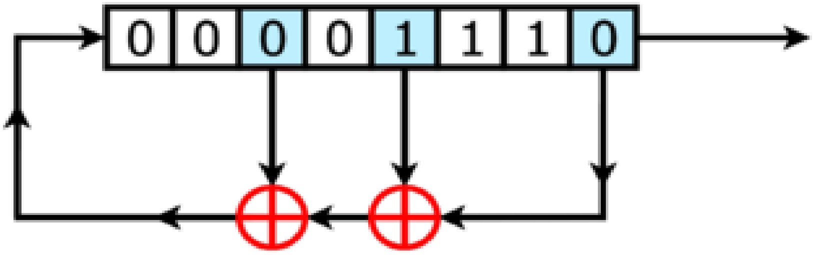


Figure 1.5. **Fibonacci Representation [40]**

When the shift register is loaded with a seed value and then clocked, the output from the LFSR will be a pseudo-random sequence of 1's and 0's. The length of the pseudo-random sequence is dependent on the length of the shift register and the position of the feedback taps. The number and the position of the taps are commonly represented like it was said before by a polynomial.

1.3.2 Galois Linear Feedback Shift Register

Galois LFSR is another type of LFSR. It is also known as modular, internal XORs as well as one-to-many LFSR. It is an alternate structure that generates the same output stream as a conventional LFSR, but offset in time. In this representation the gates are placed between the Flip Flops. Therefore, when the clock is enabled, or better, when the system is clocked, bits that are not connected to any tap are shifted one position to the right unchanged. On the other hand, bits that are connected to the taps are XORed or NORed with the output bit before they are stored in the next Flip Flop; as shown in Figure 1.6 The main advantage of Galois representation is its efficiency. Galois LFSR form in a software representation is more efficient than the Fibonacci LFSR, because XOR or NOR operations can be implemented one word at a time.

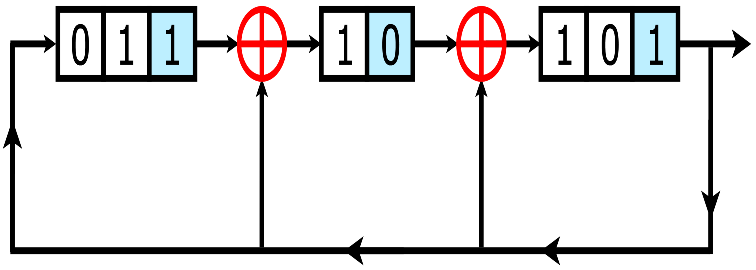


Figure 1.6. **Galois Representation [40]**

1.4 LFSRs, Stream Ciphers and Non Linear Function (NLF)

LFSRs can be used for a number of practical applications. For example, applications that require very fast generation of a pseudo-random sequence, such as direct-sequence spread spectrum radio. Also they can be used either white noise generator, or programmable sound generators, or as counters circuit testing, or Test-pattern generation, or signature analysis. However, due the ease construction from simple electromechanical or electronic circuits, long periods, and very uniformly distributed output stream they have long been used as pseudo-random number generators for use in stream ciphers.

In this project work, LFSRs are used to output one bit at a time either to use to encrypt a message, or to generate one bit of a 'random' number. Now given a general view of what LFSRs are, the natural question will be how exactly was used this Cryptography application in this project?

Firstly, to be clear, a stream cipher, or state cipher is a symmetric key cipher where either a pseudorandom number or pseudorandom cipher digit stream is combined with a plaintext or message. In other words, each bit of the message or plaintext is encrypted one at a time with the corresponding bit or digit of the key stream, as the result a bit or digit of the ciphertext is obtained. In this thesis the LFSR does the function of the pseudorandom generator. However, due to its cyclical and predictable nature, a single LFSR cannot be used for a stream cipher. It would be a very weak crypto-sytem. Therefore, multiple LFSR are connected together in a variety ways. One way can be in parallel like is shown in Figure 1.7

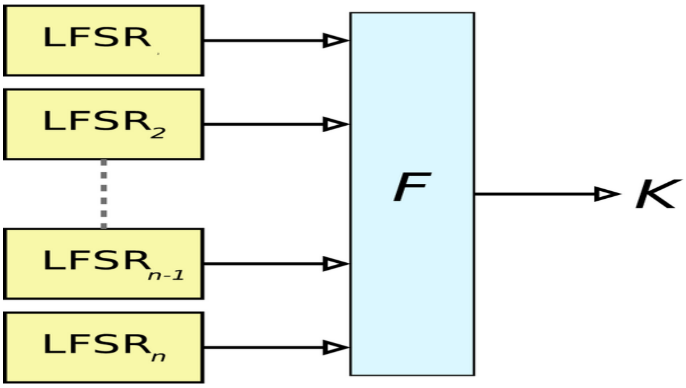


Figure 1.7. **General LFSR connection [41]**

In this configuration it is important to note that for each time the system is clocked, only one bit is output at a time. Then, it does not show too much information about the private or internal state.

Continuing with this configuration, the idea is to put n registers together (LFSR 0, LFSR 1 to LFSR n; n can be any bit length as the applications allows), and feed their output bits into define Boolean function f. That function contain a define bit logic. For example, a formula that takes bits from the LFSR 0 and ANDed with some bits from the LFSR 1 outputting a single bit for the 'keystream.' The function f can be a simple AND or NAND of all the input bit, or it can be as complicated as the designer of the system requires. The main purpose of this scheme is to make the function f a nonlinear function. Therefore, that function stave off very powerful attacks as 'Correlation attacks.' In Attacks section Correlation attacks will be described with more details.
There is another configuration called Clock-Controlled LFSR where mixing in a properly way can make a LFSR-based stream cipher strong enough. The idea of mix LFSR is to introduce non-linearity. Then, within this scheme at least two LFSR are used. One of them will be an irregular controlled by the output of the second LFSR. In other words, if three LFSRs are used, the output of one of the registers decides which of the other two is to be used; for instance, if LFSR 0 outputs a 0, LFSR 1 is clocked, and if it outputs a 1, LFSR 2 is clocked instead. The output is the exclusive OR of the last bit produced by LFSR 1 and LFSR 2. Here it is important to note that the state of the three LFSRs will be the key. There are two variants of the Clock-controlled LFSR. One of them is the Stop-and-go generator. This consists, basically, of two LFSRs. One LFSR is clocked if the output of a second is a 1, otherwise it repeats its previous output. Sometimes the output is

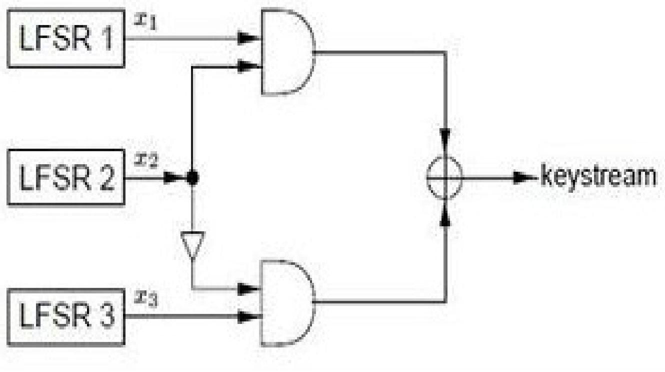


Figure 1.8. **General Representation of a Non-Linear Function [26]**

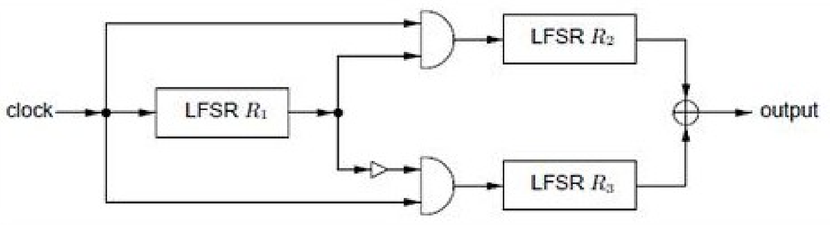


Figure 1.9. **Clock Controlled LFSR [26]**

then combined with the output of a third LFSR with a regular clock system.

Another variant of Clock-Controlled LFSR is the Shrinking generator. In this case two regular controlled LFSR are used. However, when the first LFSR outputs 1, the output of the second one becomes the output of the stream cipher. But, if the output of the first LFSR is a 0, the output of the second LFSR is discarded, and no bit is output by the stream cipher. The problem of this stream cipher configuration is that the speed is variable, depends on the output of the first LFSR. However, it can be solve by buffering the output.

Chapter 2

Elliptic Curves

2.1 Introduction

Elliptic curve cryptography (ECC) was discovered in 1985 by Victor Miller (IBM) and Neil Koblitz (University of Washington) as an alternative mechanism for implementing public-key cryptography [7]. Being the most important part of this thesis work, elliptic curves will be discussed in detail. Elliptic curves provide one of the most powerful and useful way to encrypt information. It has been increasing the number of websites that use elliptic curves to secure customers' HTTP connections, and the way of how they pass data between data centers and costumers. Therefore it is important that end users understand the technology behind any security system in order to trust it. Elliptic curve cryptography is part of modern cryptography, it is founded on the idea that the key that each user use to encrypt data can be made public while the key used to decrypt must be private. Thus, these types of systems are known as public key cryptography systems. A public key system is needed because of the fact that it provides algorithms that are easy to process in one direction but difficult to undo. For example in RSA system, the easy algorithm to multiply two prime numbers comprises encryption, whereas, factoring the product of the multiplication into two prime numbers is difficult. Undoing the function, i.e., factoring is presumably difficult. There are several modern cryptography schemes based on Elliptic Curves such as:

- The Elliptic Curve Diffie-Hellman (ECDH)
- The Elliptic Curve Integrated Encryption Scheme (ECIES), also known as Elliptic curve Augmented Encryption Scheme or simply the Elliptic Curve Encryption Scheme.
- The Elliptic Curve Digital Signature Algorithm (ECDSA), this one is based on the Digital Signature Algorithm

- The Elliptic Curve Menezes-Qu-Vanstone (ECMQV)
- And the Elliptic Curve QV implicit certificate scheme.

All these algorithms have the primary benefit promised by Elliptic Curve Cryptography. It means that they use a smaller key, taking advantage on reducing storage and requirements. However, ECC provides the same level of security afforded by an RSA-based system with a large modulus and correspondingly larger key. For example, a 256-bit ECC public key should provide comparable security to a 3072-bit RSA public key. [36]

Symmetric Key Size (bits)	**RSA and Diffie-Hellman Key Size (bits)**	**Elliptic Curve Key Size (bits)**
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Table 2.1. **NIST Recommended key size [23]**

2.2 Elliptic curve over $E(\mathbb{F}_{2^m})$ or Binary Fields

As its name suggests, Binary fields are fields where the elements are binary polynomials. Finite fields of order 2^m are also called characteristic-two finite fields. The most popular and widely used application of elliptic curve over Galois Field is in Cryptography. Since each byte of data is represented as a vector in a finite field, encryption and decryption using mathematical arithmetic is very straightforward and is easily manipulable. One way to construct $\mathbb{F}_{2^m}$ is to use a polynomial basis representation. Polynomials whose coefficients are in the field $\mathbb{F}_2 = 0{,}1$ and have degree at most $m-1$,

$$\mathbb{F}_{2^m} = a_{m-1}z^{m-1} + a_{m-2}z^{m-2} + \cdots + a_2z^2 + a_1z + a_0 \ : a_i \in 0{,}1$$

There is a small difference between an elliptic over $\mathbb{F}_p$ and an elliptic curve over $\mathbb{F}_{2^m}$. An elliptic curve E over $\mathbb{F}_{2^m}$ is defined by an equation of the form

$$y_2 + xy = x^3 + Ax^2 + B$$

Where $A{,}B \in \mathbb{F}_{2^m}$, and $B \neq 0$. The set $E(\mathbb{F}_{2^m})$ consists of all points $(x{,}y){,}x \in \mathbb{F}_{2^m}$, $y \in \mathbb{F}_{2^m}$, which satisfy the defining equation above. Together with the special point $\mathcal{O}$ called the point at infinity.

Example: Suppose an elliptic curve over $\mathbb{F}_{2^4}$; this field is represented by the irreducible trinomial

$$f(x) = x^4 + x + 1$$

Consider the elliptic curve $E\ : y^2 + xy = x^3 + \alpha^4 x^2 + 1$ over $\mathbb{F}_{2^4}$. Given continuity to the notation, in this case $A = \alpha^4$ and $B = 1$. It is not obvious, given $B \neq 0$, that the curve E is indeed an elliptic curve. Therefore, the points in $E(\mathbb{F}_{2^4})$ are $\mathcal{O}$ and the following

$$\begin{array}{ccccccc}
(0,1) & (1,\alpha^6) & (1,\alpha^{13}) & (\alpha^3,\alpha^8) & (\alpha^3,\alpha^{13}) & (\alpha^5,\alpha^3) & (\alpha^5,\alpha^{11}) \\
(\alpha^6,\alpha^8) & (\alpha^6,\alpha^{14}) & (\alpha^9,\alpha^{10}) & (\alpha^9,\alpha^{13}) & (\alpha^{10},\alpha) & (\alpha^{10},\alpha^8) & (\alpha^{12},0) \\
(\alpha^{12},\alpha^{12}) & & & & & &
\end{array}$$

As with elliptic curves over $\mathbb{F}_p$, there are two ways to add two points. One is an algebraic way and another is the geometrically way. The latter is also called chord-and-tangent rule.
The algebraic way has the following formulas for adding and doubling a point

- $P + \mathcal{O} = \mathcal{O} + P$ for all $P \in E(\mathbb{F}_{2^m})$
- If $P = (x,y) \in E(\mathbb{F}_{2^m})$, then $(x,y) + (x,x+y) = \mathcal{O}$. (The point (x,x+y) is denoted by $-P$, and is called the negative of P. Observe that $-P$ is indeed a point on the curve).
- (Point addition) Let $P = (x_1,y_1) \in E(\mathbb{F}_{2^m})$ and $Q = (x_2,y_2) \in E(\mathbb{F}_{2^m})$, where $P \neq \pm Q$. Then $P + Q = (x_3,y_3)$, where

  $$x_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)^2 + \frac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2 + A$$

  and

  $$y_3 = \frac{y_1 + y_2}{x_1 + x_2}(x_1 + x_3) + x_3 + y_1$$

- (Point doubling). Let $P = (x_1,y_1) \in E(\mathbb{F}_{2^m})$, where $P \neq -P$. The $2P = (x_3,y_3)$, where

  $$x_3 = x_1^2 + \frac{B}{x_1^2}$$

  and

  $$y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right) x_3 + x_3$$

Example: Consider the elliptic curve $y^2 + xy = x^3 + \alpha^4 x^2 + 1$. Let $P = (\alpha^6,\alpha^8)$ and $Q = (\alpha^3,\alpha^{13})$. Then $P + Q = (x_3,y_3)$ is computed as follows:

$$x_3 = \left(\frac{\alpha^8 + \alpha^{13}}{\alpha^6 + \alpha^3}\right)^2 + \frac{\alpha^8 + \alpha^{13}}{\alpha^6 + \alpha^3} + \alpha^6 + \alpha^3 + \alpha^4 = \left(\frac{\alpha^3}{\alpha^2}\right)^2 + \frac{\alpha^3}{\alpha^2} + \alpha^6 + \alpha^3 + \alpha^4 = 1$$

and

$$y_3 = \frac{\alpha^8 + \alpha^{13}}{\alpha^6 + \alpha^3}(\alpha^6 + 1) + 1 + \alpha^8 = (\frac{\alpha^3}{\alpha^2})(\alpha^1 3) + \alpha^2 = \alpha^{13}$$

Hence $P + Q = (1,\alpha^{13})$.

Now, an example of doubling a point in a binary field:

Let $P = (\alpha^6,\alpha^8)$. Then $2P = P + P = (x_3,y_3)$ is computed as follows:

$$x_3 = (\alpha^6)^2 + \frac{1}{(\alpha^6)^2} = \alpha^{12} + \alpha^3 = \alpha^{10}$$

and

$$y_3 = (\alpha^6)^2 + (\alpha^6 + \frac{\alpha^8}{\alpha^6})\alpha^{10} + \alpha^{10} = \alpha^{12} + \alpha^{13} + \alpha^{10} = \alpha^8$$

Hence $2P = (\alpha^{10},\alpha^8)$.

There are two other important operations in Binary fields different from addition and doubling. For example, when an application needs *n times* the point P, instead of adding *n times* P the multiplication operation is used. Multiplication in a finite field is a multiplication modulo an irreducible polynomial, where, at the same time, this polynomial is used to define the finite field. The symbol '.' may be used to denote multiplication in a finite field.
For example, in a binary field $E(\mathbb{F}_{2^4})$, where the elements in polynomial representation are

$$\begin{array}{cccc}
0 & \alpha^2 & \alpha^3 & (\alpha^3 + \alpha^2) \\
1 & (\alpha^2 + 1) & (\alpha^3 + 1) & (\alpha^3 + \alpha^2 + 1) \\
\alpha & (\alpha^2 + \alpha) & (\alpha^3 + \alpha) & (\alpha^3 + \alpha^2 + \alpha) \\
(\alpha + 1) & (\alpha^2 + \alpha + 1) & (\alpha^3 + \alpha + 1) & (\alpha^+\alpha^2 + \alpha + 1)
\end{array}$$

All these elements are generated by the reduction polynomial or primitive polynomial $f(\alpha) = \alpha^4 + \alpha + 1$. For instance, if it is needed to multiply two elements in the field, the result will be obtained by making operation *module* of the reduction polynomial.

Multiplication:

The following equation is true

$$(\alpha^3 + \alpha^2 + 1).(\alpha^2 + \alpha + 1) = \alpha^2 + 1$$

since

$$(\alpha^3 + \alpha^2 + 1).(\alpha^2 + \alpha + 1) = \alpha^5 + \alpha + 1$$

and

$$\alpha^5 + \alpha + 1 \mod (\alpha^4 + \alpha + 1) = \alpha^2 + 1$$

Another important operation in Binary fields is the inversion multiplication or just inversion. The idea of this operation is to find an element in the field that divided by itself would result in 1. For example, if the application needs invertion of the element $\alpha^3 + \alpha^2 + 1$, the results is α^2. It means that

$$(\alpha^3 + \alpha^2 + 1)(\alpha^2) \mod (\alpha^4 + \alpha + 1) = 1$$

In general, elliptic curves over $GF(2^m)$ are particularly attractive, because the finite field operations can be implemented efficiently in hardware and software. Normally, the most time-consuming operation of the elliptic curve cryptosystems is the multiplication. Therefore, there are several papers that discuss this subject and attempt to reduce the number of basic operations to obtain the result of the multiplication of two elements in a Binary field.

2.2.1 Fast operations

In the research conducted by Julio Lopez and Ricardo Dahab [18], they describe two methods to compute kP. It means add k times P. They first developed an algorithm in affine coordinates which required two field inversions in each iteration. Next, they developed a 'projective version' that implied more field multiplications, but with only one field inversion at the end of the computation. This is an important point because the time computation taken by the field inversion is bigger than the field multiplication. Basically, the Julio Lopez and Ricardo Dahab algorithm avoids inverse operations.

In this thesis work, the projective version algorithm obtained by Julio Lopez and Ricardo Dahab was used. The reason why this thesis uses this algorithm is because of its performance and desired objectives. The algorithm was 2P Montgomery Scalar Multiplication, where the input is an integer $k \geq 0$ and a point $P = (x,y) \in E$ and the output is $Q = kP$.

This algorithm uses an implementation of procedures called Madd, Mdouble and Mxy that are given in the Julio Lopez and Ricardo Dahab work [18]. Its steps are:

```
1. If $k = 0$ or $x = 0$ then output(0,0) and stop
2. Set $k \leftarrow (k_{l-1} \dots k_1 k_0)_2$
3. Set $X_1 \leftarrow x$,  $Z \leftarrow 1$,  $X_2 \leftarrow x^4 + B$,  $Z_2 \leftarrow x^2$
4. For $i$ from $l - 2$ downto 0 do
   If $k_i = 1$ then
   $Madd(X_1,Z_1,X_2,Z_2)$,  $Mdouble(X_2,Z_2)$
   Else
   $Madd(X_2,Z_2,X_1,Z_1)$,  $Mdouble(X_1,Z_1)$
5. Return $(Q = Mxy(X_1,Z_1,X_2,Z_2))$
```

Where l is the number of bits in binary representation of k. In this binary representation, the most significant bit is on the left and the least significant bit is on the right. This algorithm performs exactly the following number of field operations in $GF(2^m)$:

- INVERSIONS $= 1$
- $MULTIPLICATIONS = 6\lfloor log_2 k \rfloor + 10$
- $ADDITIONS = 3\lfloor log_2 k \rfloor + 7$
- SQUARE ROOTS $= 5\lfloor log_2 k \rfloor + 3$

2.3 Basic facts in elliptic curve cryptography ECC

There are two basic facts to take into account about elliptic curve and Finite Fields. The first is group order. Hasses's theorem states that the number of points on an elliptic curve (including the point at infinity) is $\#E(\mathbb{F}_p) = q + 1 - t$ where $|t| \leq 2\sqrt{q}$; $\#E(\mathbb{F}_p)$ is called the order of E and t is called the trace of E. In other words, the order of an elliptic curve $E(\mathbb{F}_p)$ is roughly equal to the size q of the underlying field. Another basic fact is the group structure. $E(\mathbb{F}_p)$ is an abelian group of rank 1 or 2. That is, $E(\mathbb{F}_p)$ is isomorphic to $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$, where n_2 divides n_1, for unique positive integers n_1 and n_2. Here, $\mathbb{Z}_n$ denotes the cyclic group of order n. Moreover, n_2 divides $q - 1$. If $n_2 = 1$, then $E(\mathbb{F}_q)$ is said to be cyclic. In this case $E(\mathbb{F}_q)$ is isomorphic to $E(\mathbb{Z}_{n_1})$, and there exists a point $P \in E(\mathbb{F}_q)$ such that $E(\mathbb{F}_q) = \{kP \ : 0 \leq n_1 - 1\}$; such a point is called a generator of $E(\mathbb{F}_q)$ [7].

Example: Consider the elliptic curve $y^2 = x^3 + x + 4$ over $E(\mathbb{F}_{2^3})$. Since $E(\mathbb{F}_{2^3}) = 29$, which is a prime number, $E(\mathbb{F}_{2^3})$ is cyclic and any point other than $\mathcal{O}$ is a generator of $E(\mathbb{F}_{2^3})$. For example, P = (0,2) is a generator as the following shows:

$1P = (0,2)$	$2P = (13,12)$	$3P = (11,9)$	$4P = (1,12)$	$5P = (7,20)$
$6P = (9,11)$	$7P = (15,6)$	$8P = (14,5)$	$9P = (4,7)$	$10P = (22,5)$
$11P = (10,5)$	$12P = (17,9)$	$13P = (8,15)$	$14P = (18,9)$	$15P = (18,14)$
$16P = (8,8)$	$17P = (17,14)$	$18P = (10,18)$	$19P = (22,18)$	$20P = (4,16)$
$21P = (14,18)$	$22P = (15,17)$	$23P = (9,12)$	$24P = (7,3)$	$25P = (1,11)$
$26P = (11,14)$	$27P = (13,11)$	$28P = (0,21)$	$29P = \mathcal{O}$	

2.4 Basic Idea of Elliptic Curve Cryptography

In this section, the basic idea of an elliptic curve cryptosystem will be described, including how and what makes elliptic curve a very strong cryptography system, based on the concept of finite or Galois Field and its laws. Imagine a set of points (x_i,y_i) in a plane. Imagine this set is very, very large but finite; let's call the set E. Next, imagine it is possible to define a group operator in this set. As it was described in the Finite Field section, a group operator is typically denoted by the symbol $'+'$ even when the operation itself is completely different from the ordinary addition. Then, given two points P and Q in the set E, the group operator $'+'$ allows for the calculation of a third point R, also in the set E, such that:

$$P + Q = R$$

Therefore, given a point $p \in E$, it is particularly interesting to use the group operator to find either $p+p$, $p+p+p$, or $p+p+p+p$, or $p+p+p+\cdots+p$ for an arbitrary number of repeated invocations of the defined group operator. Now, it is appropriate to define an ordinary integer k, and use it to define the notation $k*p$ to represent the repeated addition of $p+p+p+\cdots+p$ in which p makes k appearances, with the $'+'$ operator involved $k-1$ times. It is important to say that $k*p$ is not an attempt to define a multiplication operator on the set E. That is because k is an ordinary number, it is not part of the set E. It is just to be associated with repeated addition.

Next, imagine that the set E is magical in the sense that, after being calculated $k*p$ for a given point $p \in E$, it is extremely difficult to recover k from the result of $k*p$. This means that the only way to recover k from the result of $k*p$ is to try every single summation $p+p$, $p+p+p$, $p+p+p+p$, $p+p+p+\cdots+p$ until $k*p$ is obtained. In other words, figuring out how many times p participates in the repeated sum is referred to as solving the Discrete Logarithm Problem. Just to provide an idea of what that implies, let's consider the traditional notion of logarithm that allows to write $a^k = b$ as $k = log_a b$. Obviously, a^k is $a * a * a * a * a * \cdots * a$ with a k appearances in the repeated invocations of the binary operator involved. This is the same as what the person who wants to break into the Elliptic Curve Cryptosystem does in order to determine the values of k from the result of $k * p$. This means, the third party (the person who wants to break into the Elliptic Curve Cryptosystem) wants to find out how many times p participates in the repeated invocations of the $'+'$ operator.

It is important to remember that the operator $'*'$ does not mean multiplication; it is a shortcut for denoting the repeated adition $p + p + p + \cdots + p$ involving k apareances of p.

Cryptography Part

Chapter 3

Authentication protocols

3.1 Introduction

In this chapter, common authentication protocols will be presented. An overview of several authentication methods and authentication protocols will be discussed as well as their advantages, disadvantages, and a basic scheme of how they work. This thesis used elliptic curve cryptography to create different types of Extensible Authentication Protocols (EAP) using smart card. This study understands "authentication" as the process of determining whether someone or something is, or is not, who or what they claim to be. This process can be accomplished in many ways; however, the selection of an authentication protocol in a determined environment is perhaps the most important decision in designing secure systems. The "determined environment" refers to all the characteristics and restrictions that will be used when the protocol is implemented.

3.2 Types of Authentication Protocols

In user authentication, there are four techniques to authenticate a user's identity:

1. Knows. (e.g., password, PIN)
2. Possesses. (e.g., Smart card, key or token)
3. Is. (e.g., Fingerprint, retina)
4. Does. (Dynamic biometrics). e.g., voice, sign.

These techniques can be used alone or in combination depending on the application. All these types of authentications have issues such as complex hardware, excessive time computation, or an overwhelming number of weaknesses and, therefore, cannot supply the needed level of security. Given the four general divisions of authentication types, select common authentication methods and protocols will be outlined.

3.2.1 Password Authentication Protocol (PAP)

The PAP is the most simple, cheap, and convenient of all authentication protocols. Essentially, it transmits all information or authentication details without any type of encryption. As a result of this important characteristic, the PAP possesses a significant security risk. For instance, an unauthorized user could access this information using or knowing exactly how the protocol works.

The advantages of PAP are that it does not require complicated or robust hardware since authentication of this type is generally simple and does not require much processing power. However, there are significant disadvantages of this protocol. The following are the most evident:

- Password may be easy to guess
- Password can be accidentally discovered when users write it down
- Password can be obtained by social engineering

To avoid these problems associated with PAP, the one-time password was developed. There are two types of one-time passwords: a challenge-response password and a password list. As its name suggests, the challenge-response password responds with a challenge value after a user asks for authentication on the base station. In a one-time password list, the base station makes use of lists of the passwords that have been sequentially utilized by the user. These values are generated so that it is very hard to calculate the next values from the previous one.

3.2.2 Smart card

The second method of authenticating a user is possible by smart card. A smart card is a portable device that has a CPU, memory, some input/output ports and power ports. By using the input/output ports the reader can access the memory through the card's CPU. The smart card can perform any of the validation techniques; they are something the user possesses. Smart card technology provides an excellent platform for implementing strong authentication. Smart cards can support and protect authentication tokens, password files, and one-time password seed files. The smart card also generates asymmetric key pairs. Smart cards used with another authentication system like tokens can provide a significantly strong logical access security.

The smart card can be used for both physical and logical access authentication, enhancing the security and privacy of the whole authentication system.

In addition, smart cards can support a variety of the applications at this time such as password management, virtual private network authentication, e-mail and data encryption, and electronic signatures. Physically smart cards are available in multiple form factors, such as plastic cards (with contact or contactless communication capabilities, or both), a USB device, or a secure element that can be embedded in a mobile phone or remote control in general.

For example, consider the challenge/response scheme, where a user asks for an authentication, the base station sends to the user a challenge and the user responds to the challenge with its information. In a normal system, the user would need to possess their own private key which is used to respond to the challenge sent from the base station. In order to make the system secure, no one other than the user may know the key. In this authentication method, the smart card can save the private key better than any other system; the smart card with its anti-tamper property makes it possible that any person who does not know how the key is saved will not have access to this information.

A possible disadvantage of this authentication type is that users have to carry the device any time they want to authenticate in the system.

3.2.3 Biometrics

The third method of authenticating a user attempts to measure something intrinsic to the user. This intrinsic thing could be a fingerprint, a voiceprint, or a physical signature. The main advantage of this type of authentication is that a biometric identifier can neither be given away nor stolen. Fingerprints, voice, retinal and iris patterns are also virtually unique to each individual. This property makes a biometric authentication more secure than smart card-based authentication. This method of proving one's identity is very difficult to falsify, although it requires expensive equipment to input the fingerprints, voice, or eye scan. Another advantage over smart cards is that the user does not have to carry a device or a card; his or her biological credentials are never left at home.

In practice, there are some limitations with this type of authentication. No two signatures are absolutely identical, even from the same user without taking into account that the user can be sick, exhausted or in different moods. For that reason, some systems based on biometrics measurements use smart cards to store the biometric data about each user. This avoids the need for host databases and instead relies on the security of the card to prevent tampering. It is also possible to include a challenge in the base station that will be sent to the user in order to make the system more secure and to avoid replay attacks.

One of the main disadvantages of this authentication protocol is the fact of using special hardware. This disadvantage limits the applicability of biometric techniques of comparatively few environments. The main advantage is its security and reliability.

Chapter 4

Attacks

4.1 Introduction

In this section some types of cryptanalysis attacks will be presented. It is important to discuss how secure the system is and what type of attacks can compromise the security of the proposed schemes in this thesis work. The first thing presented will be the cryptanalytic attacks depending on the amount of information available to the attacker. It will be classified based on what type of information the attacker has available. It is also important to know that in these types of attacks it is assumed that the general algorithm is known. Just to provide a general idea of what types of attacks exist in cryptanalysis, the next scheme is presented:

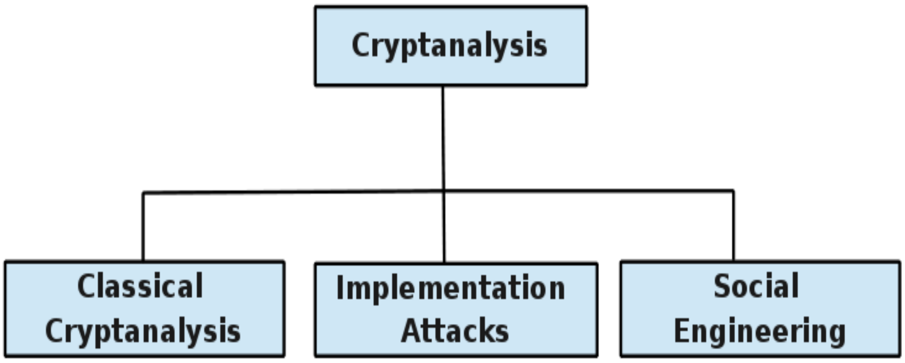


Figure 4.1. **Cryptanalysis Scheme**

4.2 Ciphertext-only attack (COA)

In general, classical cryptanalysis attacks are classified in four categories. These categories can be distinguished based on the kind of information the cryptanalysis has available to mount an attack. The Ciphertext-only attack is the first one on the list with more difficulty to the cryptanalyst to break a cryptosystem. In other words, this attack is one in which the cryptanalyst obtains a sample of ciphertext without the plaintext that generates it. In general, this information (ciphertext) is relatively easy to obtain, but a successful ciphertext-only attack is usually difficult, and requires a large ciphertext samples. It is important to know that a successful attack is given when the corresponding plaintext from the ciphertext is deduced, or better, the key is used to generate the ciphertext. Any information from the plaintext is considered a success. As previously stated, this attack is the most difficult for the Cryptanalyst. Therefore, only relatively weak algorithms can be broke to withstand a ciphertext-only attack.

4.3 Known Plaintext attack (KPA)

In this type of attack, the cryptanalyst has samples of both plaintext and the ciphertext that generates it. With these samples, the cryptanalyst is trying to find the key, or better, to decrypt other ciphertext that it supposes were generated with the same key. Examples of classical cryptosystems typically vulnerable to this type of attack are Caesar cipher and a general monoalphabetic substitution cipher. In modern ciphers such as AES (Advanced Encryption Standard) security is not compromised with a Known plaintext attack.

4.4 Chosen-ciphertext Attack (CCA)

This attack is one in which cryptanalyst has the possibility to decrypt part of the ciphertext and attempt to obtain the corresponding plaintext. In other words, the third party may obtain information by choosing a ciphertext and obtaining part of the plaintext under an unknown key. In this attack, the cryptanalyst gathers information because of having a chance to enter one or more known ciphertexts into the cryptosystem, and in that way, it is able to obtain the equivalent or resulting plaintext. Chosen-ciphertext attack is categorized as a dangerous attack for the security of the cryptosystems because of its quality of information that it brings to the cryptanalyst. There are schemes that can be broke under chosen ciphertext attacks like El Gamal cryptosystem, and early versions of RSA systems. In this thesis, this type of attack is particularly important because of the use of a smart card or tamper-resistant system. This attack can compromise security when a smart card is used because a third party has the possibility to issue a large number of Chosen-ciphertext with the objective to recover the hidden secret key. For that reason, it is important to be particularly cognizant of these attacks.

As with other types of attacks, Chose-ciphertext Attack may be adaptive or non-adaptive. In this attack, the adaptive Chosen-ciphertext attack (CCA2) is when the cryptanalyst has the chance to select ciphertexts dynamically before and after a challenge. In other words, an adaptive Chosen-ciphertext attack is an interactive attack in which the cryptanalyst may send more than one ciphertexts to be decrypted and then the cryptanalyst can use these results to select future ciphertexts. In simple words, the adaptive Chosen-ciphertext occurs when the third party is able to change the text input into the cipher based on the results of previous inputs. [ref. http://www.tech-faq.com/known-ciphertext-attack.html]

4.5 Chosen Plaintext Attack CPA

This is most dangerous attack, and it occurs the cryptosystem is considered broke. This attack takes place when the cryptanalyst has access to the encryption machine. The attacker can encrypt plaintexts of their choice. He or she is able to define his/her own plaintext, feed it into the cryptosystem and analyze the resulting ciphertext. Because of the requirements of this type of attack, mounting a chosen plaintext attack is in some cases impossible to attempt.

Chosen plaintext attack may also be adaptive. That means, the cryptanalyst is able to make a sequence or series of interactive queries, or better, the cryptanalyst can use the cryptosystem more than one time and analyze the cyphertext results.

4.6 Implementation Attacks

As it is shown in the Figure 4.1, implementation attacks are a way to practice cryptanalysis. In order to discover the key, implementation attacks take a different approach to the classical cryptanalysis attacks. Unlike the classical attacks in which they attack the mathematical properties, the implementation attacks take advantage of the physical phenomena when the cryptography algorithm is working in the respective hardware. There are four channel attacks of the implementation attacks listed in the FIPS standard 140-2 'Security requirements for Cryptography Modules,' these are:

4.6.1 Power Analysis

This way to attack is based on the analysis of power consumption. It can be divided into two categories: Simple Power Analysis (SPA) and Differential Power Analysis

(DPA). Basically, Simple Power Analysis consists in analyzing electrical power consumption patterns and timings derived from the execution of individual instructions in the cryptosystem. Those patterns obtained by this type of analysis are the result of monitoring the variations in electrical power consumption of a cryptography module for the purpose of revealing the value of the cryptography key. The difference of SPA and DPA is that DPA utilizes advanced statistical methods or techniques to analyze electrical power consumption. However, DPA and SPA have the same goal.

When the cryptosystem uses smart cards, it is important to know that Differential Power Analysis has proven to be quite effective in attacking smart card-based systems. Fundamentally, Power Analysis consists in analyzing the pattern of power consumption by the cryptosystem that performs the cryptography operations. In general, power consumption is different for each operation and even for the same operations with different values. Basically, the cause of the power consumption pattern is because of the transistor technology. This means, given the transistor acts as a voltage-controlled switch, and the power that it consumes varies or depends on the instructions processed by the module, it is possible to see a fluctuation of the power signal.

4.6.2 Timing Analysis

As it name suggests, timing analysis consists of analyzing the taken time by the cryptography module when it is performing a specific operation. This time is related or has a relationship between the input and the key used by the cryptography algorithm. It is important to know that in this type of attack, the cryptanalyst has knowledge of how the cryptographic module works.

4.6.3 Fault Induction

Basically, the fault induction uses any type of technology that induces an error within the cryptographic module. As an example, a type of fault induction attack can be the use of technology that increases the temperature of the module causing errors in it; the cryptanalyst analyzes those errors and their behavior to make reverse engineering possible for the cryptography module, revealing certain features of the cryptography algorithms and then values of the private keys. Therefore, it is important that the cryptography module has a proper physical security.

4.6.4 TEMPEST

In general, all electronic devices may emit low-level electromagnetic radiation. This happens in any electronic device that makes changes in the electric current or makes possible the generation of electromagnetic pulses radiating as invisible waves. Basically, a TEMPEST attack is a passive attack through which the electromagnetic radiation generated by the cryptography module helps to reconstruct the signal from a remote location. For example, electromagnetic radiation may come from the keystroke information, messages displayed on a video screen or operations processed by the cryptography algorithm.

Practical Implementation Part

Chapter 5

Smart Cards

5.1 Introduction

Smart card technology is used extensively in different areas such as telecommunications, government, financial, healthcare, identity and transportation. For example, in telecommunications, smart card technology is used in two applications such as prepaid telephone cards and as the microprocessor smart card-based Subscriber Identity Module (SIM) in mobile phones, just to provide an example. A smart card can be also a card carrying digital 'money' which makes it possible to pay for a ticket or to buy food in a supermarket. In this chapter, different types of smart cards, including their features and benefits will be described. Also, a small introduction about the smart card hardware, terminology and concepts will be provided.

5.2 Memory Smart Card

In general, there are three types of smart cards: the contact smart card, the contactless smart card and the multicomponent card. Within the first two types of smart cards there is a big subdivision. This subdivision is in memory cards and microprocessor cards, as is shown in Figure 5.1. In this subsection, a brief explanation of memory cards, their applications and different capacities will be discussed.

Memory cards are the most common and cheapest of the smart cards. This type of smart card contains EEPROM (Electrically Erasable Programmable Read-Only Memory), or non-volatile memory. Due to this property (non-volatile memory), the smart card stores the data either when the card is removed from a smart card reader or power is cut off. It is possible to think of EEPROM memory as just like a data storage device which has a file system and is managed via a microcontroller. This microcontroller is responsible for the access to the files and managed the communications between the memory and smart card reader. This microcontroller also

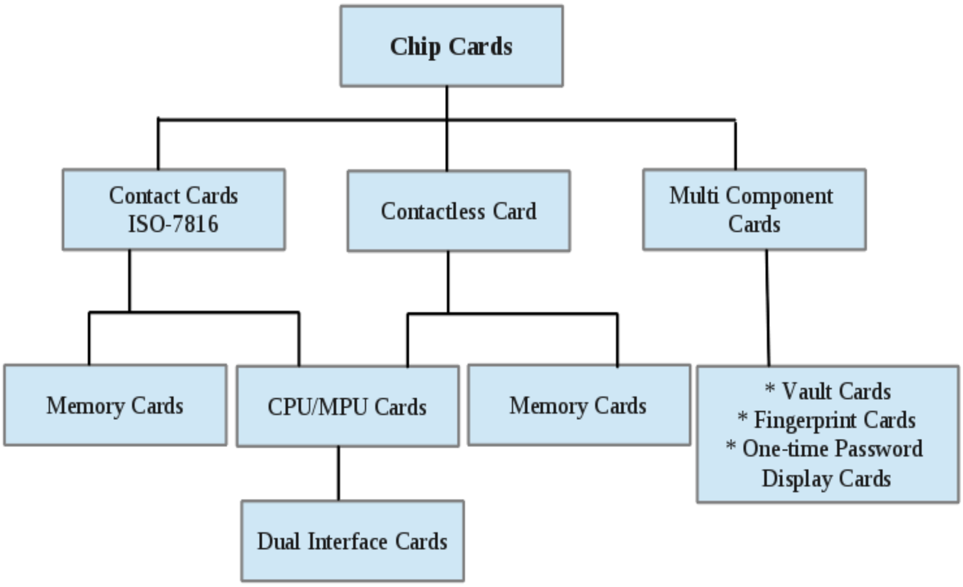


Figure 5.1. **Types of smart cards** [6]

allows that the data inside the memory can be locked with a PIN (Personal Identification Number). This number is usually composed of 3 to 8 digit numbers, which are written in a special file on the card. Depending on functionality, the memory smart cards can be divided into three subdivisions. In other words, there are three types of memory cards as follows:

1. **Straight Memory Cards:** Basically, these cards just have the function of storing data. It means this type of memory card does not have data processing capability. Due to this card being used for the GSM market, it is the cheapest card in the market. Another important fact of this type of card is that they cannot identify themselves to the reader. This means that the reader has to know what type of card is being inserted or approximated; they are easily duplicated.

2. **Protected / Segmented smart cards:** This type of card sometimes is referred to as an intelligent card. It allows to control the writing and reading of the data stored in the memory. This property is usually done through a password or a PIN. As its name suggests, segmented smart cards can be divided into logical sections to make more than one function. An important fact is that this type of card is not easily duplicated.

3. **Stored Value Memory Cards:** These cards are basically designed for tokens

or to store values. In other words, these cards are rechargeable and most of them have an incorporated security system. For example, in applications such as phone cards, the card has 12 or 60 memory cells which are used to store phone numbers.

5.3 Microprocessor Smart Cards

As its name suggests, a microprocessor smart card is a card with a microprocessor. This card also contains a non-volatile memory. In other words, a microprocessor smart card is a small and tiny computer implemented in a plastic credit card-size. The microprocessor inside the card has the function of managing memory allocation and file access. It resembles the processor inside all personal computers which manages data in organized file structures via an operating system. In this case, the operating system will be a card operating system. The operating system in the smart card different from personal computers permits different and multiple functions. This means the smart card can be used in a variety of applications such as debit cards, building access, public transportation and IDs. There are many configurations of microprocessor smart cards, including PKI (Public key Infrastructure), math co-processors or JavaCards with virtual machine hardware blocks.

In summation, microprocessor smart cards are like computers, they have RAM, ROM and EEPROM with an 8 or 16 bits microprocessor. ROM stores the operating system which has the function of managing the file system in EEPROM. In this thesis work, this type of smart card will be used, taking advantages of its computation properties and memory capacity.

5.3.1 Card Operating Systems COP

As it was told previously, smart cards resemble to a personal computer; for that reason, they need operating systems. Nowadays, there are several well-known operating systems for smart cards such as:

1. **JavaCard OS:** It was developed by Sun Microsystems. One of the main advantages of this operating system is its independence to the programmers over architecture. Also, Java OS based applications could be implemented on any vendor of smart cards that support JavaCard OS.

2. **MULTOS:** It is the acronyms for Multi-application Operating System. As its name suggests, MULTOS supports more than one application. Basically, this operating system was created for high-security needs.

3. **Windows:** This operating system is one of the newest members of the Windows operating system family. Windows smart card is basically a microcomputer without a graphical user interface.

However, most of the smart cards use their own operating system. The vendors design application that go beyond the simple ISO-7816 standard.

The ISO-7816 standard is a multi-part international standard. The part 1, 2, and 3 deal only with contact smart cards and define the various aspects of the card and its interfaces, including the card's physical dimensions, the electrical interface and the communications protocols. The parts 4, 5, 6, 8, 9, 11, 13, and 15 are related to all types of smart cards (contact and contactless). They specified logical structure, different programming commands, application management, biometric verification, cryptographic services and application naming [6].

5.3.2 Communication with the smart card

As it was told previously, the protocol that specifies the communication and other characteristics of the smart card is the ISO-7816. Basically, the communication between smart card reader and smart card has three layers. On the top layer, the communication takes place for applications in smart card and an external device. In other words, the commands in the top layer have meaning only for a particular application.

The second layer is the layer of the APDU (Application Protocol Data Units). In this layer, commands APDU are independent of the application commands, but they are related to a specific application. In other words, APDU commands are application-specific meanings.

In the third layer there are two protocols with name such as T=0 and T=1. These protocols are the most-used variants of half-duplex asynchronous protocols defined in ISO-7816-3. Basically, the difference between T=0 and T=1 is that in T=0 each character is transmitted separately, and T=1 the transmission is made by blocks of characters [9].

Chapter 6

Authentication Schemes

6.1 Introduction

In this chapter some authentication schemes written in C language will be presented; also the working conditions in which they operate, their main blocks or components, and the function in each scheme will be described. For each developed authentication scheme, how using a combination of clock controlled LFSR, an EC signature algorithm, smart card and other blocks operate as a powerful authentication scheme for an application and use of remote controls will be discussed. At the same time, the constraints and cost limitations for each working mode or authentication scheme will be covered. On the other hand, the simulations are divided into two parts, one of them has the function of describing all components of the cryptography channel and error conditions, and the second part is addressed to describe how it operates and the performance for each authentication scheme.

6.2 General description

In general words, the authentication schemes developed in this study can be used in any signature system or application of reduced resources. However, these schemes were created in order to avoid two main attacks in a system composed by a pool of remote control and base station. Mainly, the attacks are:

- **Attack 1:** Any attack or intention to copy the signal transmitted by the remote control.
- **Attack 2:** Clone the remote control and put it in the pool.

In order to avoid these attacks, three main authentications schemes were developed. First, it is important to emphasize that in order to show how the authentications schemes operate, a possible scenario will be showed; in this scenario exists a pool of remote controls,a base station and a third person who has the intention of attack the system. It means, cloning one remote control or copying and replying the signal sent by one remote control.

In this general scenario, the base station has the function to receive the signal sent by the remote controls, verify its authenticity and allow or denied an action controlled by the base station. Regarding to the remote controls, they must be work without any manufacturer intervention after they are constructed. It is important to know that the complete system (remote controls and base station) is exposed or in disposition of a third person who wants to clone in any way to clone it.

6.3 Authentication scheme 1

In this thesis, this is considered the most powerful scheme because of its components and operating mode. In order to avoid any type of attack this scheme was proposed. In this scheme, the remote control and base station are basically composed by:

- Smart card
- Transmitter and receiver device
- Microprocessor

In this working mode or scheme, the fact that there is a receiver and transmitter in the remote control, that made the system immune to any attack; allowing the implementation of the challenge authentication. This means, each time the remote control wants to access to the resource controlled by the base station (open a door, allow to access to a file, etc), it must have to answer a challenge sent by the base station. The figure 6.1 shows how works this authentication scheme in this scenario.

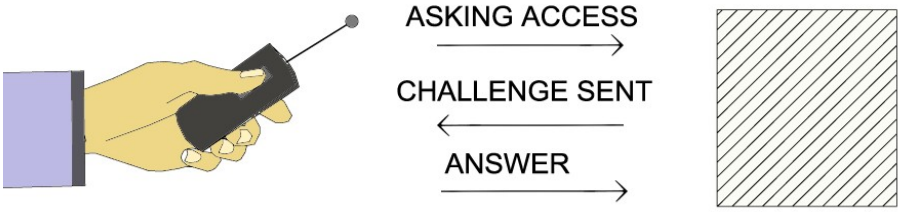


Figure 6.1. **Challenge protocol**

In the figure 6.2, it is supposed that the user operates the remote control in the range of base station. In other words, the remote control is into the distance it can covered with the available energy.

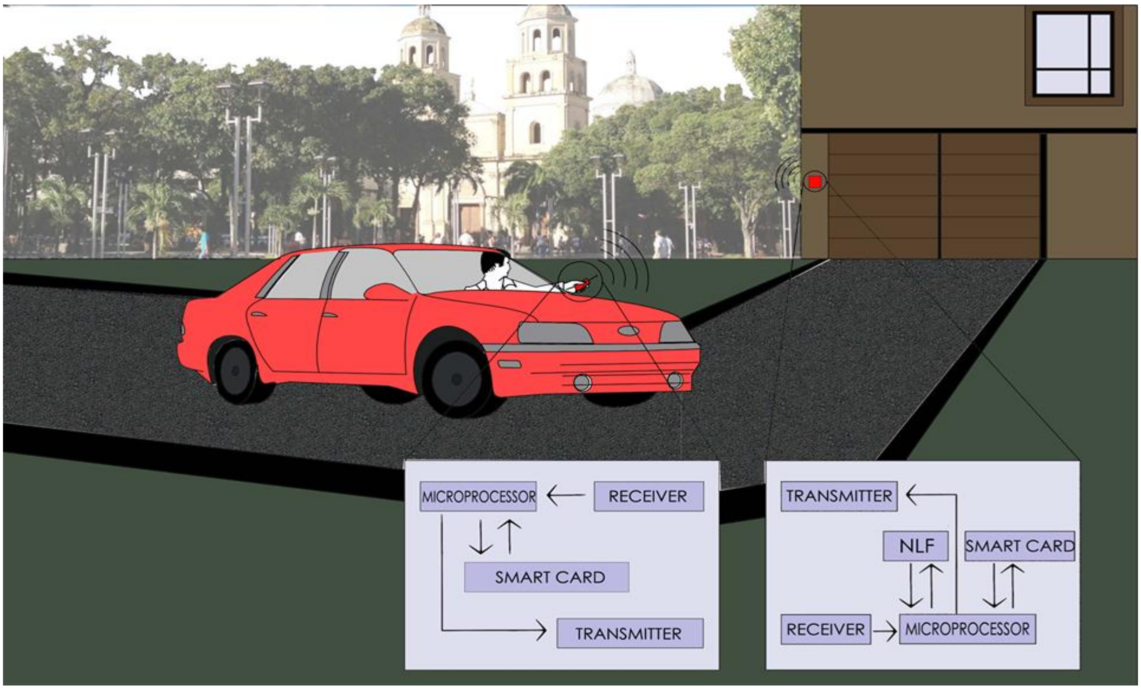


Figure 6.2. **Authentication scheme 1**

6.3.1 Composition blocks

In this section, each block in the remote control and base station will be described. How they are composed and their configuration.

Remote control

Remote control is composed by a source of energy (battery), a microprocessor, a smart card, a transmitter and receiver.

- **Receiver:** The receiver has the easy function of receive signals sent by the base station. Each time the remote control ask access for the resource controlled by the base station, this will be send a challenge to the remote control that it will be computed with parameters saved in the base station.

- **Transmitter:** As its name suggests, the transmitter is responsible to transmit the computed signal in the remote control when the user want to access to the resource or when the remote control has the answer to the challenge sent by the base station.

- **Smart card:** Basically, the smart card has as input part of the sequence sent by the base station; this part is the random sequence calculate in the base station. With this sequence, the smart card takes 10 bits that are in different positions in the sequence. Those positions are secret. Therefore, these 10 bits form the multiplier k. k is the value which multiply the initial point of the Elliptic curve. It is important to say that the initial point in the Elliptic curve are composed in projective coordinates or inversion-free coordinate system. Which means that is fast to do operations.

  After that, the result of multiply k by the initial point is another point in the binary field with components in X, Y and Z. Having these coordinates, and secret number saved in the smart card, it is computed the output with a simple operation.

  In the smart card:

  Input: Sequence sent by the base station (70 bits)

  Output: Elliptic curve signature computations (60 bits)

  Operations:

  1. Takes 10 bits from the stream sequence of the input that composed the number k
  2. Calculates k times initial point: $kP_0 = (X_k,Y_k,Z_k)$
  3. Calculates the output with a simple operation such as:

  $$Output = X_k + 2Y_k - Z_k + \text{secret number} \tag{6.1}$$

  The secret number, the initial point and the steps to choose the number k are memorized and protected by the anti-tamper property of the smart card.

  In this smart card are implemented an Elliptic curve signature with the next properties:

  - **Initial point in polynomial representation:**

    $$P_0 = x^{59} + x^7 + x^4 + x^2 + 1 \tag{6.2}$$

    The Galois field is $GF(2^{59})$, and composed by $4 * 14321189 * 10063074221$ elements.

  In order to make the computations faster inside the smart card, the points $2P_0$, $4P_0$, $8P_0, \ldots, 2^9P_0$ may be precomputed and stored on the smart card. Therefore, at most 9 point addition are required; knowing that k is composed by 10 bits.

- **Microprocessor:** This is a very important part of the remote control in all authentication schemes. The microprocessor is whose composes the final sequence before it is sent to the transmitter and it is whose has direct communication with the receiver and transmitter . When the button is pressed, the microprocessor generates 13 bits of the Barker code and send them together with remote control ID. Also, the microprocessor is whose does the function of correlator. It means, when the base station sends the challenge to the remote control, the receiver detects the signal and the microprocessor correlate that signal searching initial 13 bits related to the Barker code. After the microprocessor detect this 13 bits, the next 70 bits will be the input of the smart card. When the smart card finishes its calculations, the microprocessor is whose composed the final stream sequence that will be send it to the transmitter as answer of the challenge. All this procedure will be better explained in the next subsection 'How it operates'.

- **Battery:** It has the important job to energize the entire system of the remote control. The power of the battery will be depend on the average energy consume by the microprocessor, smart card, transmitter and receiver.

Base station

The base station for this authentication scheme is composed by a smart card, a transmitter, a receiver, a block called NLF (Non Linear Function), a microprocessor and obviously a source of energy (battery).

- **Receiver:** As its name suggests, it receives the signals sent by the remote control. Each time the remote asks for the access to the resource managed by the base station, the receiver is whose first detect the signal.

- **Transmitter:** Is the block whose has the job to transmit the challenge made by the base station and sent to the remote control.

- **Smart card:** The smart in the base station has the same properties of the smart card in the remote control. Its input will be a sequence of 70 bits and its output a sequence of 60 bits. When the smart card receives the input, it takes 10 bits of them that will compose the number k, and then it will be possible to compute the output doing the multiplication of the number k with the initial point of the Elliptic curve P_0.

- **NLF (Non Linear Function):** The block NLF is controlled by the microprocessor, it is basically a combination of Linear Feedback Shift Registers, or better, clock controlled LFSR. As it is shown in the next figure:

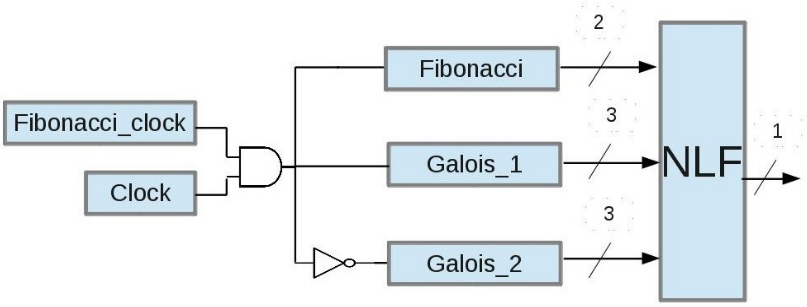


Figure 6.3. **clock controlled LFSR, NLF**

where:

Fibonacci_clock and Fibonacci are defined by:

$$x^{15} + x^{13} + x^{11} + x^9 + x^8 + x^7 + x^4 + x^3 + 1 \tag{6.3}$$

Galois_1 is defined by:

$$x^{15} + x^{12} + x^{11} + x^8 + x^5 + x^2 + 1 \tag{6.4}$$

Galois_2 is defined by:

$$x^{15} + x^{14} + x^{12} + x^3 + 1 \tag{6.5}$$

Basically, the output of the 'Fibonacci_clock' LFSR is ANDed with the clock. Therefore, when there is clock and the output of 'Fibonacci_clock' is '1', the 'Fibonacci' and 'Galois_1' LFSRs are clocked and the 'Galois_2' LFSR is not. Instead, when the output of 'Fibonacci_clock' is '0', the 'Galois_2' LFSR is clocked and the 'Fibonacci' and 'Galois_1' LFSRs are not clocked. Then, each clock pulse, the NLF block takes three bits from the 'Galois_1', three bits from the 'Galois_2', and 2 bits from 'Fibonacci' LFSR calculating the next function:

$$x_1 z_5 y_{14} + y_1 x_{14} + z_1 y_5 + x_{14} \tag{6.6}$$

Where the variables x_i are taken from the 'Fibonacci' LFSR, the variables y_i are taken from from the 'Galois_1' LFSR and the variables z_i are taken from 'Galois_2' LFSR. The NLF generates one bit for each pulse and 70 bits each time the remote control ask for access. These 70 bits are sent together with the Barker code.

- **Microprocessor:** As the microprocessor in the remote control, it does the same function as correlator. When the remote control answer the challenge, the microprocessor correlates the receive signal with the Barker code in order to 'search' the Elliptic curve signature output sent by the remote control. Another important task made by microprocessor in the base station is to control the NLF in order that it generates the 70 bits corresponding to the challenge that will be sent to the remote control. Again, this procedure will be better explained in the next subsection 'How it operates'.
- **Battery:** As in the remote control, it has the important job to energize the entire system of the base station. The power of the battery will be depend on the average energy consume by the microprocessor, smart card, transmitter, receiver and NLF.

In this scheme, it is used a NLF block. This block can be made like a independent block or simply as instructions made inside the microprocessor.

6.3.2 How it operates

Described all the main parts of the remote control and base station, this section will discuss the working mode of the authentication scheme 1. Some details of this procedure was given above, however in this section it will summarize the entire modus operandi of this scheme.

1. The user presses the button in the remote control, which makes the microprocessor generates the Barker code and together with the remote control ID sends the sequence though the transmitter.

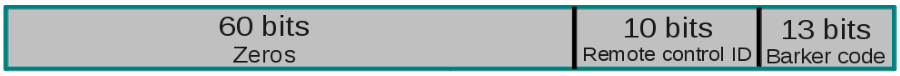


Figure 6.4. **Asking access sequence**

2. Supposing that base station is in the range covered by the remote control, it receives the signal through the receiver. After that, the microprocessor in the base station correlate the received signal searching the Barker code (13 bits). When it is founded, the microprocessor enables the NLF to generate a random sequence composed by 70 bits. In this case, all LFSRs have same initial value and it is changed each time the remote control ask for access. Then, microprocessor sends to transmitter the random sequence together with the Barker code, and then, transmitter sends to the remote control. The

random sequence generated in the NLF is also sent to the smart card in the base station.

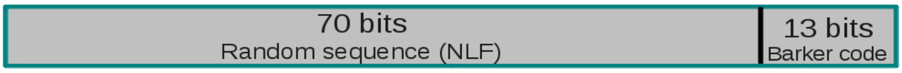


Figure 6.5. **Challenge sequence**

3. After base station sends the challenge to the remote control, it receives the signal through the receiver. Then, microprocessor correlates the signal searching the Barker code. After that, the next 70 bits in the sequence are sent as input to the smart card in the remote control. The smart card takes 10 of the 70 bits composing the factor k. Inside the smart is calculated kP_0. it means, k times the initial point in the Elliptic curve, and finally, it computes the output with the x, y, z components of the result point and the secret number. Therefore, microprocessor sends to the transmitter the output of the smart card together with Barker code and remote control ID. In the end, the transmitter transmits the answer of the challenge sent by the base station.

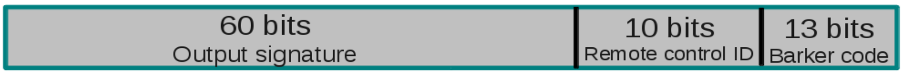


Figure 6.6. **Answer to the challenge**

4. Finally, base station receives through the receiver the sequence sent by the remote control. First, microprocessor puts the same 70 bits generated by NLF and sent to the remote control as input to the smart card in the base station. The smart card does the same procedure as in the remote control; it takes 10 of the 70 bits, it composes the factor k and calculates the output with the coordinates of the result point kP_0 and the secret number.

   The output of the smart card is correlated with the output sent by the remote control as answer of the challenge. Then, if they are the same, the resource will be enabled, if they are not, the resource will not. In the example case, the resource can be a door. It means, if the sequences are the same, the door will be opened, if not, the door will not open.

6.3.3 Results

In order to measure the performance of this scheme, it was developed an algorithm written in C language that simulates the whole system, including the Elliptic curve signature inside the smart card, receivers, transmitters, NLF and the function of the microprocessor (correlator and logically connection). The procedure takes 1 second and 190 milliseconds to do 10.000 authentications. It means, 0.119 milliseconds for one authentication.

It is important to say that this test was made with a computer with high performance. Which means that the time will change depending on the implemented microprocessor on the remote control and base station.

Another important fact to say about this authentication scheme is that depending on the time taken by smart card to compute Elliptic curve signature; it can be replaced by the second version on the scheme. This second version consists on implementation of a clock controlled LFSR in the smart card. It means, instead of having an Elliptic curve signature it will have a clock controlled LFSR. Basically, the system will work with same procedure but it changes the way the smart card calculated the answer of the challenge sent by the base station.

Smart card will have the same 60 bits output and the same 70 bits input. However the way it calculates the output will change. The smart card takes 16 of the 70 bits in the input. These 16 bits will be the initial value for all LFSRs in the clock controlled LFSR. After that, the NLF inside the smart card will generate 60 bits, and send as output of the smart card.

The performance of the system with this smart card improves notoriously in time. The system with a high performance processor takes 140 milliseconds to do 10.000 authentications. It means, 0.014 milliseconds for one authentication.

6.4 Authentication scheme 2

As in the authentication scheme 1, the main goal in the scheme 2 is to define a mathematical function that will be implemented in the smart card. This function must be secret and impossible to deduce from the input and output of the smart card. This function also must not require to much calculation because of the constraints imposed for the system and in order to save the required execution time for the application of this authentication scheme. Then, the remote control and base station are composed for these blocks:

- Smart card
- Transmitter device
- Flash memory
- Microprocessor

Given there is no transmitter in base station, this scheme operates in different way. It means, it must have an authentication protocol different from scheme 1. The reason is because the transmitted signal is different each time the user pulses the button of the remote control; there is a count that increase by 1 in the remote control and base station. Then, it is possible that system lost synchronization and the base station denies the access.

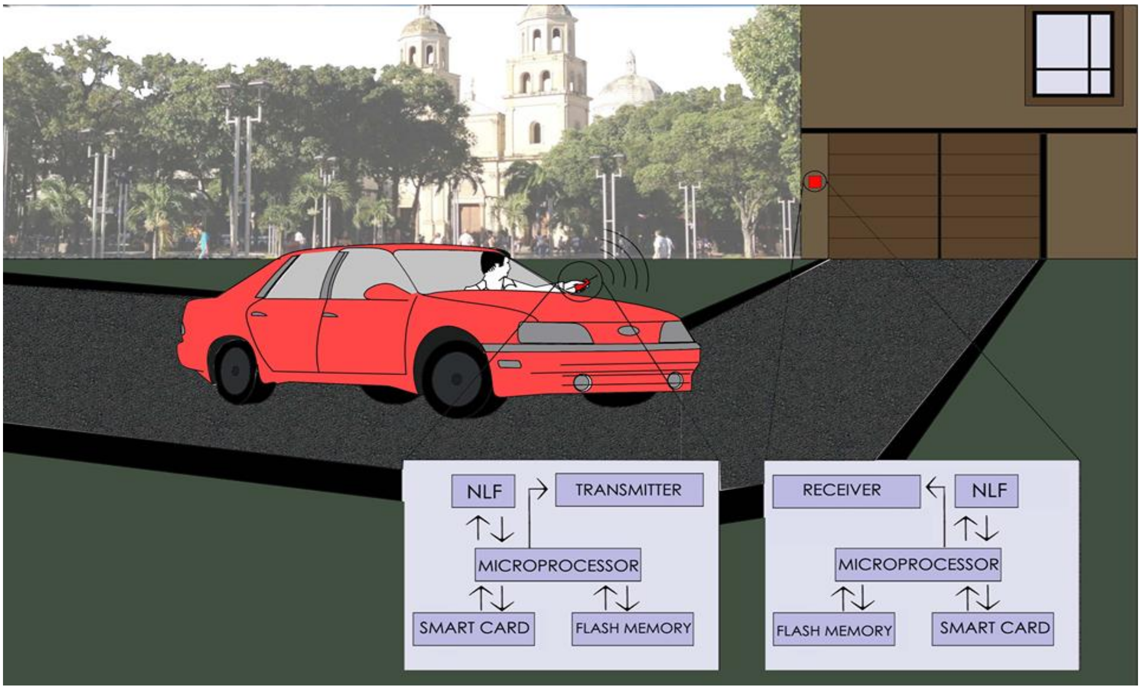


Figure 6.7. **Authentication scheme 2**

6.4.1 Composition blocks

In this section each part of the remote control and base station will be described, how they interact each other and their main tasks in the whole system.

Remote control

- **Transmitter:** As its name suggests, the transmitter is responsible to transmit the computed signal in the remote control when the user want to access to the resource.
- **Smart card:** In this scheme the smart card has the same function as the scheme 1. It means, inside the smart is implemented the same Elliptic curve

signature, where having an input of 50 bits the smart card takes 10 bits that are in different positions in the sequence. Those positions are secret. Therefore, these 10 bits form the multiplier k. k is the value which multiply the initial point of the Elliptic curve. It is important to say that the initial point in the Elliptic curve are composed in projective coordinates or inversion-free coordinate system. Which means that is fast to do operations.

After that, the result of multiply k by the initial point is another point in the binary field with components in X, Y and Z. Having these coordinates, and secret number saved in the smart card, it is computed the output with a simple operation.

In the smart card:

Input: Sequence sent by the base station

Output: Elliptic curve signature computations

Operations:

Summarizing, the smart card:

1. Takes 10 bits from the stream sequence of the input that composed the number k
2. Calculates k times initial point: $kP_0 = (X_k, Y_k, Z_k)$
3. Calculates the output with a simple operation such as:

$$\mathit{Output} = X_k + 2Y_k - Z_k + \text{secret number} \tag{6.7}$$

Finally, the output is a sequence composed by 60 bits.

The secret number, the initial point and the steps to choose the number k are memorized and protected by the anti-tamper property of the smart card.

- **Flash memory:** Given the remote control has not receiver, the flash memory is an important block responsible of the synchronization in the system. Each time the user pulses the button of the remote control and it is out of the range, the system will lost synchronization. In order to avoid the system starts always at the same point, in the flash memory is memorized the last state or reached state when the system was synchronized. The reached state means the reached values of the clock controlled LFSR and the number of keystrokes made before the system lost synchronization. In the section 'How it operates' this procedure will be better illustrated.

- **Microprocessor:** In the remote control the microprocessor does the main tasks. First, it does the function of the 'clock controlled LFSR and NLF' or it controls the block 'clock controlled LFSR and NLF' in case this will be a different block like the scheme 1. In any case, this block generates a random sequence of 50 bits with the initial value saved in the flash memory. Secondly, the microprocessor puts the 50 bits sequence as input of the smart card. Thirdly, it makes the final sequence composed by 173 bits, which has the first 13 bits of Barker code, 10 bits of remote control ID, 40 bits of the number of keystrokes, 50 bits of the random sequence generated by the NLF and the 60 bits output of the smart card. Finally, it sends through the transmitter the final sequence and update the memorized values in the flash memory (number of keystrokes and LFRS's values).

Obviously, the remote control needs a source of energy. Depending on what type of microprocessor and the energy consumption of all blocks in the remote control, the power of the battery will be easily calculated.

Base station

The base station in this authentication scheme is composed by a smart card, a receiver device, a microprocessor and a flash memory.

- **Receiver:** As its name suggests, this block is the responsible of receiving signal from the remote control. Each time the remote requests access, this block receives the authentication sequence to be processed inside the base station.

- **Smart card:** The smart card in base station and remote control does the same function. It has an input of 50 bits and an output of 60 bits. From the 50 bits input, the smart card takes 10 bits being the k factor. This number multiply the initial point P_0 generating three coordinates X, Y and Z of the result point kP_0. Then, it is implemented a basic function as next:

  $$Output = X_k + 2Y_k - Z_k + \text{secret number} \tag{6.8}$$

  The secret number could be the builder identifier. Finally, the output is a sequence composed by 60 bits.

  It is important to highlight that due to the 50 bit sequence in the input of the smart card is always different, the output is also different. Then, it avoids store-and-reply attacks.

- **Flash memory:** The base station as the remote control, it is composed by a flash memory. The main function of this, is to save the reached values for the clock controlled LFSR and the number of keystrokes for each remote control. In this case, flash memory will be memorized these values for all the pool of remote controls. Each remote control in the pool is identified by a serial number, this number does not allows to memorize into the pool more than one remote control with the same serial number. Another important fact in the flash memory is when there is a successful authentication by one remote control; it will be updated the number of keystrokes and the value reached by the clock controlled LFSR associated to the remote control. As it was told before, in the section remote control, the flash memory does, in indirect way, the function of the missing receiver in the remote control. It means, the flash memory allows the resynchronization procedure. This resynchronization procedure considers the idea of transmitting signals always different.

- **Microprocessor:** Generally speaking, the main goals of the microprocessor are to correlate the received sequence with a sequence created inside the base station, and the resynchronization procedure; in case the system needs it. In this scheme, the microprocessor needs to execute more operations, due to the fact this scheme must perform operations to keep the system synchronize. Next, it will be described in detail the operations and function of the microprocessor in the base station and remote control.

  It is obvious that also in the base station it is needed a battery that supply energy to all the components.

6.4.2 How it operates

This section is dedicated to described the modus operandi of the authentication scheme 2. Starting from the user presses the button in the remote control until base station allows access to the resource controlled by it. Supposing the example of using this scheme implemented in a system where the base station controls the access to a door.

1. The user presses the button in the remote control. Then, microprocessor in remote control enables the clock to the 'clock controlled LFSR' block, or in any case, it generates a random sequence composed by 50 bits with initial read values from the flash memory. Generated this sequence, microprocessor puts it as input to the smart card. Then, the smart card generates its output composed by 60 bits. How the smart card generates its output was described before. The next step of microprocessor is to compose the final sequence ready to be sent through the transmitter. It sends through the transmitter together

the Barker code, the remote control ID, the read number of keystrokes from the flash memory, the random sequence generated by 'clock controlled LFSR' and the output of the smart card. Finally, and after the signal is transmitted, the microprocessor updates the values:

- Number of keystrokes.
- Reached values by the 'clock controlled LFSR'.

in the flash memory.

The transmitted sequence by remote control is shown in next figure:

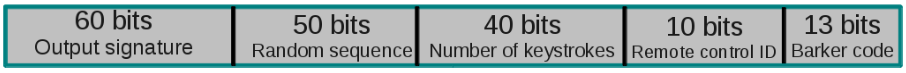


Figure 6.8. **Asking access sequence**

2. Supposing the remote control was operated when it was close to a base station and it receives the signal. The microprocessor, in base station, correlates the received signal searching the Barker code. If it is founded, microprocessor reads the remote control ID in the received signal in order to search the associated values saved in flash memory for that remote control. The next step of the microprocessor is to generate a random sequence of 50 bits with the 'clock controlled LFSR' putting as initial value, the read value in flash memory. Later, microprocessor compares the random sequence generated in base station with the random received sequence; if they are the same, the next step is to compare the output of the smart card in base station with the output received from the remote control. It means, in case of the random sequences are equal (received sequence and composed sequence by base station), microprocessor in base station puts the 50 bits as input of the smart card, and then, smart card generates an output of 60 bits. With this output, it is compared the last 60 bits of the received sequence. If the random sequences and the outputs of the smart card are the same, base station allows the access to the remote control, and updates the values for that remote control (number of keystrokes and reached values by the 'clock controlled LFSR'), in the flash memory. But, what happens when the random sequence are not the same?.

   In case the received random sequence and composed random sequence by base station are not the same, microprocessor reads the number of keystrokes from the received signal as well as the number of keystrokes from the flash memory for the remote control that is asking access. Then, microprocessor compares those values; if the read value from the received signal is lower than the read value from the flash memory, the access is immediately denied. However, if the read value from the received signal is greater than the read value from the flash memory, microprocessor enables the clock to the 'clock controlled LFSR' generating 50 bits that corresponds to the number of keystrokes associated to the read number from the received signal. Then, the random sequence from the received signal is compared with the random sequence generated by the base station. In this case, if those sequences are the same, the next step will be to compare if the last 60 bits concerning to the output of smart card are the same. If they are the same, the access is allowed. However, if the random sequences are not the same, the outputs of the smart card is not compare and the access is denied.

Reviewing, there are two main cases, the first case is when base station and remote control are synchronize, and the second case is when they are not synchronize. When they are synchronize, it implies that the number of keystrokes are the same, and then, random sequences are also the same.

The second case is when they are not synchronize. It implies that the number of keystrokes and random sequences are not the same, but the registered number of keystrokes in the remote control is greater than the number of keystrokes in base station. In this case, microprocessor tries to resynchronize the system by shifting the values of the 'clock controlled LFSR'.

6.4.3 Results

As scheme 1, it was developed an algorithm written in C language that simulates the whole system, including Elliptic curve signature inside the smart card, transmitter in remote control, receiver in base station, flash memories and the function of the microprocessor (correlator, clock controlled LFSR, NLF and logically connection). The procedure takes 1 second and 210 milliseconds to do 10.000 authentications. It means, 0.121 milliseconds for one authentication, when the system is synchronized. When the system is not synchronized, it takes 1 second and 270 milliseconds to do 10.000 authentications. It will be 0.127 milliseconds for one authentication.

As in scheme 1, this test was made with a computer with high performance. Which means that the time will change depending on the implemented microprocessor on the remote control and base station.

As in scheme 1, in this scheme is used a NLF block. This block can be made like a independent block or simply as instructions made inside the microprocessor, depending on what way has better performance.

There is another possible version of this scheme. Instead of implementing an Elliptic curve signature in the smart card, it is possible to implement a clock controlled LFSR with NLF (Non Linear Function). It can reduce significantly the require time for each authentication. However, it is not secure as when it is used an Elliptic curve signature.

6.5 Authentication scheme 3

Basically, the main difference between this scheme and the previous one is the use of smart card or anti-tamper system. This means, this scheme does not use smart card. The Elliptic curve signature or the function implemented in the smart card in the previous scheme is implemented as instructions on the microprocessor in this scheme. Therefore, the remote control and base station are composed for the next blocks:

- Transmitter device
- Flash memory
- Microprocessor

As in scheme 2, given the fact there is no transmitter in base station, it makes use of the flash memory to keep the synchronization. Using flash memory, it is avoided that the system starts always at the same point each time it loses synchronization.

6.5.1 Composition blocks

In this section each part of the remote control and base station will be described, how they interact each other and their main tasks in the whole system.

Remote control

- **Transmitter:** As its name suggests, the transmitter is responsible to transmit the computed signal in the remote control when the user want to access to the resource.
- **Flash memory:** In order to avoid the system starts always at the same point, in the flash memory is memorized the last state or reached state when the system was synchronized. The reached state means the reached values of the clock controlled LFSR and the number of keystrokes made before the system lost synchronization.
- **Microprocessor:** In this scheme, remote control microprocessor does all calculations and operations. First, microprocessor:

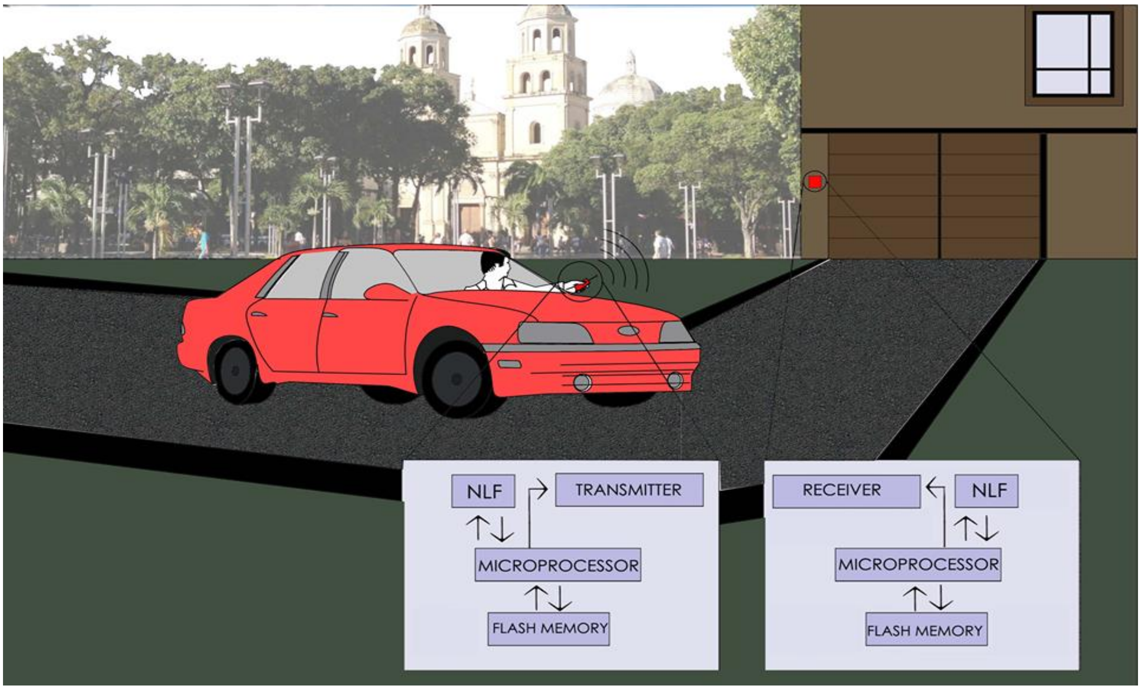


Figure 6.9. **Authentication scheme 3**

1. Generates 10 bits using a clock controlled LFSR. These bits compose the number k
2. Calculates k times initial point: $kP_0 = (X_k, Y_k, Z_k)$
3. Calculates a simple operation such as:

$$Signature = X_k + 2Y_k - Z_k + \text{secret number} \tag{6.9}$$

Finally, the sequence is composed by 60 bits.

As all the schemes, the secret number or builder number, the initial point and the structure of the clock controlled LFSR are private information.

Secondly, microprocessor makes the final sequence composed by 123 bits, which has the first 13 bits of Barker code, 10 bits of remote control ID, 60 bits of signature procedure and 40 bits of the number of keystrokes. Finally, it sends through the transmitter the final sequence and update the memorized values in the flash memory (number of keystrokes and LFRS's values).

Obviously, the remote control needs a source of energy. Depending on what type of microprocessor and the energy consumption of all blocks in the remote control, the power of the battery will be easily calculated.

Base station

The base station in this authentication scheme is basically composed by a receiver device, a microprocessor and a flash memory.

- **Receiver:** As its name suggests, this block is the responsible of receiving signal from the remote control. Each time the remote requests access, this block receives the authentication sequence to be processed inside the base station.

- **Flash memory:** Also in the base station there must be a flash memory. The main function of this, is to save the reached values for the clock controlled LFSR and the number of keystrokes for each remote control. As in scheme 2, flash memory will be memorized these values for all the pool of remote controls. Each remote control in the pool is identified by a serial number; this number does not allows to memorize into the pool more than one remote control with the same serial number.

- **Microprocessor:** Fundamentally, functions of the microprocessor are to correlate the received sequence with a sequence created inside the base station, and to execute the resynchronization procedure; in case the system needs it. In this scheme, different from the previous one, the microprocessor needs to execute more operations, due to the fact there is no smart card.

  It is obvious that also in the base station it is needed a battery that supply energy to all the components.

6.5.2 How it operates

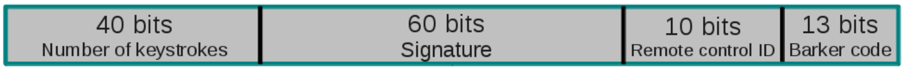
This section is dedicated to described the modus operandi of the authentication scheme 3. Starting from the user presses the button in the remote control until base station allows access to the resource controlled by it. Supposing the example of using this scheme implemented in a system where the base station controls the access to a door.

1. The user presses the button in the remote control. Then, microprocessor in remote control, using a 'clock controlled LFSR' structure and initial values from the flash memory, generates 10 bits composing the number k. After it generates these bits, microprocessor multiply k times the initial point P_0. After this operation, microprocessor executes a simple operation such as:

$$Signature = X_k + 2Y_k - Z_k + \text{secret number} \tag{6.10}$$

Finally, microprocessor composes the final sequence to be sent through the transmitter. This sequence is composed by the Barker code, the remote control ID, Elliptic curve signature bits and the read number of keystrokes from the flash memory.

The transmitted sequence by remote control is shown in the next figure:



40 bits Number of keystrokes	60 bits Signature	10 bits Remote control ID	13 bits Barker code

Figure 6.10. **Asking access sequence**

After that, the microprocessor updates the values:

- Number of keystrokes.
- Reached values by the 'clock controlled LFSR'.

in the flash memory.

2. Supposing the remote control was operated when it was close to a base station and it receives the signal. The microprocessor, in base station, correlates the received signal searching the Barker code. If it is founded, microprocessor reads the remote control ID in the received signal in order to search the associated values saved in flash memory for that remote control. The next step of the microprocessor is to generate a sequence of 10 bits with the 'clock controlled LFSR' putting as initial value, the read value in flash memory. These bits conformed the number k. Later, microprocessor computes kP_0 and the simple function with the result coordinates X, Y and Z. As next:

$$Signature = X_k + 2Y_k - Z_k + \text{secret number} \qquad (6.11)$$

After that, microprocessor compares the signature generated in base station with the received signature. If they are the same, base station allows the access to the remote control, and updates the values for that remote control (number of keystrokes and reached values by the 'clock controlled LFSR'), in the flash memory. But, what happens when the signatures are not the same?.

In case the signatures are not the same, microprocessor reads the number of keystrokes from the received signal as well as the number of keystrokes from the flash memory for the remote control that is asking access. Then, microprocessor does as in scheme 2, it compares those values; if the read value from the received signal is lower than the read value from the flash memory, the access is immediately denied. However, if the read value from the received signal is greater than the read value from the flash memory, microprocessor generates 10 bits shifting the values of the 'clock controlled LFSR' until the corresponding number of keystrokes of the received signal. With these 10 bits, microprocessor multiply kP_0 and calculate the basic equation (previous equation). Finally, if the signatures are the same, the access is allowed; else the access is denied.

Reviewing, there are two main cases, the first case is when base station and remote control are synchronize, and the second case is when they are not synchronize. When they are synchronize, it implies that the number of keystrokes are the same, and then, generated signatures are also the same.

The second case is when they are not synchronize. It implies that the number of keystrokes and signatures do not match, but the registered number of keystrokes in the remote control is greater than the number of keystrokes in base station. In this case, microprocessor tries to resynchronize the system by shifting the values of the 'clock controlled LFSR'.

6.5.3 Results

As scheme 1 and 2, it was developed an algorithm written in C language that simulates the whole system, including Elliptic curve signature, transmitter in remote control, receiver in base station, flash memories and other functions of the microprocessor (correlator, clock controlled LFSR, NLF and logically connection). The procedure takes 1 second and 340 milliseconds to do 10.000 authentications. It means, 0.134 milliseconds for one authentication, when the system is synchronized. When the system is not synchronized, it takes 1 second and 350 milliseconds to do 10.000 authentications. It will be 0.135 milliseconds for one authentication.

As in scheme 1 and 2, this test was made with a computer with high performance. Which means that the time will change depending on the implemented microprocessor on the remote control and base station.

There is another possible version of this scheme. Instead of implementing an Elliptic curve signature, it is possible to replace it with a clock controlled LFSR with NLF (Non Linear Function). It can reduce significantly the require time for each authentication. However, it is not secure as when it is used an Elliptic curve signature.

Instead of using Elliptic curve signature, it was used a structure of a clock controlled LFSR and it was simulated using the same computer and the results were 100 milliseconds for 10.000 authentications. It means, 0.001 milliseconds for one authentication, when the system is synchronized. When the system is not synchronized, it takes 120 milliseconds to do 10.000 authentications. It will be 0.015 milliseconds for one authentication.

Conclusions and Future Work

- It was developed three authentication schemes written in C and using Elliptic curve signature with inversion-free coordinate system. This implies very fast operations without precomputations and a system with fair balance between processing capacity and needed resource to be implemented.

- Based of the wide diffuse of the smart card and taking advantage of their anti-tamper property. Two of the three schemes were developed using it. Fundamentally, the difficulty to be copy any of these schemes is based on the anti-tamper property of the smart card. It means, because of the facts that all information about the Elliptic curve signature and the configuration of the clock controlled LFSR is private; also the transmitted signals are always different, there is no way that the authentication protocol can be copied unless the information of the smart card will be revealed.

- In the presented results and for each authentication scheme, it was not taken into account the needed time to read data from the flash memory. It obviously depends on the type of memory and used microprocessor. For that reason, the real time for each authentication will know when it decides to implement the schemes.

- The implementation of the scheme 2 is probably cheaper than the scheme 1, because of the fact there is no receiver device or there is not two-way communication. Also, this scheme does not offer unconditional protection to a possible attacks made by a third part. However, it is too difficult to make a clone of the system.

- The implementation of the scheme 3 is probably cheaper than the scheme 2 and scheme 1, because of the fact it does not make used of smart card. However, it is not secure as the scheme 1 or 2. This means, it can be in some ways easily cloned.

Annexes

Annex A

Finite Fields

A.1 Introduction

The motivation for writing about finite fields or Galois fields in this thesis work comes from the fact that the efficient implementation of finite field arithmetic is an important prerequisite in elliptic curve cryptosystems. This is because curve operations are performed using arithmetic operations in Galois fields.

A.2 Definition of Field

Finite fields are one of the essential parts in coding theory and cryptography. Coding theory has applications in error-free communications and data storage [14]. In this thesis work, finite fields are used to build several signature schemes in smart cards. Both coding theory and cryptography use algorithms based on arithmetic of elements over a field $GF(2^n)$ (This type of finite field will be explained later). Therefore, they appear in many areas in IT security. Fundamentally, a field is a set F (with at least two elements) together with two binary operations [32]:

- Addition, denoted by '+', and
- Multiplication, denoted by '.'

Those operations must satisfy the following rules, or field laws:

- F is closed under + and ., it means for all

$$x, y \in F,\ x + y \in F,\ x.y \in F$$

- Addition and multiplication are commutative. That means, for all

$$x,y \in F,\ x + y = y + x \text{ and } x.y = y.x$$

- Addition and multiplication are associative. That means, for all

$$x,y,z \in F,\ x + (y + z) = (x + y) + z \text{ and } x.(y.z) = (x.y).z$$

- There is an additive identity element. There must exist and element in F called 0 such that $x + 0 = x$ for all $x \in F$

- There is a multiplicative identity element. There must exist an element of F called 1 such that $1.x = x$ for all $x \in F$

- Additive inverses exist. It means, for all $x \in f$, there exists an element $-x \in F$ such that $x + (-x) = 0$

- Reciprocals exist. That means, for all $x \neq 0 \in F$, there must exist and element $x^{-1} \in F$ such that $x(x^{-1}) = 1$

- Distributivity. For all $x,y,z \in F,\ x(y + z) = x.y + x.z$

Based on those rules or field laws, many other interesting properties of fields can be deduced. These other properties are similar to the arithmetic properties of the real numbers, such as:

- The identity elements for addition and multiplication, and additive and multiplicative inverses are unique.

- For all $x \in F,\ 0.x = 0$

- For every $x \in F,\ -x = -1.x$

- For every $x,y,z \in F,\ (x + y).z = x.z + y.z$

Some well-known examples of fields are the field of rational numbers $\mathbb{Q}$, the field of real numbers $\mathbb{R}$, and the field of complex numbers $\mathbb{C}$. Obviously, there are other fields more exotic, perhaps more interesting, such as Galois fields.

A.3 Finite Field or Galois Field

As its name suggests, a finite field is a field that contains a finite number of elements, called its order. As with any field, a finite or Galois field is a set on which the operations of multiplication and addition have been defined as it was enunciated above. Finite fields only exist when the order or size is a prime power p^k where p is a prime number and k is a positive integer.

In every finite field there are important elements called primitive elements. These elements are generators of the multiplicative group of the field. In other words, the non-zero elements can be expressed as the powers of a single element (primitive element).

A.4 Classification of Finite or Galois Fields

Finite or Galois Fields are classified as follows [38]:

- The number of elements in a finite field is of the form p^n, where p is a prime number called the characteristic of the field, and n is a positive integer.
- For every prime number p and positive integer n, there exists a finite field with p^n elements.
- Any two finite fields with the same number are isomorphic. That means, under some renaming of elements of one of those, both its addition and multiplication tables become identical to the corresponding tables of the other one.

According with this classification, use of a naming scheme for finite fields that specifies only the order of the field is justified. That means, a notation for finite field is F_{p^n} or $GF(p^n)$, where the letters GF stand for 'Galois Field'. Based on this classification of finite fields, a prime power field with $p = 2$ is also called a binary field or characteristic-two finite field.

A.5 Representation of Finite Field Elements

There are several ways to represent elements in a finite field. They can be represented in standard basis, normal basis, dual basis, or power representation. In this thesis work, a power representation and standard or binary representation is used. For example, in power representation, let $\alpha \in GF(p^n)$. The minimum integer r such that $\alpha^r = 1$, is called the order of α. The maximum order of any element is $p^n - 1$, and an element with order $p^n - 1$ is called a primitive element. Any irreducible polynomial whose roots have order $p^n - 1$ is called a primitive polynomial. All the nonzero elements of $GF(p^n)$ can be represented by the powers of a primitive element α

$$1,\alpha,\alpha^2,\dots,\alpha^{p^n-2} \tag{A.1}$$

Just to show an example of element representation in a finite field, consider the following table which several different representations of the elements are considered. The columns are the power, polynomial representation, triples of polynomial representation coefficients (the vector representation), and the binary or standard representation corresponding to the vector representation.

Power	**Polynomial**	**Vector**	**Regular**
0	0	(0 0 0)	0
x^0	1	(0 0 1)	1
x^1	x	(0 1 0)	2
x^2	x^2	(1 0 0)	4
x^3	$x+1$	(0 1 1)	3
x^4	x^2+x	(1 1 0)	6
x^5	x^2+x+1	(1 1 1)	7
x^6	x^2+1	(1 0 1)	5

Table A.1. **Different representations of the elements in a Finite Field**

The set of polynomials in the second column is closed under addition and multiplication modulo x^3+x+1 and these operations on the set satisfy the laws explained before for any finite field. In particular, this field will be an extension field of degree 3 of $GF(2)$, written $GF(2^3)$, and the field $GF(2)$ is called the base field of $GF(2^3)$. Like it was said in other words before, a primitive element generates all elements in the set, represented by a power of it. For any prime or prime power q and any positive integer n, there exist a primitive irreducible polynomial of degree n over $GF(q)$.

In general, in a field for any element c of $GF(q)$, $c^q = c$, and for any nonzero element d of $GF(q)$, $d^{q-1} = 1$. There is a smallest positive integer n satisfying the sum condition:

$$\underbrace{e+e+e+e+\cdots+e}_{\text{n times}} = 0$$

for some element e in $GF(q)$. This number is called the field characteristic of the finite field $GF(q)$. The field characteristic is a prime number for every field, and it is true that

$$(x+y)^p = x^p + y^p$$

over a finite field with characteristic p. [43]

Example: Consider the finite field $GF(2^3)$ constructed from the primitive polynomial x^3+x+1. It is formed by the following elements:

$$0,1,x,x^2,x^3,x^4,x^5,x^6$$

That will be in power representation. In polynomial representation, the element x^3 will be:

$$\begin{aligned}&x^3+x+1=0\\&x^3=x+1\end{aligned}$$

And then:

$$0,1,x,x^2,(x^2+1),x^4,x^5,x^6$$

Same process for the next elements, so

$$\begin{aligned}&x^4=x(x^3)=x(x+1)=x^2+x\\&x^5=x^3+x^2=x^2+x+1\\&x^6=x(x^2+x+1)=x^3+x^2+x=x^2+1\\&x^7=x^3+x+1\end{aligned}$$

Finally,

Power	Polynomial	Vector
0	0	(0 0 0)
x^0	1	(1 0 0)
x^1	x	(0 1 0)
x^2	x^2	(0 0 1)
x^3	$x+1$	(1 1 0)
x^4	x^2+x	(0 1 1)
x^5	x^2+x+1	(1 1 1)
x^6	x^2+1	(1 0 1)

Table A.2. **Elements of Finite Field $GF(2^3)$ in different representations**

In this work the field of interest will be $GF(2^{59})$ constructed from:

$$x^{59} + x^7 + x^4 + x^2 + 1 \;=\; 0$$

Although finding multiplicative inverses (or inversion) is relatively cheap, in particular we try to avoid having to perform it repeatedly.

Bibliography

Chapter 1 [17, 20, 10, 39, 41, 16, 29, 11, 26, 30, 9, 21, 27]

Chapter 2,3,etc [34, 1, 28, 8, 18, 15, 2, 24, 42, 12, 33, 3, 4, 35, 37, 5, 45, 22, 31, 13]

Annex B

Elliptic Curves basics

B.1 Introduction

In this appendix, the mathematical concept of elliptic curves will be discussed. Starting from the very first idea where the elliptic curve comes from, treating them as a mathematical objects. Given the concept of finite fields and their properties and laws, elliptic curves and their relation with finite or Galois field will be shown.

B.2 Plane Curves

A plane curve is the set of the form $\{(x,y) \in F * F \ \ or \ \ F^2 \ : f(x,y) = 0\}; \ \ F = \mathbb{Q}$ where $f(x,y)$ is a polynomial in two variables over the field F. There are many familiar examples of plane curves. For example, the circle

$$(x-3)^2 + (y-2)^2 = 4$$

is a plane curve. As well as an ellipse

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

In those cases $f(x,y)$ will be

$$(x-3)^2 + (y-2)^2 = 4$$

for the circle, and

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

for the ellipse.

The degree of the curve is the total degree of f. This is by definition, the maximum of $i + j$ such that there is a monomial ax^iy^j occurring in f with $a \neq 0$. For example, the plane curve

$$x^3 - 10x^2y^2 + 9y^3 + 20 = 0$$

has degree 4 because of the monomial of largest degree in it is $-10x^2y^2$, which has degree $(i = 2 + j = 2) = 4$ Therefore, planes curves of degree 1 are called lines. They are defined by equations of the form

$$ax + by + c = 0$$

As well as planes curves of degree 2 are called conic sections or simply conics. That is because they arise by slicing a double cone in space such as

$$x^2 + y^2 = z^2$$

In general, planes curves of degree 2 have the form

$$ax^2 + bxy + cy^2 + dx + ey + f = 0$$

Well-known examples of planes curves of degree 2 are parabolas, hyperbolas, and ellipses.
Planes curves of degree 3 are called cubic curves. They have a general form as

$$a_1x^3 + a_2x^2y + a_3xy^2 + a_4y^3 + a_5x^2 + a_6xy + a_7y^2 + a_8x + a_9y + a_{10} = 0$$

Where $a_1 \ldots a_{10}$ are numbers. Given the idea of what plane curves means, an elliptic curve will be a certain type of plane curve of degree 3 or cubic curves; namely they are the curves defined by equations of the form

$$y^2 = f(x)$$

Or equivalently

$$y^2 - f(x) = 0$$

Where $f(x)$ is a square-free polynomial of degree 3. 'squarefree' means that $f(x)$ has no multiple roots. For instance

$$y^2 = x^3 - 3x + 2$$

Does not define an elliptic curve, because

$$x^3 - 3x + 2 = (x - 1)^2(x + 2)$$

has 1 as a multiple root. Similarly $y^2 = x^3$ is not an elliptic curve, but $y^2 = x^3 + 1$ is an elliptic curve. By scaling the coordinates and translating, it is possible to convert any elliptic curve into one of the form

$$y^2 = x^3 + Ax + B \tag{B.1}$$

where A and B are numbers (i.e., elements of the field $\mathbb{Q}$). The idea of scaling and translating coordinate allows us to describe any elliptic curve in the form

$$y^2 = x^3 + Ax + B$$

That defines an elliptic curve if only if

$$-(4A^3 + 27B^2) = 0$$

So far, it was assumed that all those plane curves were defined over $\mathbb{Q}$; this means that the coefficients of the polynomial defining an elliptic curve are rational numbers. However, for cryptography purposes the idea is to work with elliptic curve over finite or Galois field. [25]

B.3 Elliptic curves over Finite Fields

Elliptic curves over finite field is heavily applied in cryptography and for the factorization of large integers. It is actually the idea in which this thesis work is based; the use of a defined and private equation of an elliptic curve over a finite field with large number of elements such as $GF(2^{59})$. Basically, and for cryptography purposes, an elliptic curve is a plane curve over a finite field (rather than the field $\mathbb{Q}$ or others fields) which consists of the points (x,y) satisfying the equation

$$y^2 = x^3 + Ax + B$$

Although the problem of computing the points that satisfy this equation over the rational numbers $\mathbb{Q}$ has fascinated mathematicians since the time of the ancient Greeks, it was until 20ths that it was proved that it is possible to construct all the points starting from a finite number by drawing chords and tangents. This is basically the famous theorem of Mordell.

B.3.1 Elliptic curves over $\mathbb{F}_p$

Let $p > 3$ be an odd prime. An elliptic curve E over $\mathbb{F}_p$ (Galois field with p elements) is defined by an equation of the form:

$$y^2 = x^3 + Ax + B$$

Where A, $B \in \mathbb{F}_p$ and $4A^3 + 27B^2 \neq 0(\text{mod p})$. The set $E(\mathbb{F}_p)$ consists of all points (x,y), $\in {\mathbb{F}_p}^2$, (or $\mathbb{F}_p$x$\mathbb{F}_p$) which satisfy the equation written above, together with a special $\mathcal{O}$ called the point at infinity.

Example: Let $p = 23$ and consider the elliptic curve E where $A = 1$ and $B = 4$ such as:

$$y^2 = x^3 + x + 4$$

For the purpose of this example, let's define this curve E over the finite field ($\mathbb{F}_{23}$). In order to verify if this equation is or is not an elliptic curve, this operation must be applied:

$$4A^3 + 27B^2 = 4 + 432 = 436 \equiv 22(\text{mod } 23)$$

Since this equation satisfies the condition, the curve E is indeed an elliptic curve. Therefore the points in $E(\mathbb{F}_{23})$ are $\mathcal{O}$ and these:

(0,2)	(0,21)	(1,11)	(1,12)	(4,7)	(4,16)	(7,3)
(7,20)	(8,8)	(8,15)	(9,11)	(9,12)	(10,5)	(10,18)
(11,9)	(11,14)	(13,11)	(13,12)	(14,5)	(14,18)	(15,6)
(15,17)	(17,9)	(17,14)	(18,9)	(18,14)	(22,5)	(22,19)

B.3.2 Adding points on elliptic curves geometrically

There is a rule called chord-and-tangent method that can be used to add points on an elliptic curve. Here is how it works. Suppose P and Q are points on the elliptic curve E. Join P and Q by the line l. Now l meets E in a third point called $P * Q$. The result of the sum of P and Q is defined to be the reflection of $P * Q$ about the x-axis, not $P * Q$ itself. However, this is a geometry method to add two points on an elliptic curve. What about algebraic methods?

B.3.3 Adding points on elliptic curves algebraically

In this subsection the algebraic method used to add two points on an elliptic curve will be described. It is interesting to see that it is not an automatic operation when the field is a finite or Galois Field. There are two approaches to the addition of two points. In the algebraic way, if $P = (x_1,y_1)$ and $Q = (x_2,y_2)$, then the line l has an equation of the form

$$y = mx + b$$

Solving the simultaneous equations,

$$y = mx + b$$

And

$$y^2 = x^3 + Ax + B$$

Leads to the one-variable equation

$$x^3 - m^2x^2 + (A - 2mb)x + B - b^2 = 0$$

This cubic polynomial has three roots, namely x_1,x_2, and the x-coordinate x_3 of $P * Q = (x_3,y_3)$. Reflecting $P * Q$ in the x-axis, results in $P + Q = (x_3, - y_3)$.

Example: Let E_1 be the curve $y^2 = x^3 - 2x$, $P = (0,0)$ and $Q = (-1,1)$. Then l is the line $y = -x$ and

$$x^3 - m^2x^2 + (A - 2mb)x + B - b^2 = 0$$

becomes

$$x^3 - x^2 - 2x = 0$$

whose roots are $0, -1, and\ 2$. Then $P * Q = (2, - 2)$ and so $P + Q = (2,2)$.

In general, the algebraic method is derived from the geometric method, and then, the following equations are used to sum two points and double the same point, respectively:

- $P + \mathcal{O} = \mathcal{O} + P$ for all $P \in E(\mathbb{F}_p)$
- If $P = (x,y) \in E(\mathbb{F}_p)$, then $(x,y) + (x, - y) = \mathcal{O}$. (The point (x,-y) is denoted by $-P$, and is called the negative of P. Observe that $-P$ is indeed a point on the curve).
- (Point addition) Let $P = (x_1,y_1) \in E(\mathbb{F}_p)$ and $Q = (x_2,y_2) \in E(\mathbb{F}_p)$, where $P \neq \pm Q$. Then $P + Q = (x_3,y_3)$, where

$$x_3 = \frac{y_2 - y_1}{x_2 - x_1}^{2} - x_1 - x_2$$

and

$$y_3 = \frac{y_2 - y_1}{x_2 - x_1}(x_1 - x_3)$$

- (Point doubling). Let $P = (x_1,y_1) \in E(\mathbb{F}_p)$, where $P \neq -P$. The $2P = (x_3,y_3)$, where

$$x_3 = \frac{3x_1^2 + A}{2y_1}^2 - 2x_1$$

and

$$y_3 = \frac{3x_1^2 + A}{2y_1}(x_1 - x_3) - y_1$$

This is actually one of the most fundamental parts in any elliptic curve-based cryptosystem. It is essential to observe that the addition of two elliptic curve points in $E(\mathbb{F}_p)$ requires a few arithmetic operations such as addition, subtraction, multiplication, and inversion, in the underlying field $E(\mathbb{F}_p)$

Example: For an example of Elliptic curve addition, consider the elliptic curve

$$y_2 = x^3 + x + 4$$

Let $P = (4,7)$ and $Q = (13,11)$. Then $P + Q = (x_3,y_3)$ is computed as follows:

$$x_3 = \frac{11 - 7}{13 - 4}^2 - 4 - 13 = 3^2 - 4 - 13 = -8 \equiv 15(\text{mod } 23)$$

and

$$y_3 = 3(4 - 15) - 7 = -40 \equiv 6(\text{mod } 23)$$

Therefore, $P + Q = (15,6)$.

Example: For an example of Elliptic curve doubling, let $P = (4,7)$, then $2P = P + P = (x_3,y_3)$ is computed as follows

$$x_3 = \left(\frac{3\,.\,4^2 + 1}{14}\right)^2 - 8 = 15^2 - 8 = 217 \equiv 10(\text{mod } 23)$$

and

$$y_3 = 15(4 - 10) - 7 = -97 \equiv 18(\text{mod } 23)$$

Therefore, $2P = (10,18)$.

List of Figures

List of Tables

Bibliography

[1] Naveed Ahmed and Christian D. Jensen. *Identity and Privacy in the Internet Age*, chapter A mechanism for identity delegation at authentication level, pages 148–162. 14th Nordic Conference on Secure IT Systems. Springer, Oslo, Norway, October 2009.

[2] Jörg Arndt. *Matters Computational. Ideas, Algorithms, Source Code*. Springer, 2010.

[3] John Gill at Stanford University. Finite fields. http://web.stanford.edu/class/ee387/handouts/notes4.pdf, October.

[4] Keith Conrad at Storrs CT. Finite fields. http://www.math.uconn.edu/∼kconrad/blurbs/galoistheory/finitefi elds.pdf.

[5] Diego F. Aranha at University of Brasilia. Efficient binary field arithmetic and applications to curve-based cryptography. https://www.cosic.esat.kuleuven.be/ches2012/slides/tutorial2.pdf, 2012.

[6] CardLogix. Smart card standards. http://www.smartcardbasics.com/smart-card-standards.html#emv.

[7] Certicom. Elliptic curve cryptography, ecc. https://www.certicom.com/index.php/ecc.

[8] David A. Elizondo, Agusti Solanas, and Antoni Martínez-Ballesté. *Computational Intelligence for Privacy and Security*. Springer, 2012.

[9] Uwe Hansmann, Martin S. Nicklous, Thomas Schäck, Achim Schneider, and Frank Seliger. *Smart Card Application Development Using Java*. Springer, second edition edition, 2002.

[10] Maxim Integrated Products Inc. Pseudo random number generation using linear feedback shift registers. http://www.maximintegrated.com/en/app-notes/index.mvp/id/4400, July 2014.

[11] New Wave Instruments. Linear feedback shift register. http://www.newwaveinstruments.com/resources/articles/m_seque nce_linear_feedback_shift_register_lfsr.htm, July 2014.

[12] Texas Instruments. What's an lfsr. http://www.ti.com/lit/an/scta036a/scta03 6a.pdf, December 1996.

[13] Jeremy Kun. Elliptic curves as elementary equations.

http://jeremykun.com/2014/02/10/elliptic-curves-as-elementary-equations/, February 2014.

[14] Tanja Lange. Finite fields. http://hyperelliptic.org/tanja/teaching/CCI11/online-ff.pdf.

[15] Tanja Lange. A note on lópez-dahab coordinates. *Proceedings 4th Central European Conference on Cryptology, (Tatra Mountains Mathematical Publications*, pages 75–81, July 2004.

[16] Allan Li. Registers, serial-in, serial-out shift registers. http://www.ee.usyd.edu.au/tutorials/digital_tutorial/part2/regist er02.html, July 2014.

[17] Xilins Logicore. Linear feedback shift register v3.0. *Product Specification*, 2003.

[18] Julio López and Ricardo Dahab. *Cryptographic Hardware and Embedded Systems*, chapter Fast Multiplication on Elliptic Curves Over $GF(2^m)$ without precomputation, pages 316–327. Lecture Notes in Computer Science Volume 1717. Springer, 1999.

[19] Luca Macchiarulo. Random number generator design. http://www2.hawaii.edu/∼lucam/EE260/S10/Labs/Lab9/tasks_rng.shtml, September 2014.

[20] Clive Maxfield. Linear feedback shift registers (lfsrs) - part 1. http://www.eetimes.com/document.asp?doc_id=1274550, July 2014.

[21] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, first edition edition, 1996.

[22] J.S. Milne. Elliptic curves. http://www.jmilne.org/math/Books/ect ext5.pdf, 2006.

[23] US National Security Agency. The case for elliptic curve cryptography. http://www.nsa.gov/business/programs/elliptic_curve.shtml, January 2009.

[24] Christof Paar and Jan Pelzl. *Understanding Cryptography. A Textbook for Students and Practitioners.* Springer, 2009.

[25] Bjorn Poonen. Elliptic curves. http://mathcircle.berkeley.edu/BMC4/Handou ts/elliptic.pdf.

[26] Emmanuel Pouly. A stream cipher based on several lfsrs: The geffe generator. http://emmanuel.pouly.free.fr/cipher1.html, July 2014.

[27] Dr. Rainer A. Rueppel. *Analysis and Design of Stream Ciphers.* Springer, 1986.

[28] Mark Stamp and Richard M. Low. *Applied Cryptanalisis, Breaking Ciphers in the Real World.* Wiley-Interscience, 2007.

[29] Wayne Storr. The shift register. http://www.electronics-tutorials.ws/sequential/seq_5.html, July 2014.

[30] C. Stroud. Linear feedback shift register, lfsr. http://www.eng.auburn.edu/∼strouce/class/elec6250/LFSRs.pdf, July 2014.

[31] Nick Sullivan. A (relatively easy to understand) primer on elliptic curve

cryptography. http://arstechnica.com/security/2013/10/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/, October 2013.

[32] Math 28S Swarthmore College. Definition of a field. http://www.swarthmore.edu/NatSci/dmcclen1/math028S/m028f20 11fields.pdf, Fall 2011.

[33] Mike Thomsen. Linear feedback shift registers, galois fields, and stream ciphers. http://www.cs.rit.edu/∼mxt4877/Crypto2/LFSR,%20GF,%20SC.p df, May 2012.

[34] Unknown. Authentication. http://en.wikipedia.org/wiki/Authentica tion, July 2014.

[35] Unknown. Discrete mathematics/finite fields. http://en.wikibooks.org/wiki/Discrete_Mathematics/Finite_fields, July 2014.

[36] Unknown. Elliptic curve cryptography. http://en.wikipedia.org/wiki/Elliptic_curve_cryptography, July 2014.

[37] Unknown. Field (mathematics). http://en.wikipedia.org/wiki/Field_(mathematics)#Reals.2C_complex_numbers.2C_and_p-adic_numbers, July 2014.

[38] Unknown. Finite field. http://en.wikipedia.org/wiki/Finite_field# F16, July 2014.

[39] Unknown. Linear feedback shift registers. http://en.wikipedia.org/wiki/Linear_feedback_shift_register, July 2014.

[40] Unknown. Registre à décalage à rétroaction linéaire. http://fr.wikipedia.org/wiki/Registre_à_décalage_à_rétroaction_linéaire, September 2014.

[41] Unknown. Stream cipher. http://en.wikipedia.org/wiki/Stream_cip her, July 2014.

[42] Lawrence C. Washington. *Elliptic Curves. Number Theory and Cryptography.* CHAPMAN HALL/CRC, 2003.

[43] Eric W. Weisstein. Finite field. http://mathworld.wolfram.com/FiniteField.html.

[44] Eric W. Weisstein. Primitive polynomial. http://mathworld.wolfram.com/PrimitivePolynomial.html, July 2014.

[45] Xinmiao Zhang. Finite field arithmetic and implementations. http://fetweb.ju.edu.jo/staff/EE/jrahhal/PDF/finitefield.pdf.