A usability study of Elliptic Curves

DANIYAL TARIQ

A thesis presented for the degree of
Master of Science

DEPARTMENT OF MATHEMATICS AND STATISTICS
UNIVERSITY OF TURKU
FINLAND
DECEMBER 2018

"A smile is a curve that sets everything straight."

Phyllis Diller

A usability study of Elliptic Curves

DANIYAL TARIQ

Abstract

In the recent years, the need of information security has rapidly increased due to an enormous growth of data transmission. In this thesis, we study the uses of elliptic curves in the cryptography. We discuss the elliptic curves over finite fields, attempts to attack; discrete logarithm, Pollard's rho algorithm, baby-step giant-step algorithm, Pohlig-Hellman algorithm, function field sieve, and number field sieve. The main cryptographic reason to use elliptic curves over finite fields is to provide arbitrarily large finite cyclic groups having a computationally difficult discrete logarithm problem.

Key-words: Elliptic Curve, Diffie-Hellman Protocols, Discrete Logarithm, Function Field Sieve, Number Field Sieve

Acknowledgements

In the name of Allah, the most Beneficent, the most Merciful

This thesis is written as part of Master's Degree programme in Information Security and Cryptography at University of Turku, Finland. I would like to express my sincere gratitude to my supervisor Mika Hirvensalo for the patience, support, fruitful discussions and excellent guidance. He supported me all the way during my thesis work. I would like to thank the respectable members of the faculty of Science and Engineering. Finally, I thank all my family members especially to my mother for her love, father for his moral support, grandfather and friends for their encouragement.

List of Figures

3.1 Diffie Hellman Protocol 9

5.1 EC Addition 27
5.2 ECDLP 28
5.3 EC $y^2 = x^3 + 7x + 20$ 31
5.4 EC $E_{41}(7, 20)$ 32

List of Tables

1.1 Comparing the key size of ECC and RSA 2

4.1 Pollard's rho 18

5.1 EC over $\mathbb{F}_q$ 29

7.1 Attacks Compare Complexity 35

List of Acronyms

DL	Discrete Logarithm
DH	Diffie-Hellman
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
FFS	Function Field Sieve
NFS	Number Field Sieve
OTP	One-Time Pad
RSA	Rivest Shamir Adleman

Contents

Acknowledgements iii

List of Figures iv

List of Tables v

1 Introduction 1

2 Cryptographic Protocols 4
- 2.1 Symmetric Cryptography . . . 4
- 2.2 Asymmetric Cryptography . . . 5
- 2.3 Key distribution protocols . . . 7

3 Diffie-Hellman Protocol 8
- 3.1 Background . . . 8
- 3.2 Implementation . . . 8
- 3.3 Feasibility and security . . . 9
- 3.4 ElGamal . . . 12

4 Attempts to break Diffie-Hellman 14
- 4.1 Discrete Logarithm . . . 14
- 4.2 Brute-Force . . . 16
- 4.3 Pollard's rho algorithm for discrete logarithm . . . 16
- 4.4 Baby-step giant-step Algorithm . . . 18
- 4.5 Pohlig-Hellman Algorithm . . . 19
- 4.6 Function field sieve . . . 20
- 4.7 Number field sieve . . . 21
- 4.8 On Victor Shoup's lower bound . . . 23

5 Elliptic Curves 25
- 5.1 Background . . . 25
- 5.2 The addition formulas for the Curve . . . 26
- 5.3 Group of rational points . . . 29
- 5.4 Diffie-Hellman protocol on Elliptic Curves . . . 29

6 Other uses in cryptography 33

7 Conclusion **35**

Bibliography **37**

Chapter 1

Introduction

Cryptography is the study of secure communication in the presence of an adversary. The word "kryptos" comes from Greek and means "hidden" as well as "graphein" which means "to write". Most of the cryptographic algorithms are based on mathematical problems that are hard to solve. The first known evidence of cryptography was found in an inscription carved around 1900 BC, but only about one hundred years ago cryptography was being studied as a science.

A cipher is an algorithm used for encryption and decryption. In the second half of the 20th century, the ciphers were used substantially. Since then, they have been used in business, banking, and electronic fund transfers. Nowadays, in the beginning of 21st century, cryptography is used in everyday life information transmission, like Wifi, Bluetooth, 4G security, Transport Layer Security(TLS)/Secure Sockets Layer(SSL) secure web transactions.

In mathematics, there are many hard problems which are computationally infeasible to calculate. The elliptic curve discrete logarithm problem is one for which no simplifying shortcut is known. This motivates the use of elliptic curves to implement cryptographic systems. The first practical public key cryptosystem was the Diffie-Hellman key exchange protocol [27, 31].

Elliptic curve cryptography (ECC) [25] was introduced independently by two mathematicians Neal Koblitz and Victor Miller. In 1985, EC over finite field have been applied for construction of cryptosystem, factoring integers and primality testing. For at least 150 years elliptic curves have been studied a lot as geometric or algebraic entities. From that study, many theories have emerged. In 2003, the Pollard's rho algorithm was used to compute the elliptic curve DL for an elliptic curve with 109 defining bits. Pollard proposed the original version of the number field sieve (NFS) in 1988. It appears as the

best advanced factoring method. Daniel Gordon [10] was probably the first to find out that the number field sieve method provides a good algorithm for discrete logarithm. In 2005, a 200-digit number factorization using number field sieve (NFS) taking sub-exponential time was also implemented. In ECC, the experiments seem to indicate for the same level of security with small parameters than in RSA [21, 24]. In this thesis, theoretical background justifying this observation will be presented.

The comparison of the EC to RSA can be represented as depicted in Table 1.1.

TABLE 1.1: Comparing the key size of ECC and RSA

Symmetric key	RSA/DLP	ECC
64bit	700bit	128bit
80bit	1024bit	160bit
128bit	2048-3072bit	256bit

RSA (Rivest Shamir Adleman) cryptosystem security is based on the integer factorization problem. It takes sub-exponential time [38] to solve integer factorization problem (IFP) but solving ECDLP using the best known algorithm appears somewhat harder. Consequently, ECC with the same security level takes less memory on devices compared to RSA. But in this current advanced digital world the public key cryptography segments are mainly covered by RSA. Moreover, in RSA, the multiplication is computationally an easy task, whereas the other direction, to find the prime factors of a composite number n is computationally difficult. This one-way property can also be observed in other important processes in number theory. The exponentiation in a large finite field is one important example.

Elliptic Curve Cryptography (ECC) is considered as one of the most advanced techniques to secure the systems. In 2004–2005, ECC started being used widely. A paper on Elliptic Curve Cryptography (ECC) [38] ensured a level of security which is equivalent to other public key systems but having a shorter key length. Examples include: Elliptic Curve versions of El-Gamal, Diffie-Hellman, etc.

Because of that, ECC has advantages over other systems based on encryption and decryption algorithm and the time lapse on the key generation [24]. Therefore, in a resource constraint device ECC may be more suitable choice. The main reason for the advantages provided by elliptic curves is that their rational point groups appear resistant over the best discrete logarithm algorithms designed for finite fields [38, 24].

The outline of the thesis is as follows. In Chapter 2, the cryptographic protocols are presented. In Chapter 3, the Diffie-Hellman Protocol is introduced. Chapter 4 continues with the Diffie-Hellman and presents attempts to break it. In Chapter 5, the

main focus of the thesis, elliptic curves, and the example of implementing elliptic curves are illustrated. In Chapter 6, the main uses of elliptic curve cryptography are described. In Chapter 7, this thesis is summed up and we discuss future aspects.

Standard Notations

Throughout this thesis, the symbols

$$\mathbb{Q}, \mathbb{C}, \mathbb{R}, \mathbb{F}_q, n, N, \mathbb{Z}, \text{ and } O$$

are used to denote the rational numbers, complex numbers, real numbers, a field with q elements, bit length (of a group element), group size, integers, and ordo-notation, respectively.

Chapter 2

Cryptographic Protocols

2.1 Symmetric Cryptography

Modern cryptography is divided into two types, symmetric and asymmetric. Symmetric-key cryptography provides message confidentiality, using single key for both encryption and decryption system. More precisely, to encrypt and decrypt the both parties use the same key to communicate. In symmetric cryptography, absolute information security can be achieved (for example One-time pad, OTP, for short).

An OTP encrypts and decrypts as follow: Generate a random bit string as long as the message, this will serve as a key. Calculate the exclusive or (XOR) of the plain text and the key to create the cipher text. For decryption, XOR the cipher text with the original key. Thus, one-time pad as presented here is a symmetric and reciprocal cipher. Other functions (e.g. addition modulo n) could be used to combine the plain text and the key to yield the cipher text, although the outcome system may not be a reciprocal cipher.

Symmetric cryptography has a disadvantage, as the communicating parties must meet prior to communication to share the key.

Example 1. (The following simple example is from [20]) If we are using the 26-letter alphabet A-Z with numerical equivalents $0-25$. Let the letter $P \in \{0, 1, ..., 25\}$ stand for a plain text message unit. Define a function f from the set $\{0, 1, ..., 25\}$ to itself by the rule

$$f(P) = \begin{cases} p+3, & \text{if } x < 23 \\ p-23, & \text{if } x \geq 23. \end{cases}$$

In simple way, f adds 3 modulo 26: $f(P) \equiv P + 3 \pmod{26}$. We see it is convenient to use modulo. For example, to encipher the word "YES", we add 3 modulo 26 and obtain 1, 7, 21. Then translating back to letters we obtain "BHV". More generally, $C \equiv P + B \pmod{26}$, where the shift B serves as an encryption and decryption key. Decryption is simply $P \equiv C - B \pmod{26}$.

Example 2. This example is also from [20]. Let us consider an affine matrix transformation of pairs of digraphs

$$C \equiv \begin{pmatrix} a & b \\ c & d \end{pmatrix} P + \begin{pmatrix} e \\ f \end{pmatrix} \pmod{N^2},$$

where

$$0 \leq a, b, c, d, e, f < N^2.$$

Here a column vector P corresponds to the numerical equivalents of two consecutive plain text digraphs in an N-letter alphabet. However, we choose k between 0 and N^{12} where k is a randomly chosen integer, and a, b, c, d, e, f to be the six digits in k written to the base N^2. Here the matrix and the column vector together serve as an encryption key. The decryption is done by computing

$$P \equiv \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} \left(C - \begin{pmatrix} e \\ f \end{pmatrix} \right) \pmod{N^2},$$

Use the explicit forms of A and g as above.

2.2 Asymmetric Cryptography

Public-key cryptography, also known as asymmetric-key cryptography provides good key distribution and key management systems. The technique provides authentication, confidentiality and integrity of information transformation.

Asymmetric key encryption uses two separate keys for the encryption (public key) and decryption (private key). Mainly, one key is used to encrypt the message and different key is used to decrypt it.

The security of asymmetric cryptography is based on the assumption that it should be computationally **very** difficult to obtain the private key from the public one. The advantage of asymmetric cryptography is obvious: The communicating parties do not need to meet before the communication. Unfortunately, the disadvantage is that there

is no absolute information security. The security is merely based on computational hardness. Here is RSA as an example.

Example 3. (RSA keys) In public-key cryptosystems the key is divided into two parts e and d, where e is the encryption exponent and d is used for decryption. Here, idea is that e can be published while only d is kept secret.

- Let $n = pq$, where p and q are two large primes. Then we have $\varphi(n) = (p-1)(q-1)$ where φ is Euler's totient function.
- Public key: pair of (n, e), where $gcd(e, \varphi(n)) = 1$.
- Private key: pair of (n, d), where d is the inverse of e modulo $\varphi(n)$, i.e., $ed = 1 \pmod{\varphi(n)}$.

Mathematical descriptions

- **Symmetric cryptosystem:** Let P, C, and K be the set of all plain texts, cipher texts, and keys, respectively. The encryption can be then viewed as a function

$$e : P \times K \to C, \quad e(p, k) = c$$

  and the decryption as a function

$$d : C \times K \to P, \quad d(c, k) = p.$$

  Then encryption and decryption can be done by anyone knowing the key.

- **Asymmetric cryptography:** Let P, C, and K be as above, but now it is assumed that each $k \in K$ is divided into two parts, $k = (k_{\text{publ}}, k_{\text{priv}})$. The encryption can be seen as a function

$$e : P \times K \to C, \quad e(p, k_{\text{publ}}) = c,$$

  and the decryption as a function

$$d : C \times K \to P, \quad d(c, k_{\text{priv}}) = p.$$

  Now anyone knowing k_{publ} can perform the encryption, but decryption is possible only for those knowing also k_{priv}. It is supposed that it is computationally intractable to compute k_{priv} from k_{publ}.

2.3 Key distribution protocols

The public key cryptosystems implementations appear slow when compared to those of symmetric-key cryptosystems [13]. However, the procedure of concurring on a key for a classical cryptosystem can be accomplished impartially and efficiently using a public key system. One of the first such proposals was the system of Diffie-Hellman [13], and its security was based on the Discrete Logarithm Problem.

Rounds of asymmetric protocols are used to form a shared key between communicating parties. Once the shared key is established, it can be used in a symmetric protocol. An advantage that there is no need to meet in advance, and the disadvantage of key distribution that there is no absolute security. The security of the key distribution protocols depend on the belief that nobody can extract confidential information (easily) from the information being sent from communicating parties to each other. Diffie-Hellman key distribution [20] is one of the most famous protocols for key distribution and would logically belong to this subsection, but as it includes so many aspects relevant to the study, the Diffie-Hellman key exchange system is explained in the next chapter.

Chapter 3

Diffie-Hellman Protocol

3.1 Background

In 1976 Diffie and Hellman suggested digital signature and public key cryptography. The publication of their paper *New directions in cryptography* leads to beginning of public-key cryptography. Nowadays their protocol plays a vital role on the internet because most of the security foundation stands on it. It is protecting daily internet communication and financial transactions [9]. We mentioned in the previous chapter that the Diffie-Hellman is a protocol to transfer cryptographic key through unsecure communication channel. In this chapter, we present the protocol in details and show that its security depends on the hardness of DLP.

3.2 Implementation

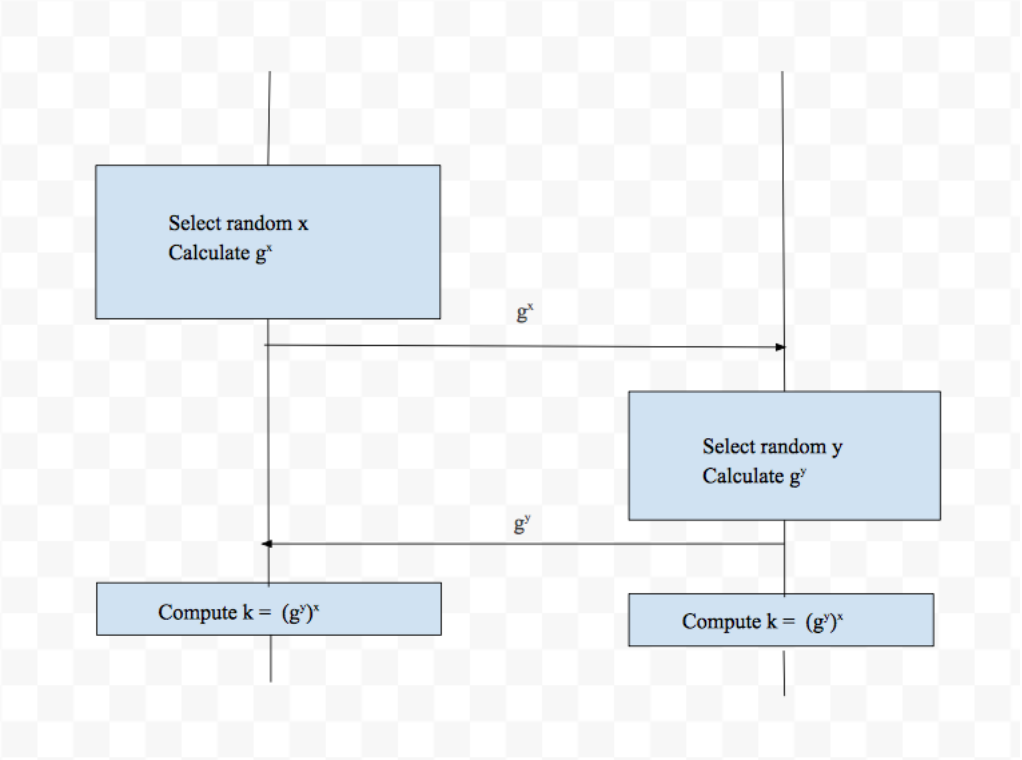
The protocol can be implemented for any cyclic group $G = \langle g \rangle$. Let $N = |G|$ be the cardinality of G and $n \approx \log_2 N$ the number of bits needed to present an element of G. Group generator g can be public. Both Alice and Bob know the group (also the public audience may know the group). Alice picks random $x \in \{0, 1, \ldots, N-1\}$, computes $x \to g^x$ and sends g^x to Bob via a public channel. Bob picks random $y \in \{0, 1, \ldots, N-1\}$, computes $y \to g^y$ and sends g^y to Alice via a public channel. As now Alice has both x and g^y, she can compute $(g^y)^x = g^{xy}$. Equally, Bob has g^x and y, and hence he can compute $(g^x)^y = g^{xy}$. Hence, g^{xy} can be used as a common key, which is shared by Alice and Bob. However, an eavesdropper knows only g^x and g^y. Without computing the DLP, it seems impossible to compute g^{xy} knowing just g^x and g^y. In what follows, the algorithm is explained in more details.

Diffie-Hellman Protocol for key distribution

For the protocol, it is supposed that $G = \langle g \rangle$ is a cyclic group of order $N = |G|$.

1. Alice chooses random $x \in \{0, 1, \ldots, N-1\}$.
2. Alice computes $x \to g^x$ and sends it to Bob (via a public channel).
3. Bob chooses random $y \in \{0, 1, \ldots, N-1\}$.
4. Bob computes $y \to g^y$ and sends it to Alice (via a public channel).
5. Alice computes $(g^y)^x = g^{xy}$.
6. Bob computes $(g^x)^y = g^{xy}$.

FIGURE 3.1: Diffie Hellman Protocol



3.3 Feasibility and security

As protocol users must compute $x \to g^x$, the (discrete) exponential function should be feasible, i.e. easy to compute in the group. Indeed, it can always be computed efficiently via squaring method.

As also g^x (and g^y) will be available for public audience, the operation $g^x \to x$ (the discrete logarithm, will be discussed in next chapter) should not be feasible, meaning that is should be difficult to compute. The aforementioned assumption is not straightforward for all cyclic groups.

Repeated Squaring Method for g^k

Modular exponentiation, i.e. calculating $g^k \mod N$, is time consuming operation and frequent task, see [18]. For very large integers, the repeated squaring can be used. In the following examples, we discuss the repeated squaring method and compare it to the trivial one.

Example 4. Compute g^{100} directly as $g^{100} = g \cdot \ldots \cdot g$. Here 99 multiplications are needed.

Example 5. Compute g^{100} by first finding the binary expansion $100 = 2^6 + 2^5 + 2^2$.

Then compute $g^2 = g \cdot g$, $g^4 = g^2 \cdot g^2$, $g^8 = g^4 \cdot g^4$, $\ldots$, $g^{64} = g^{32} \cdot g^{32}$,

and finally

$$g^{100} = g^{64} \cdot g^{32} \cdot g^4.$$

The number of multiplications needed to calculate g^{100} by this method is only 8.

In a general case, calculating g^k using the trivial method requires $k-1$ simple multiplications. Hence, as k may be close to the order N of the group, too many multiplications are needed. In any case, we may assume the number of multiplications is $k < N \approx 2^n$, where n is the number of bits needed to represent a group element.

The repeated squaring procedure goes as follow: $g^2 = g \cdot g$, $g^4 = g^2 \cdot g^2, \cdots, g^{2^L} = g^{2^{L-1}} \cdot g^{2^{L-1}}$ requires L multiplications where L is length of the binary representation of k and

$$g^k = g^{k_L 2^L + k_{L-1} 2^{L-1} + \cdots + k_1 2 + k_0} = (g^{2^L})^{k_L} \cdot (g^{2^{L-1}})^{k_L - 1} \cdots (g^2)^{k_1} \cdot g^{k_0}$$

requires L multiplications. Therefore we need at most $2L \leq 2 \log_2 N$ multiplications. Since in a binary representation,

$$k = k_L 2^L + k_{L-1} 2^{L-1} + \cdots + k_1 2^1 + k_0 2^0 \quad \geq \quad 2^L,$$

it follows that $\log_2 k \geq L$. This implies that $L \leq \log_2 k \leq \log_2 N$, which is approximately $\log_2 2^n = n$. The complexity of the repeated squaring multiplications for computing g^k is $O(L + L) = O(\log_2 N) = O(n)$.

Example 6. For $G = \mathbb{Z}_N = \langle g \rangle$ the exponentiation becomes $x \to xg \pmod N$, and the discrete logarithm becomes a problem of resolving x when xg (mod N) is known. Here N is the cardinality of the group, and $n = \log_2 N$ is the number of bits required to present group elements.

The Euclidean algorithm helps to find x from xg. In fact, the inverse of $g \pmod N$ can be computed in time $O(n^3)$ by standard Euclidean algorithm. Hence, in this group, the discrete logarithm problem can be solved in polynomial time, when measured in terms of the representation length of the elements. In any case, the discrete logarithm in this group is feasible, and the group cannot therefore be used to implement any cryptographic protocol.

Example 7. Consider the multiplicative group $G = \mathbb{F}_p^* = \mathbb{F}_p \setminus \{0\}$ of the finite field $\mathbb{F}_p$ with $p \in \mathbb{P}$ elements. To compute, the discrete logarithm $g^a \to a$ appears to be problematic because no fast (general) algorithm is known. Here $N = p - 1$, and the number of bits needed to present an element is $n = \log_2(p - 1) = \log_2 N$.

Example 8. Let Alice and Bob agree on $p = 37$ and $g = 2$. The following example is from [2], and it demonstrates the Diffie-Hellman key exchange.

Alice chooses $x = 14$ and sends to Bob

$$A = 2^{14} \pmod{37} \equiv 30.$$

Bob chooses $y = 23$ and sends to Alice

$$B = 2^{23} \pmod{37} \equiv 5.$$

Bob receives 30 and calculates

$$K_y \equiv 30^{23} \pmod{37} \equiv 28.$$

Alice receives 5 and computes

$$K_x \equiv 5^{14} \pmod{37} \equiv 28.$$

Alice and Bob now agree on 28 as their shared key.

As the modular exponentiation can be computed quickly, the Diffie-Hellman key exchange protocol can be executed efficiently.

3.4 ElGamal

Neal Koblitz [20] stated that the encryption of ElGamal system is an asymmetric key encryption algorithm for public-key cryptography, which is based on the Diffie-Hellman key exchange. In this section, we shortly discuss ElGamal.

ElGamal encryption can be defined over any cyclic group G. The security of ElGamal depends on the difficulty of a certain problem in G related to computing DL. In the forthcoming explanation, we will use multiplicative notations, and g is the generating element.

If the problems appearing in Diffie-Hellman are hard, then the ElGamal public key cryptosystems are secure [14].

The security of other protocols for instance Diffie-Hellman key exchange, pairing-based cryptosystems, and digital signature schemes depend on intractability of solving Diffie-Hellman problems. Thus, in the point of view of practical cryptography, the study of the hardness of Diffie-Hellman problems is extremely important [23].

ElGamal encryption

This following algorithm is originally from the book of applied cryptography [27].

Alice (A) creates public and private keys by the following steps:

- Step 1: Create a large random prime p and a generator g of the multiplicative group $\mathbb{Z}_p^*$ of the integers modulo p.
- Step 2: Select a random integer $a, 1 \leq a \leq p-2$, and calculate $g^a \mod p$
- Step 3: Alice's public key is (p, g, g^a); Alice's private key is a.

Encryption: Bob (B) follows these steps

- Receive Alice's authentic public key (p, g, g^a).
- Encode the message as an integer m in the range $\{0, 1, \cdots, p-1\}$.
- Choose a random integer $k, 1 \leq k \leq p-2$.

- Calculate $g_1 = g^k \mod p$ and $\delta = m \cdot (g^a)^k \mod p$.
- Send the ciphertext $c = (g_1, \delta)$ to Alice.

Decryption: Alice's should follow these steps:

- Using a as a private key, calculate $g_1^{p-1-a} \mod p$ (note: $g_1^{p-1-a} = g_1^{-a} = g^{-ak}$).
- Recover m by calculating $(g_1^{-a}) \cdot \delta \mod p$.

Chapter 4

Attempts to break Diffie-Hellman

4.1 Discrete Logarithm

Interest in studying the discrete logarithm problem rose in the mid-1970s with the public key cryptography. While the ordinary (continuous) logarithm on real numbers is not difficult to compute, its discrete version in many groups appears hard. The routine applications of discrete logarithms in small or large fields depend on computer algebra system [32].

Now let there be a finite group, such as $\mathbb{F}_q^*$. Let β be an element of the form α^x and assume that the base α is fixed. So the question comes that how can we get the power of α that gives β, that is, how can we calculate $x = \log_\alpha \beta$?

This **problem** is called the discrete logarithm problem (DLP).“Discrete” is the word distinguishing the finite group case from the classical or continuous case [20].

Definition 4.1 (Discrete logarithm)**.** Let G be a finite cyclic group of order N and let g_1 be a generator of G, and let $g \in G$. The discrete logarithm of g to the base g_1, denoted by $\log_{g_1} g$, is the unique integer $x \in \{0, 1, \ldots, N-1\}$, such that $g = g_1^x$.

Remark 4.2. The above definition uses the multiplication notation for the group operation. In additive notation, the discrete logarithm of g to the base g_1, denoted by $\log_{g_1} g$ is the unique integer $x \in \{0, 1, \ldots, N-1\}$, such that $g = xg_1$.

A group where the additive notation is used by default, is the group of rational points of an EC defined later.

Remark 4.3. In group $\mathbb{F}_p^*$, we use the the multiplicative notation. On the other hand, in a group such as $\mathbb{Z}_N$, the additive notation is used.

Example 9. Consider the group $\mathbb{F}_{11}^*$, in which we can choose $g = 2$ as generator. Since $2^6 \equiv 9 \pmod{11}$, it follows that $\log_2 9 = 6$ in $\mathbb{Z}_{11}^*$. Also, $2^6 \equiv 2^{16} \equiv 2^{26} \equiv 9 \pmod{11}$.

Example 10. Consider another group $\mathbb{F}_{97}^*$ in which, we can choose 5 as a generator. Computing $5^{32} \equiv 35$, we obtain that $\log_5 35 = 32$ in $\mathbb{Z}_{97}^*$.

Example 11. Consider the generator $g = 2$ of $\mathbb{F}_{19}^*$. The successive powers of 2 are then 2, 4, 8, 16, 13, 7, 14, 9, 18, 17, 15, 11, 3, 6, 12, 5, 10, 1. Then the DL of 7 to the base 2 is hence 6.

By the following theorem, any algorithm that evaluates logarithm in some base can also be apply to calculate it in different base which is generator of G.

Theorem 4.4. *Suppose g and g_1 are two generators of G, where G is a cyclic group and $\beta \in G$. Any efficient algorithm to evaluate $\log_g \beta$ can be converted to an efficient algorithm to evaluate $\log_{g_1} \beta$.*

Proof. We can write $\beta = g_1^{\log_{g_1} \beta} = (g^{\log_g g_1})^{\log_{g_1} \beta}$. On the other hand we know that $\beta = g^{\log_g \beta}$. Hence, we have $\log_g \beta = (\log_g g_1)(\log_{g_1} \beta)$.

□

The group $\mathbb{Z}_n^*$ is used in many cryptographics mechanisms, such as in the identification scheme of the Girault, and in the key agreement protocols of the McCurley and Yacobi [24, 12]. Moreover in the Diffie-Hellman problems in a cyclic subgroup $\mathbb{Z}_n^*$, discrete logarithm and the problems of factoring n have a connection to cryptography.

In the next sections we will study various algorithms for discrete logarithm. These include the brute force technique, Pollard's rho, baby-step, Pohlig-Hellman algorithm, function field sieve and the number field sieve.

The four first algorithms can be apply in any cyclic group. The performance of Pohlig-Hellman depends very much on the factorization of the credential of the cardinality N of the group; it is especially efficient if N can be present as a product of small primes. Furthermore, the function field sieve algorithm is especially designed for groups $\mathbb{F}_{p^n}$ where p is a prime for computing the problem of DL and cannot be implemented in a general cyclic group.

4.2 Brute-Force

The Brute-Force attempt has been introduced as the most obvious way to attack DLP. Consider a cyclic group $G = \langle r \rangle$ with a generator g and cardinality N. Let $h = g^x$ be the element of which logarithm $\log_g h$ we are trying to calculate.

Compute g^y for all $y \in \{0, 1, \ldots, N-1\}$, stopping when we get a match with h.

If x is selected randomly and uniformly, then in average one must compute $N/2$ powers of g to find the correct x with at least 50 % probability. This exhaustive search algorithm is clearly exponential in the input size as we can represent each member of G using $\log_2(N)$ binary digits. Notice that for finding the correct power absolutely surely, you must, in the worst case, check g^x for all $x \in \{0, 1, \ldots, N-1\}$. Thus, if this were the best-known algorithm for finding the DLP, then our systems would be secure [50].

4.3 Pollard's rho algorithm for discrete logarithm

Pollard's rho algorithm for the DLP is a randomized algorithm just like baby-step giant-step algorithm (it will be discussed in section 4.4) but it takes negligible amount of storage. That is why in practical problems the Pollard's rho algorithm is used, see [27].

In this algorithm we are trying to calculate $\log_\alpha \beta$. As described in the book, see [27], the group G is partitioned into three same size disjoint subsets, S_0, S_1 and S_2. Selection process of subsets is important; for instance, we require $1 \notin S_2$. Describe a sequence of group elements $x_0, x_1, x_2, \ldots$ by $x_0 = 1$ and

$$x_{i+1} = f(x_i) = \begin{cases} \beta \cdot x_i, & \text{if} \quad x_i \in S_0 \\ x_i^2, & \text{if} \quad x_i \in S_1 \\ \alpha \cdot x_i, & \text{if} \quad x_i \in S_2 \end{cases}$$

for $i \geq 0$. Using this sequence, we may define two new ones a_0, a_1, ... and b_0, b_1, ... satisfying as follows: $a_0 = b_0 = 0$ and for $i \geq 0$ we have

$$a_{i+1} = \begin{cases} a_i, & \text{if} \quad x_i \in S_0 \\ 2_{a_i} \pmod{n}, & \text{if} \quad x_i \in S_1 \\ a_i + 1 \pmod{n}, & \text{if} \quad x_i \in S_2 \end{cases}$$

and

$$b_{i+1} = \begin{cases} b_i + 1 \pmod{n}, & \text{if } x_i \in S_0 \\ 2b_i \pmod{n}, & \text{if } x_i \in S_1 \\ b_i, & \text{if } x_i \in S_2 \end{cases}$$

Floyd's cycle-finding algorithm [27] can be used to find two elements x_i and x_{2i} so that $x_i = x_{2i}$. Thus $\alpha^{a_i}\beta^{b_i} = \alpha^{a_{2i}}\beta^{b_{2i}}$ and we have $\beta^{b_i - b_{2i}} = \alpha^{a_{2i} - a_i}$.

Calculating the logarithms to the base α for both sides of this last equation provides

$$(b_i - b_{2i}) \cdot \log_\alpha \beta \equiv (a_{2i} - a_i) \pmod{n}.$$

It can be seen that $b_i \neq b_{2i} \pmod{n}$. Note that $b_i \equiv b_{2i}$ appears with a vanishing probability.

Algorithm: Pollard's rho algorithm for computing discrete logarithms

The previous discussion leads to the following algorithm [27].

Input: a generator α of a subgroup G, and an element $\beta \in G$

Output: the discrete logarithm $x = \log_\alpha \beta$

- Set $x_0 \leftarrow 1, a_0 \leftarrow 0, b_0 \leftarrow 0$.
- For $i = 1, 2, \ldots$, compute the values x_i, a_i, b_i and x_{2i}, a_{2i}, b_{2i} using the quantities x_{i-1}, a_{i-1}, b_{i-1} and x_{2i-2}, a_{2i-2}, b_{2i-2} (as calculated earlier).
  - If $x_i = x_{2i}$, then the algorithm proceeds as follows:
    - Set $r \leftarrow b_i - b_{2i} \pmod{n}$
    - If $r = 0$ then the algorithm leads to fail; otherwise, calculate $r^{-1}(a_{2i} - a_i) \pmod{n}$ and return x.

Example below illustrates Pollard's rho algorithm with small parameters from lecture notes, see [39].

Example 12. (Pollard's rho algorithm for logarithms in a subgroup of $\mathbb{F}_{503}^*$)

We choose $\alpha = 2$ as the generator of the subgroup G of $\mathbb{F}_{503}^*$. The order of G is 251. Also, $\beta = 169$ belongs to this subgroup. Divide G to the partitions as follows: $x \in S_0$ if $x \equiv 1 \pmod 3$, $x \in S_1$ if $x \equiv 0 \pmod 3$, and $x \in S_2$ if $x \equiv 2 \pmod 3$. Below the table describes $x_i, a_i, b_i, x_{2i}, a_{2i}$, and b_{2i}. Calculations show that $x_{14} = x_{28} = 263$. We gain

by computing the $r = b_{14} - b_{28} = 247 \pmod{251}, r^{-1} = 247^{-1} = 188 \pmod{251}$, and $r^{-1}(a_{28} - a_{14}) = 123 \pmod{251}$. Hence, $\log_2 169 = 123$.

TABLE 4.1: Pollard's rho

i	x_i	a_i	b_i	x_{2i}	a_{2i}	b_{2i}
1	169	0	1	393	0	2
2	393	0	2	205	0	5
3	28	0	4	323	0	12
4	205	0	5	286	2	12
5	441	0	6	229	2	14
6	323	0	12	443	3	15
7	143	1	12	263	5	15
8	286	2	12	46	7	15
9	46	2	13	473	7	17
10	229	2	14	383	9	17
11	473	2	15	23	11	17
12	443	3	15	229	12	18
13	383	4	15	443	13	19
14	263	5	15	263	15	19

The following lemma is from [27].

Lemma 4.5. *Let G be a group of order N, where N is a prime. Assuming the function $f : G \to G$ of the Pollard's rho algorithm behaves like a random function, the anticipated running time of Pollard's rho algorithm for discrete logarithms in G is $O(\sqrt{N})$ group operations. In addition, the algorithm requires negligible storage.*

4.4 Baby-step giant-step Algorithm

We are trying to compute $\log_\alpha \beta$. Let $m = [\sqrt{N}]$, where N is the order of α. If $\beta = \alpha^x$, then it is possible to write $x = im + j$, where $0 \leq i, j < m$. Thus, $\alpha^x = \alpha^{im}\alpha^j$ implying $\beta(\alpha^{-mi}) = \alpha^j$. This proposes the following algorithm for calculating x, see [27].

Computing the DL by baby-step giant-step algorithm

Input: A generator α of a cyclic group G of order N, and an element $\beta \in G$
Output: the discrete logarithm (DL) $x = \log_\alpha \beta$.

1. Denote $m = [\sqrt{N}]$.
2. Construct a table with items (j, α^j) for $0 \leq j < m$ and arrange it according to the second component.

3. Calculate α^{-m} and let $\gamma \leftarrow \beta$.

4. For i from 0 to $m-1$ do

   (a) Check whether γ is the second component of some entry in the table.

   (b) If $\gamma = \alpha^j$ then return $x = im + jv$.

   (c) Otherwise, let $\gamma \leftarrow \gamma \cdot \alpha^{-m}$.

4.5 Pohlig-Hellman Algorithm

Pohlig-Hellman Algorithm for solving logarithms uses the factorization of the order N of the group G. We represent here the Pohlig-Hellman algorithm from [27].

Assume that $N = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r}$ is the prime factorization of N. If $x = \log_\alpha \beta$, next the aim is to find $x_i = x \pmod{p_i^{e_i}}$ for $1 \leq i \leq r$, and after that apply the CRT (Chinese remainder theorem) to calculate $x = \sum_{i=1}^{k} a_i N_i M_i \pmod{n}$, where $N_i = N/N_i$, and $M_i = N_i^{-1} \pmod{N_i}$. These calculations can be executed with $O(\log N)^2$ bit operations. Each integer x_i is found by solving the digits $l_0, l_1, ..., l_{e_{i-1}}$ in a p_i - ary representation: $x_i = l_0 + l_1 p_i + ... + l_{e_{i-1}} p_i^{e_{i-1}}$, where $0 \leq l_j \leq p_i - 1$.

Pohlig-Hellman algorithm for computing discrete logarithms

Input: A generator α of a cyclic group G of order N and an element $\beta \in G$.

Output: The discrete logarithm $x = \log_\alpha \beta$.

1. Determine the prime factorization of N: $N = p_1^{e_1} p_2^{e_2} .....p_r^{e_r}$, as $e_i \geq 1$

2. For i from 1 to r do the following:

   Calculate $x_i = l_0 + l_1 p_i + ... + l_{e_{i-1}} p_i^{e_i - 1}$, where $x_i = x \pmod{{p_i}^{e_i}}$.

   - Let $q \leftarrow p_i$ and $e \leftarrow e_i$.
   - Let $r \leftarrow l$ and $l_{-1} \leftarrow 0$.
   - Calculate $\bar{\alpha} \leftarrow \alpha^{n/q}$.
     - For j from 0 to $e-1$ do the following:
     - Calculate $\gamma \leftarrow \gamma \alpha^{l_{j-1} q^{j-1}}$ and $\bar{\beta} \leftarrow (\beta \gamma^{-1})^{n/q^{j+1}}$
     - Calculate $l_j \leftarrow \log_{\bar{\alpha}} \bar{\beta}$
   - Set $x_i \leftarrow l_0 + l_1 q + ... + l_{e-1} q^{e-1}$.

3. Apply the Chinese Remainder Theorem to find an integer x, $0 \leq x \leq N-1$, so that $x \equiv x_i \pmod{{p_i}^{e_i}}$ for $1 \leq i \leq r$.

4. Return(x).

4.6 Function field sieve

The function field sieve was introduced by Ming-Deh A. Huang and Leonard M. Adleman in 1994, see [17], for solving the discrete logarithms in $\mathbb{F}_{p^n}$ with small values of p. Notice that here the number of vertices in the multiplicative group of $\mathbb{F}_p^n$ is $N = p^n - 1$. It is quite similar to the number field sieve which we will discuss in next section 4.7, and it has a complexity of the same order $L_{p^n}[\frac{1}{3}]$ [17].

Let $\mathbb{F}_{p^n}$ denote the finite field of p^n elements, and suppose x is a generator of the multiplicative group of $\mathbb{F}_{p^n}$. We take the general principle of function field sieve from a paper [16].

Step 1: Select a subset $S = \{\gamma, ..., \gamma_{|S|}\}$ of $\mathbb{F}_{p^n} \simeq \mathbb{F}_p[t]$ which is known as the factor base (here t is an element of degree n over $\mathbb{F}_p$). Try to find relations between products of S elements

$$\sum_{(\epsilon,\gamma)\in\mathbb{Z}\times S} \epsilon \log_x \gamma = 0,$$

where x is the generator of the multiplicative subgroup of order l in $\mathbb{F}_{p^n}$. When sufficient number of such connections are known, one acquires $\log_x \gamma$ via the corresponding linear system.

Step 2: To calculate the discrete logarithm of y, which is not in S, choose random integers v until $x^v y$ is a product of elements of S. Then

$$\log_x y = -v + \sum_{(\epsilon,\gamma)\in\mathbb{Z}\times S} \epsilon \log_x \gamma \pmod{l}.$$

The method to select the factor base S is determined to individual variation. In the original Function Field Sieve, the factor base is the image of a morphism ϕ in $\mathbb{F}_{p^n}$ of the generators S_α and S_β.

The set S_α is exactly the set of $\mathbb{F}_p$-rational principal places in the rational function field $\mathbb{F}_p(t)$ whereas S_β is the set of $\mathbb{F}_p$-rational principal places in an algebraic function field.

When a random polynomial $\mu(t) \in \mathbb{F}_p$[t], this function field is defined by an absolutely irreducible bivariate polynomial

$$H(t, X) = \sum_{i=0}^{d} \sum_{j=0}^{\acute{d}} h_{i,j} X^i t^j,$$

so that $H(t, \mu(t)) = 0 \pmod{f(t)}$. The mapping ϕ from this algebraic function field to $\mathbb{F}_{p^n}$ is then defined by $X \to \mu(\mathrm{t})$. The algorithm, seeks for pairs $(r(t), s(t)) \in \mathbb{F}_p[t]^2$, where $r(t)$ and $s(t)$ are relatively prime, so that the polynomial $r(t)\mu(t)+s(t)$ corresponds to a product of irreducible polynomials in S_α, and so that the divisor associated to the function $r(t)X + s(t)$ is a sum of places in S_β. Following Adleman and Huang [3], such a pair $(r(t), s(t))$ is known "doubly smooth" since $r(t)\mu(t) + s(t)$ is smooth and $r(t)X + s(t)$ is smooth in the sense that the norm over $\mathbb{F}_p[t]$ of $r(t)X + s(t)$ is smooth [16]. The function field sieve will split n (heuristically) in expected time, where c is the constant

$$exp[(c + o(1))(\log n)^{1/3}(\log n)^{2/3}).$$

The faster algorithm for factoring integers is described in [3]. The method gives the complexity $\exp(\log N)^{\frac{1}{3}+\epsilon}$ where $\epsilon > 0$ is arbitrary. The function field sieve "is an analog of the number field sieve method originally developed for factoring integers. It is asymptotically faster than the previously known algorithms when applied to finite fields $\mathbb{F}_{p^n}$, where $p^6 \leq n$" [3].

Pohling-Hellman defined how to resolve the DL modulo the group order N [5]. In addition, Pollard's rho algorithm enable to solve the DL for small prime numbers.

The FFS is dedicated to computing discrete logarithms in a finite field $\mathbb{F}_{p^n}$, where $q = p^n$ and p is a small prime. In this case, this gives the reason to believe that ECC method may be more useful. We observed that FFS use to solve DL in $\mathbb{F}_{p^n}$ in short time comparing for DLP in field of a similar size of the form $\mathbb{F}_p$ (with p prime) and for $\mathbb{F}_{2^n}$ (with n prime), see [17].

4.7 Number field sieve

In order to compute the DL, the NFS can be implemented for $G = \mathbb{F}_p^*$, for a prime number p. The number field sieve method provides a good algorithm for DL in $\mathbb{F}_p^*$. A good presentation is found in [11].

For describing the number field sieve complexity, the following notation will be useful:

$$L_p[\nu, \delta] = \exp(\delta(\log p)^\nu \log\log p^{1-\nu})$$

The number field sieve is inspired by the index calculus methods.

The number field sieve over $\mathbb{F}_p^*$ is outlined as follows.

- Firstly, find the factors of $p-1$ by using the factorization version of the NFS [33].
- After that, find the discrete logarithm modulo each $l \mid p-1$ as follows:
  - Choose two irreducible polynomials $f_1,\ f_2 \in \mathbb{Z}[x]$ of small degree with a common root. One potential method is as follows [11]
  - Choose number $d \in \mathbb{Z}$ and $m = \lfloor p^{\frac{1}{d}} \rfloor$, then find an m-adic expansion for $p = \sum a_i m^i$
  - Select $f_1 = \sum a_i x^i$ and $f_2 = x - m$. For other methods selecting f_1 and f_2, see [11]. Notice that then $f(m) \equiv 0 \pmod{p}$, where f is a polynomial

$$f = \sum_{i=0}^{d} a_i X^i$$

- Suppose $\alpha \in \mathbb{C}$ is a root of f_1. For a set $S \subseteq \mathbb{Z} \times \mathbb{Z}$ (explained later), find representations

$$\beta^l = \prod_{(a,b)\in S} (a + b_\alpha)^{x_{a,b}}$$

and

$$o^l = t^{x_t} g \prod_{(a,b)\in S} (a + bm)^{x_{a,b}},$$

where $\beta \in \mathbb{Z}[\alpha]$, $o \in \mathbb{Z}$, $t, g \in \mathbb{Z}$.

- By slight abuse of notations, we have then

$$(o\beta^{-1})^l = t^{x_t} g,$$

and because $o\beta^{-1} = t^b$ for some b, we have $t^{bl} = t^{x_t} g$, which implies that

$$bl = x_t + \log_t g$$

in $\mathbb{F}_p^*$. Since l is a divisor of $p-1$, we have

$$\log_t g = -x_t \pmod{l}.$$

After finding the discrete logarithm for each l, apply the Chinese remainder theorem to find $k \equiv \log_t g \pmod{p-1}$. Thus, set S and the representations related to S above

remain to be explained. S should be selected so that numbers $a + bm$ and $a + b\alpha$ are smooth in the sense that their prime factors are small. Selection procedure is known as lattice sieving, see [11].

The expected running time of the algorithm is proportional to

$$L_p[\frac{1}{3}, (\frac{64}{9})^{\frac{1}{3}}] = \exp(((\frac{64}{9})^{\frac{1}{3}} + o(1))(\log p)^{\frac{1}{3}}(\log\log p)^{\frac{2}{3}}) \leq \exp(c(\log p)^{\frac{1}{3}+\varepsilon}),$$

where $c = (\frac{64}{9})^{\frac{1}{3}} = 1.9229994\ldots$ and $\epsilon > 0$ is arbitrary, see [11].

Comparison

Here $N = p - 1$, and when comparing the complexity to some power N^α (where $\alpha > 0$) we have

$$\ln \frac{e^{c(\log p)^{\frac{1}{3}+\varepsilon}}}{N^\alpha} = c(\log p)^{\frac{1}{3}+\varepsilon} - \alpha \log(p-1).$$

Due to the cube root in the first term, we can conclude that the above expression tends to $-\infty$ as $p \to \infty$. It follows that for each $\alpha > 0$,

$$e^{c(\log p)^{\frac{1}{3}+\varepsilon}} = o(N^\alpha) = o(2^{\alpha n})$$

where $n = \log_2 N$ is the representation length of $\mathbb{F}_p$ element in bits. The above estimation shows that the number field sieve provides a sub-exponential algorithm for the DLP.

4.8 On Victor Shoup's lower bound

Current knowledge on computational difficulty does not allow any absolute lower bound for the complexity of solving DL. In the additive group $\mathbb{Z}_N$ discrete logarithm problem can be solved in polynomial time

$$O(n^3) = O((\log N)^3)$$

with the Euclidian algorithm (polynomial in n where n is the bit size). Pollard's rho (works in every cyclic group) gives the complexity of $O(\sqrt{N}) = O(\sqrt{2}^n)$, which is exponential in n (bit size).

As discussed before, the number field sieve gives complexity which is subexponential, but still worse than polynomial. However, the NFS cannot be applied to any group, obviously only to $\mathbb{F}_p^*$ and its variant FFS to $\mathbb{F}_{p^n}^*$.

Victor Shoup has shown [44] that for a *generic group* (aka black box group), the complexity for computing the DL requires $\Omega(\sqrt{N})$ group operations. Hence there is a strong reason to believe that the e.g. Pollard's rho is the foremost one can achieve without knowing any details of the group structure.

As a generic (or black box) group above we mean a group where the group operations (multiplication and the inverse) are not computed algoritmically but are given via an external agent called oracle, see [44].

Chapter 5

Elliptic Curves

5.1 Background

The term "elliptic curve"here refers to a set of points in a Cartesian plane. As we will see, there are natural subsets of the elliptic curve points for which we can define a group operation. EC have no direct relation to the ellipses. EC appear in different areas of mathematics, such as cryptography, mathematical physics, number theory, and complex analysis, see [45]. Elliptic curves were not used in cryptography before 1984. The name "elliptic"itself was coined in the 19th century. The first application in cryptography is found in integer factorization method by Lenstra [4]. Since then, EC have been used for different cryptographic purposes, e.g. primality proving, and integer factorization.

A mathematical problem related to elliptic curves (as known today) was first mentioned by Diophantus. Differential, and integral calculus were developed in the end of the 17th century, and it was possible to exhibit the arc length of an ellipse as an integral. However, it was not so straightforward how to calculate the integral.

The study on the ellipse arc length typically leads to integral of form

$$F(x) = \int_0^x R(t, \sqrt{P(t)})\, dt,$$

where $P(t)$ is a polynomial, and R a rational function. It was noticed in the 19th century that it may be fruitful to study the inverse function F^{-1} instead of F; those inverse functions were named elliptic functions subsequently [37, 32].

Weierstrass $\wp$-function

Weierstrass showed that an elliptic function satisfies a differential equation $(\wp')^2 = 4\wp^3 - g_2\wp - g_3$, where g_2 and g_3 are constants. That means, the point $(\wp'(z), \wp(z))$ lies

in curve $y^2 = 4x^3 - g_2x - g_3$. In general, an elliptic curve is $y^2 = p(x)$, where $p(x)$ is a cubic polynomial with distinct roots. Weierstrass normal form $y^2 = x^3 + ax + b$ for any elliptic curve can be achieved if the field characteristic is not 2 or 3, see [21].

5.2 The addition formulas for the Curve

Diophantine method of finding new rational points in a curve is based on drawing a secant through known rational points, and finding another intersection of the line, and the curve. This gives rise to the addition law on the EC. By introducing the projective geometry, we can also introduce a special rational point "at the infinity", see [45]. The rational points in the EC together with the infinity point form an abelian group.

Definition 5.1. The following definition is from the book, see [21]. Let E be an elliptic curve and consider points P and Q on E. Then the inverse of P and the sum $P+Q$ are defined by the following principle: the sum of the three intersection points of the curve and a line is zero. The explicit presentations of the inverse and the sum are as follows:

Assume first that the field $\mathbb{F}$ has characteristic 0. Then any elliptic over $\mathbb{F}$ has form $y^2 = x^3 + ax + b$. Let $P = (x_1, y_1)$, $-P = (x_1, -y_1)$. And $Q = (x_2, y_2) \neq -P$. Sum $P + Q = (x_3, y_3)$ is then defined as $x_3 = \lambda^2 - x_1 - x_2$ and $y_3 = \lambda(x_1 - x_3) - y_1$ where

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, & \text{if} \quad \mathrm{P} \neq \mathrm{Q} \\ \frac{3x_1^2 + a}{2y_1}, & \text{if} \quad \mathrm{P} = \mathrm{Q} \end{cases}.$$

Suppose then that $\mathbb{F}_q$ is a field of characteristic 2. Then there are two types of elliptic curves. In so-called supersingular case the form is

$$y^2 + cy = x^3 + ax + b,$$

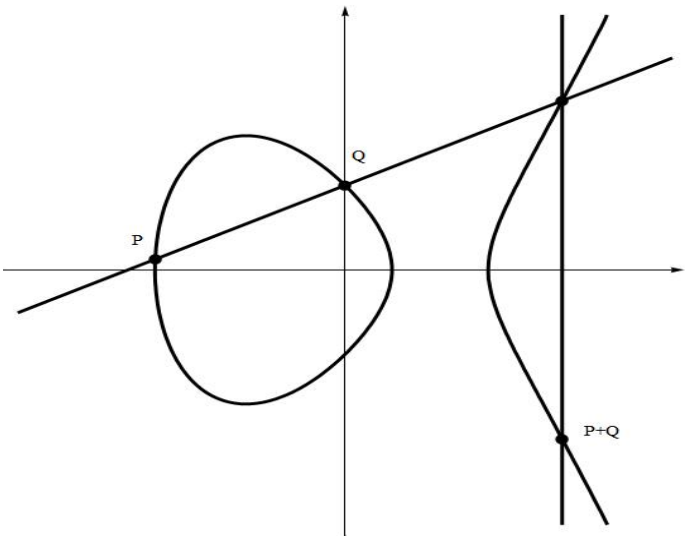
where $a, b, c \in \mathbb{F}_q, c \neq 0$, jointly with the point at ∞. The other type is

$$y^2 + xy = x^3 + ax^2 + b,$$

where $a, b \in \mathbb{F}_q, b \neq 0$, together with the point at infinity. In both cases, E is an (additive) abelian group with the point infinity serving as the identity. For these two categories the addition formulas of curves over $\mathbb{F}_{2^m}$ are explained below.

For the supersingular case, let $P = (x_1, y_1) \in E$; then $-P = (x_1, y_1 + c)$. If $Q = (x_2, y_2) \in E$ and $Q \neq -P$, $P + Q = (x_3, y_3)$, then $P + Q = (x_3, y_3)$, where

FIGURE 5.1: EC Addition



$$x_3 = \begin{cases} (\frac{y_1+y_2}{x_1+x_2})^2, & \text{if} \quad \text{P} \neq \text{Q} \\ \frac{x_1^4+a^2}{c^2}, & \text{if} \quad \text{P} = \text{Q} \end{cases}$$

and

$$y_3 = \begin{cases} \frac{y_1+y_2}{x_1+x_2}(x_1+x_3)+y_1+c, & \text{if} \quad \text{P} \neq \text{Q} \\ \frac{x_1^2+a}{c}(x_1+x_3)+y_1+c, & \text{if} \quad \text{P} = \text{Q} \end{cases}.$$

Let $P = (x_1, y_1) \in E$; thus $-P = (x_1, y_1 + x_1)$.

In the second category,

$$x_3 = \begin{cases} (\frac{y_1+y_2}{x_1+x_2}) + \frac{y_1+y_2}{x_1+x_2} + x_1 + x_2 + a, & \text{if} \quad \text{P} \neq \text{Q} \\ x_1^2 + \frac{b}{x_1^2}, & \text{if} \quad \text{P} = \text{Q} \end{cases}$$

and

$$y_3 = \begin{cases} (\frac{y_1+y_2}{x_1+x_2})(x_1+x_3)+x_3+y_1, & \text{if} \quad \text{P} \neq \text{Q} \\ x_1^2 + (x_1 + \frac{y_1}{x_1})x_3 + x_3, & \text{if} \quad \text{P} = \text{Q} \end{cases}.$$

The characteristic 3 is not treated in this thesis. The example below is from the book by Neal Koblitz, see [21].

Example 13. Consider an EC $E : y^2 = x^3+x+1$ over $\mathbb{F}_{23}$. We will see that $\#E(\mathbb{Z}_{23}) = 28$, $E(\mathbb{F}_{23})$ has a cyclic subgroup, and that a generator of $E(\mathbb{F}_{23})$ is $P = (0, 1)$. The

points in of $E(\mathbb{F}_{23})$, depicted as multiples of P, are illustrated below.

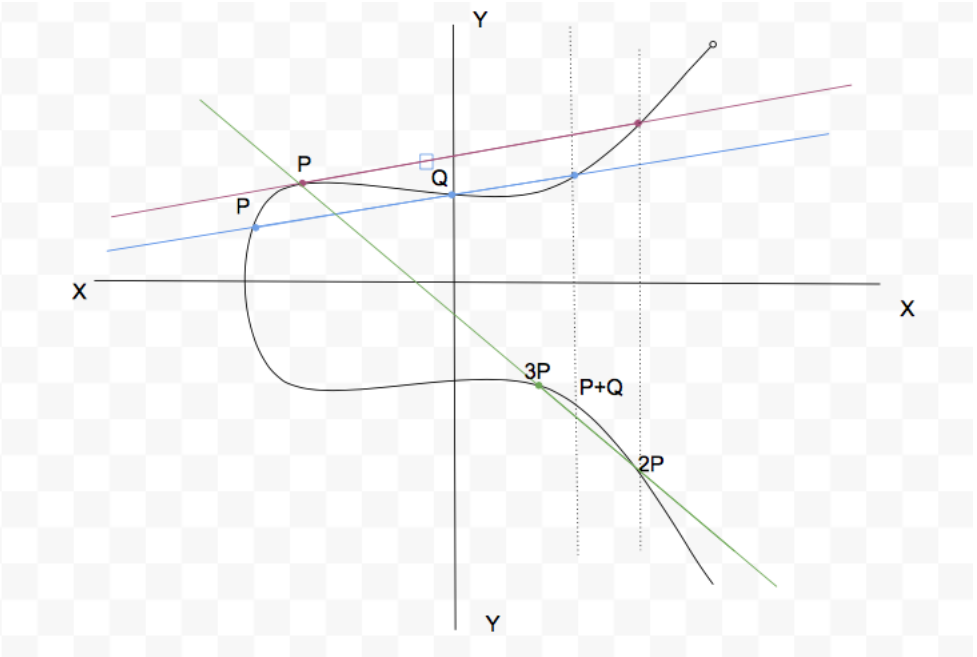
$P = (0, 1)$	$7P = (11, 3)$	$13P = (9, -7)$	$19P = (4, 5)$	$25P = (3, 10)$
$2P = (6, -4)$	$8P = (5, -4)$	$14P = (4, 0)$	$20P = (5, 4)$	$26P = (6, 4)$
$3P = (3, -10)$	$9P = (-4, -5)$	$15P = (9, 7)$	$21P = (11, -3)$	$27P = (0, -1)$
$4P = (-10, 7)$	$10P = (12, 4)$	$16P = (-6, 3$	$22P = (7, -11)$	$28P =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

Given any elliptic curve, the point multiplication is a fast operation, since it can be performed by using the repeated squaring method presented in section 3.3. For example, to reach $400P = 256P + 128P + 16P$, compute the following:

$P \to 2P$	$16P \to 32P$
$2P \to 4P$	$32P \to 64P$
$4P \to 8P$	$64P \to 128P$
$8P \to 16P$	$128P \to 256P$

The "division" problem $Q = nP$ is the same as the discrete logarithm problem.

FIGURE 5.2: ECDLP



5.3 Group of rational points

The inverse and the addition law introduced in the previous section guarantee that the rational points of an elliptic curve form an abelian group.

The rational point group of an elliptic group over a finite field is always finite (as there are only finitely many points in $\mathbb{F}_q \times \mathbb{F}_q$), but a (large) cyclic subgroup is of special interest.

Mordell showed, in 1922, that the abelian group of the rational points of elliptic curve over $\mathbb{Q}$ is finitely generated. It means that the finite "torsion group" consists of rational points having finite order. In addition, the finite number of points of infinite order generate a subgroup. Hence

$$E(\mathbb{Q}) = E_{\text{tors}} \oplus \mathbb{Z}^r,$$

where r is called rank.

Example 14. Rong-Jaye Chen illustrated an example for the EC over finite fields, see [15]. With $y^2 = x^3 + x + 1$ over $\mathbb{F}_5$. $\mid E(\mathbb{F}_5) \mid = 9$.

TABLE 5.1: EC over $\mathbb{F}_q$

x	$x^3 + x + 1$	Points	y
0	1	(0,1), (0,4)	± 1
1	3	-	-
2	1	(2,1),(2,4)	± 1
3	1	(3,1),(3,4)	± 1
4	4	(4,2),(4,3)	± 2
∞		∞	∞

5.4 Diffie-Hellman protocol on Elliptic Curves

In this section we discuss elliptic curves E over finite fields, see [49]. Assume that the $\mathbb{F}_q$ has a characteristics greater than 3. An elliptic curve E over $\mathbb{F}_q$ is then the set of all solutions $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ to an equation

$y^2 = x^3 + ax + b,$

where $a, b \in \mathbb{F}$, and $4a^3 + 27b^2 \neq 0$, together with a special point ∞ called the point at q infinity. E is known to be an abelian group with the infinity point as the identity element.

Definition 5.2. (Discrete logarithm on elliptic curves). Let P be a generating element of a subgroup G of rational points of E, and $B \in G$. The discrete logarithm problem on E is then to discover n from equality $B = nP$.

The discrete logarithm problem on EC appears more difficult than the DLP in finite fields. Moreover, the most powerful approaches in the finite fields do not seems to run on elliptic curves.

The key distribution method here is the same Diffie-Hellman as introduced before: As Alice and Bob agree on elliptic curve domain parameters on an elliptic curve E_p, where a, and b and a generator G are given. The protocol itself is the same: Alice randomly chooses a private integer a for calculating the point aG. That is sent to Bob, whereas Bob also chooses a private integer b randomly, and calculates the point abG, and sends it to Alice. Thus, both Alice and Bob calculate a mutual key $K = abG$.

The description of the algorithm is hence as follows:

1. Alice and Bob choose a and b randomly.
2. Next Alice, and Bob calculate aG and bG before sending it to each other.
3. Both Alice, and Bob calculate the shared secret k

$$K = a(bG) = a(bG) = abG$$

Example 15. Let implement Diffie-Hellman protocol using the EC. Select the generator as above, $G = (35, 7)$, Alice chooses a number a arbitrarily between 0-41, and Bob chooses a number b arbitrarily between 0-41.

- Alice choose number as her secret key $a = 3$, computes $A = 3G = 3(35, 7)$ is $A = (19, 40)$.
- Bob choose number as his secret key $b = 5$, computes $B = 5G = 5(35, 7)$ is $B = (24, 21)$.

Alice receives $B = 5G = 3(5G) = 15G$.

Bob receives $A = 3G = 5(3G) = 15G$.

Example 16. Let there be an EC over $\mathbb{F}_p$ whereas $p = 41$, and parameters $a = 7, b = 20$. Then $\triangle = -16(4a^3 + 27b^2) = -194752 = 39 \pmod{41} \neq 0$. This example is from [25].

The elliptic curve E $y^2 = x^3 + 7x + 20 \pmod{41}$. The points on the EC are depicted below:

∞, $(0, 15)$, $(0, 26)$, $(2, 1)$, $(2, 40)$, $(5, 4)$, $(5, 37)$, $(6, 14)$, $(6, 27)$, $(7, 17)$, $(7, 24)$, $(9, 19)$, $(9, 22)$, $(14, 19)$, $(14, 22)$, $(16, 13)$, $(16, 28)$, $(17, 3)$, $(17, 38)$, $(18, 19)$, $(18, 22)$, $(19, 1)$, $(19, 40)$, $(20, 1)$, $(20, 40)$, $(21, 11)$, $(21, 30)$, $(22, 11)$, $(22, 30)$, $(24, 20)$, $(24, 21)$, $(26, 5)$, $(26, 36)$, $(31, 4)$, $(31, 37)$, $(35, 7)$, $(35, 34)$, $(37, 16)$, $(37, 25)$, $(39, 11)$, $(39, 30)$

FIGURE 5.3: EC $y^2 = x^3 + 7x + 20$

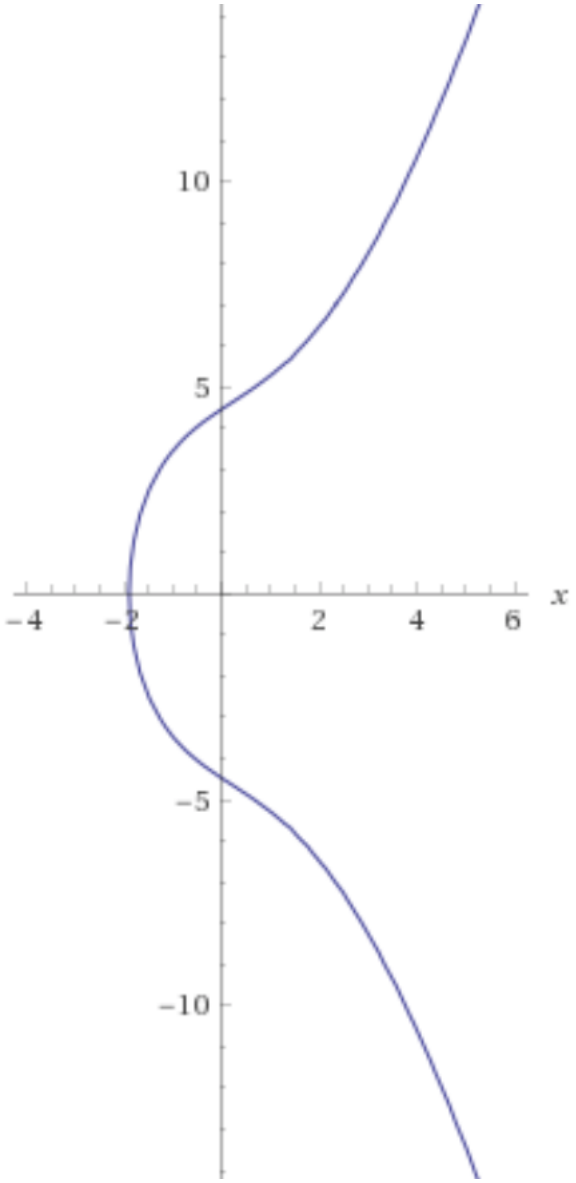
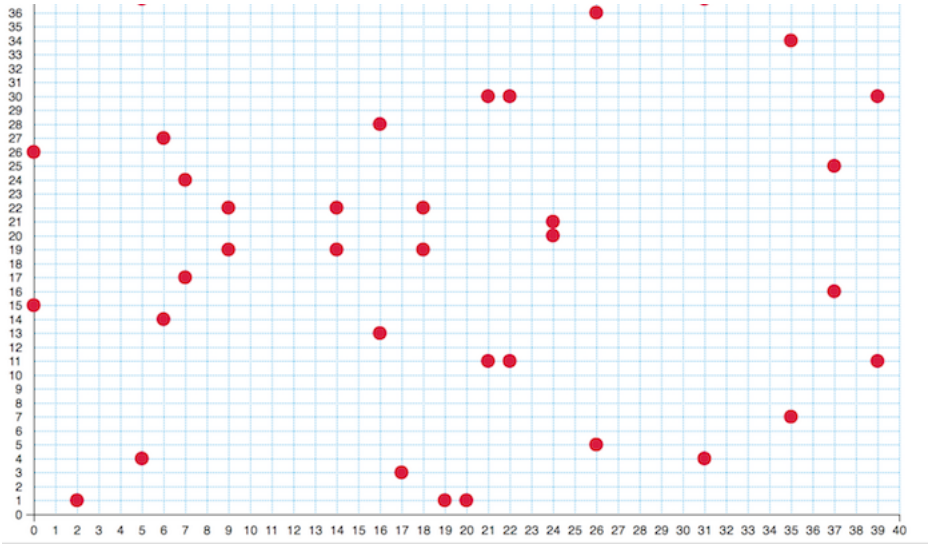


FIGURE 5.4: EC $E_{41}(7, 20)$



Chapter 6

Other uses in cryptography

Diffie-Hellman Key distribution works in an EC group that provides a secure protocol for authenticated key agreement, see [35]. In general, a large amount of Diffie-Hellman protocols have been proposed over the years and many protocols are analyzed by the following attributes:

- Known Key Security: In every run of a protocol between A and B we establish a unique secret key. A key agreement protocol must achieve its goal in front of an adversary who has learned some other session key.
- (Perfect) Forward Secrecy: This security notifies that by disclosing the long-term private key of A and B or a single user, the adversary cannot obtain the previous session keys.
- Key Compromise Impersonation: Supposing A's long-term private key is disclosed, an adversary can obviously impersonate A. However, it may be wanted that an adversary cannot impersonate any other party.
- Unknown Key-Share: Here A shares a key with B, but B believes that he has shared a key with another party $C \neq A$.
- Key Control: Neither A or B can choose the session key to be a preselected value.

Password-based authenticated key exchange (PAKE) protocols: PAKE protocols are used in many applications such as remote login, database management systems, and internet. It is based on key agreement protocols, in this case, two parties agree on a high-entropy cryptographic key using a pre-shared low entropy password, see [8]. PAKE is the best method for the client-server based applications, but not good in large-scale environments. It is costly and inefficient. However, PAKE protocols are vulnerable to

the brute-force attack for the reason that each user remembers a weak password for the key agreement protocols and authentication.

Chapter 7

Conclusion

In this thesis, the elliptic curve cryptography(ECC) is considered in detail. Different advantages of ECC like quick key generation, smaller key, signature and ciphertext along with the security of ECC which lies in the hardness of discrete logarithm having a high complexity are explained in the previous sections. Discrete logarithm problem is used as a common feature of the Diffie-Hellman key protocol. In the same way, ECC is an asymmetric key cryptosystem which is most suitable for a memory constrained device. In this thesis, we compared function-field sieve and number-field sieve and concluded both as the fastest method for computing the discrete logarithm in finite fields. The attacks we performed are as follows, Pollard's rho, baby-step giant-step, Pohlig-Hellman, function field sieve, number field sieve, and the aforesaid are techniques for solving the discrete logarithm. The comparison of attacks is demonstrated in Table 7.1.

TABLE 7.1: Attacks Compare Complexity

Attacks	Complexity
Number field sieve	$e^{c(\log p)^{\frac{1}{3}+\varepsilon}}$
Function field sieve	$exp[(c+o(1))(\log n)^{1/3}(\log n)^{2/3})$
Pollard's rho	$O(\log N)^2$
Baby-step giant step	$O(\log N)^2$
Pohling-Hellman	$O(\log N)^2$

In this thesis, we have examined different cryptographic protocols. In present situation, information security is the main problem in network communication. A large number of devices use cryptographic algorithm, while the rapid advancement in computers have made the previous algorithms insecure. With the advent of emerging multimedia applications and inter-connected devices, internet security becomes an integral part of the security setup. Quantum computing has been discussed for three decades. The idea was expressed by the Richard Feynman in 1982, and Deutsch in 1985. The most

famous algorithm was designed by Shor's in 1994 and Grover's algorithm in 1996 for polynomial time factoring, and the discrete logarithm. Quantum Computing established reformulating information and computation in the mechanical quantum framework. The mathematical description of quantum information is much more complicated than that of classical information, containing the structure of Hilbert Spaces.

The developments in quantum information have been remarkable for the last twenty years. Quantum computing is now a well-developed discipline, however, its implementation is a big challenge. Its primitives include also the secure cryptography in the quantum information processing. Researchers have been working to find out how to implement large-scale quantum computing and quantum cryptography in practice.

When quantum computers are available, there is a chance that cryptography will rely on the quantum protocols, such as protocols for digital signatures: KKNY05, GC01, and the quantum key distribution protocols BB84, BBM92, and B92. Also, the quantum zero-knowledge proof, oblivious transfer, and quantum commitment all make the present public-key cryptosystems unreliable. It is already well known due to Deutsch, that the quantum computer cannot compute functions that are not computable on a classical computer. On the other hand, Quantum computer can compute efficiently functions that cannot be computed efficiently on a classical computer. In other words, the quantum algorithm provides quicker methods than the best known classical computer algorithms.

For an algorithm based on the quantum Fourier transformation, see [42]. Some quantum Fourier transform based algorithms are known to be exponentially better than the best known classical algorithms. Anyway quantum computers allow faster searching for the solution of the computational problem. Thus cryptosystems have to keep developing different mechanism constantly. Richard Feynman suggested that quantum mechanics work in the optimization of the simulation of the quantum system with the computer. These simulations are now studied and experimented, see [42].

Cryptographic systems are mechanisms developed to protect our important information from an intruder. An intruder may use advanced technologies to breach the system security. There are a lot of significant innovations in quantum computing that are yet to be addressed. In the end, using elliptic curves we would establish effective algorithms to improve computation.

Bibliography

[1] R.R.Ahirwal, M.Ahke: Elliptic Curve Diffie-Hellman Key Exchange Algorithm for Securing Hypertext Information on Wide Area Network. International Journal of Computer Science and Information Technologies. Vol 4 No.2, pages 363–368 (2013).

[2] Yasin Fitri Bin Alias, Mohd Anuar Mat Isa, Habibah Hashim: Sieving Technique to Solve the Discrete Log Hard Problem in Diffie-Hellman Key Exchange (2015).

[3] Leonard M. Adleman and Ming-Deh A. Huang: Function Field Sieve Method for Discrete Logaithms over Finite Fields (1999).

[4] Minal Wankhede Barsagade, Dr. Suchitra Meshram: Overview of History of Elliptic Curves and its use in cryptography (2014).

[5] Razvan Barbulescu, Pierrick Gaudry and Thorsten Kleinjung: The Tower Number Field Sieve.

[6] Ezra Brown and Bruce T. Myers: Elliptic Curves from Mordell to Dio- phantus and Back. The American Mathematical Monthly 109:7, pp. 639–649 (2002).

[7] Levente Buttyan: Lecture notes History of Cryptography (2017). Electronically available at

www.crysys.hu/downloads/vihima05/2017/slidedeck_HistoryOfCryptography.pdf

[8] Majid Bayat: A Secure and efficient elliptic curve based authentication and key agreement protocol suitable for WSN (2013).

[9] Henri Cohen, Gerhard Frey, and Roberto M. Avanzi: Handbook of Elliptic and Hyperelliptic Curve Cryptography, Chapman and Hall/CRC (2006).

[10] Daniel M. Gordon: *Discrete logarithms in $GF(P)$ using the number field sieve.* SIAM Journal on Discrete Mathematics 6:1, pp. 124–138 (1993).

[11] Henrik Røst Haarberg: *The Number Field Sieve for Discrete Logarithms*. M.Sc Thesis, Norwegian University of Science and Technology (2016). https://brage.bibsys.no/xmlui/handle/11250/2394427

[12] Johan Håstad: Advanced Algorithms, lecture notes (2000). Electronically available at http://www.csc.kth.se/tcs/aalg/lectures.html

[13] Withfield Diffie and Martin E. Hellman: New directions in cryptography. IEEE Transactions on Information Theory. 22:6, pp. 644–654 (1976).

[14] Min-Shiang Hwang, Li-hua Li: A new remote user authentication scheme using smart cards (2000).

[15] Rong-Jaye Chen: Lecture Notes Elliptic curve over Finite Fields (2008). Electronically available at http://people.cs.nctu.edu.tw/~rjchen/ECC2009/10_ECoverFq.pdf

[16] Antoine Joux and Reynald Lercier: The Function Field Sieve is quite special (2013).

[17] Antoine Joux and Reynald Lercier: The Function Field Sieve in the Medium Prime Case (2006).

[18] Shmuel T. Klein: Should one always use repeated squaring for modular exponentiation? (2008).

[19] Neal Koblitz: Elliptic curve cryptosystems Mathematics of Computation 48, pp. 203–209 (1987).

[20] Neal Koblitz: A Course in Number Theory and Cryptography. Springer-Verlag (1987).

[21] Neal Koblitz: Algebraic Aspects of Cryptography. Springer (1998).

[22] Neal Koblitz, Alfred Menezes, Scott Vanstone: The state of Elliptic Curve Cryptography. Kluwer Academic Publisher (2000). Electronically available at https://link.springer.com/article/10.1023/A:1008354106356

[23] Prabhat Kushwaha: Towards the Equivalence of Diffie-Hellman Problem and Discrete Logarithm Problem for Important Elliptic Curves Used in Practice (2017).

[24] Dindayal Mahto and Dilip Kumar Yadav: RSA and ECC: Comparative Analysis. International Journal of Applied Engineering Research 12:19, pp. 9053-–9061 (2017).

[25] Nissa Mehibel: A New Approach of New Elliptic Curve Diffie-Hellman Key Exchange. Limose Laboratory University M'hamed bougara of Boumerdes Boumerdes, Algeria (2017).

[26] Alfred J Menezes, Tatsuaki Okamoto, Scott A Vanstone: Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Transactions on Information Theory 39: 1639–1646 (1993). Electronically available at

http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=259647

[27] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone: Handbook of Applied Cryptography, CRC Press (1996). Electronically available at

http://cacr.uwaterloo.ca/hac/

[28] Victor S. Miller: Uses of elliptic curves in cryptography. Lecture Notes in Computer Science 218 (proceedings of CRYPTO'85), pp. 417–426 (1986).

[29] Markus Maurer, Alfred Menezes, Edlyn Teske: Analysis of the GHS Weil Descent Attack on the ECDLP over Characteristic Two Finite Fields of Composite Degree, In: Proc. of Int'l Conference on Cryptology in India, INDOCRYPT 2001, LNCS, vol. 2247, pp. 195–213. (2001).

[30] National Institute of Standards and Technology, Secure hash standard (SHS), FIPS Publication 180–1 (1995).

Electronically available at http://ws680.nist.gov/publication/get_pdf.cfm?pub_id=910977

[31] Andrew Odlyzko: Discrete logarithms over finite fields. University of Minnesota. Electronically available at http://www.dtc.umn.edu/~odlyzko/doc/discrete.logs.hff.pdf

[32] Waghmare S P, Shikwal Simran, Nimje Shreyas, Pawar Tanvi: History of cryptography (2017).

[33] Carl Pomerance: *A Tale of Two Sieves*. Notices of the AMS 43:12, pp. 1473–1485 (1996).

[34] C. Popescu: An Identification Scheme Based on the Elliptic Curve Discrete Logarithm Problem (2000). Electronically available at

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=843508

[35] C. Popescu: A Secure Key Agreement Protocol Using Elliptic Curves. International Journal of Computers and Applications (2005).

[36] J. M. Pollard: Monte carlo methods for index computation mod p. Mathematics of Computation 32: 918–924, (1978) Electronically available at

https://pdfs.semanticscholar.org/e164/b11488ba143de0ee94887db924a0574d2361.pdf

[37] Stephen C. Pohling, Martin A. Hellman: An improved algorithm for computing logarithms over gf(p) and its cryptographic significance. IEEE Transactions on Information Theory 24: 106–110, (1978). Electronically available at

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1055817

[38] Uthayyanathan Priyatharsan, Prabath Lakmal Rupasinghe, Lain Murray: A New Elliptic Curve Cryptographic System over the Finite Fields. Sri Lanka Institute Perth West Australia (2017).

[39] Ari Renvall: Kryptografia I. Matematiikan laitos Turun yliopisto (2008).

[40] Adrian Rice and Ezra Brown: Why Ellipses Are Not Elliptic Curves. Mathematics Magazine 85:3, pp. 163–176 (2012).

[41] Martin Roetteler, Michael Naehrig, Krysta M. Svore, Kristin Lauter: Quantum Re- source Estimates for Computing Elliptic Curve Discrete Logarithms (2017). Electronically available at https://arxiv.org/abs/1706.06752

https://www.cse.iitk.ac.in/users/nitin/courses/WS2010-ref2.pdf

[42] Grzegorz Rozenberg, Thomas Bäck , Joost N. Kok: Handbook of Natural Computing. Springer (2012).

[43] A Semaev: Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p . Mathematics of Computation 67: 353–356, (1998). Electronically available at http://www.ams.org/journals/mcom/1998-67-221/S0025-5718-98-00887-4/S0025-5718-98-00887-4.pdf

[44] Victor Shoup: Lower Bounds for Discrete Logarithms and Related Problems. Advances in Cryptology EUROCRYPT 97. pp. 256–266 (1997).

[45] Joseph H. Silverman: An Introduction to the Theory of Elliptic Curves(2006).

[46] Nigel P Smart: The discrete logarithms problem on elliptic curves of trace one. preprint (1997). Electronically available at http://www.hpl.hp.com/techreports/97/HPL-97-128.pdf

[47] Michael Travis: Elliptic curve over C (2008). Electronically available at

http://www.math.uchicago.edu/~may/VIGRE/VIGRE2008/REUPapers/Travis.pdf

[48] Lawrence C. Washington: Elliptic Curves: Number Theory and Cryptography, 2nd ed. CRC Press, (2008).

[49] Mao Wenbo: Modern Cryptography: Theory and Practice.Hewlett-Packard Company. pp. 181–191, (2003).

[50] Nolan Winkler: The Discrete Log Problem and Elliptic Curve Cryptography (2014).

[51] Shi-ping Yang: Based on the agent of elliptic curve Diffie-Hellman Key Establishment Protocol (2010).