INTRODUCTION TO ELLIPTIC CURVE CRYPTOLOGY

JOSHUA KOO

1. Introduction

The evolution of cryptographic techniques can be categorized into two distinct epochs: the classical era and the modern era. A pivotal moment marking the transition between these periods was in 1977, when the introduction of groundbreaking algorithms such as RSA and the Diffie-Hellman key exchange was made. These innovations indicated a paradigm shift as they represented the first practical cryptographic schemes where security hinged on the principles of number theory. Crucially, they facilitated secure communication between two entities without the necessity of a shared secret, transforming cryptography from the cumbersome task of safeguarding secret codebooks to enabling provably secure exchanges, free from eavesdropping during key exchanges.

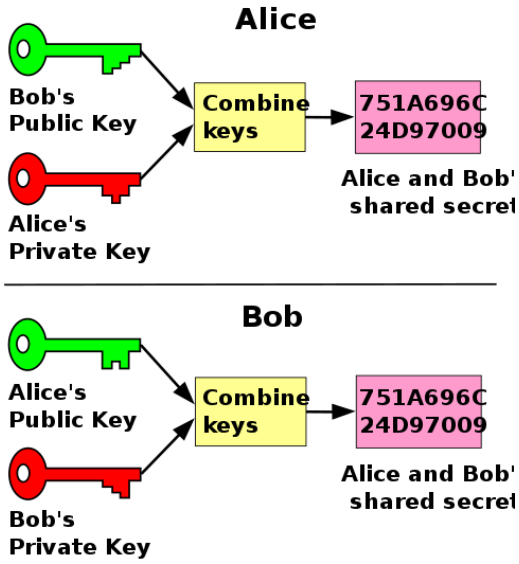
Modern cryptography, as we know it, is built upon the foundational concept that the encryption key can be public, while the decryption key remains private—a concept encapsulated in public key cryptographic systems. The essence of a functioning public key cryptographic system lies in algorithms that are easily executable in one direction yet hard to reverse. Such algorithms are termed Trapdoor Functions. The efficacy of a secure public key cryptographic system depends on the careful selection of a working Trapdoor Function, with the system's security directly proportional to the spread between the simplicity and complexity of the function's two directions. The trapdoor mechanism we will look at in this paper is Elliptic curve cryptography. Elliptic curve cryptography (ECC) is a branch of public key cryptography that leverages the mathematical properties of elliptic curves for securing communications and data. It has gained widespread adoption due to its efficiency and strong security guarantees, making it a fundamental component of modern cryptographic systems. To become familiar with EEC, let us first go over a few cryptographic techniques that will help us later understand how elliptic curves work.

1.1. Diffie-Hellman Key Exchange.

We will use the previous explanation I used in my previous expository paper on Zero-Knowledge and Interactive proofs:

One of the most basic but important concepts in cryptography is the Diffie-Hellman Key Exchange, a mathematical method of creating a cryptographic key to exchange information between two parties in a secure manner. The goal of the exchange is so that even if you know all the public info, not knowing any of the private info makes it near impossible to construct the key, while knowing at least one piece of private info allows you to do so. To see how the exchange works, let us label the two parties as Alice and Bob. Together, they will first pick some prime number p. They then pick a base g such that g is a primitive root modulo p. Then, in secret, Alice picks a number a, and similarly, Bob picks a number b. Alice then computes $g^a \pmod p$ and sends it out to the public including Bob (this is public information) and Bob similarly computes $g^b \pmod p$ and sends it to Alice. Alice now knows the number $g^b \pmod p$ so she can now compute $(g^b)^a = g^{ab} \pmod p$. In a similar manner, Bob can also compute $g^{ab} \pmod p$. Here, $g^{ab} \pmod p$ is the private key that can be used

to encrypt messages that the public does not know. This is because even if you know $g^a \pmod p$ or $g^b \pmod p$, it is very difficult to figure out $g^{ab} \pmod p$. The figure below shows a conceptual idea of how the cryptographic aspect of the system works.



1.2. **RSA.** The RSA algorithm, denoted by Rivest-Shamir-Adleman (RSA), as discussed above was pivotal in public-key cryptography. Announced in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman, the algorithm relies on the mathematical complications of factoring large composite numbers into their prime factors. The foundational security of RSA relies upon the presumed computational complexity of factoring large numbers, a challenge deemed hard for classical computers.

The RSA system has three primary steps: key generation, encryption, and decryption. The key generation process begins with the selection of two distinct prime numbers, conventionally denoted as p and q. The modulus n is subsequently determined through the multiplication of these primes, establishing the equation $n = p \cdot q$. Euler's totient function $\phi(n)$ is then calculated as $(p-1) \cdot (q-1)$, quantifying the count of positive integers less than n that are coprime with n.

Further, an encryption exponent e is chosen, a positive integer greater than 1 and less than $\phi(n)$, and is coprime to $\phi(n)$. The decryption exponent d is then computed as the modular multiplicative inverse of e modulo $\phi(n)$, satisfying the equation $(d \cdot e) \equiv 1 \pmod{\phi(n)}$. The resulting public key comprises the modulus n and the encryption exponent e, while the private key comprises n and the decryption exponent d.

The encryption phase involves representing the plaintext message m as a numeric value, ensuring $0 \leq m < n$. The encryption function $c \equiv m^e \pmod n$ transforms the message into ciphertext c. In the decryption process, which is dependent on the private key, utilizes the equation $m \equiv c^d \pmod n$. The successful decryption yields the original plaintext message.

A critical factor contributing to the security of RSA is the selection of the modulus size (n) and the prime numbers (p and q). The security of the system relies on the size of these parameters, with the vulnerability of the system increasing if the primes are small.

2. Why Elliptic Curves over these Systems?

RSA and Diffie-Hellman rely heavily on factoring, which is known to be a difficult problem. However, it has been addressed by specialized algorithms such as the Quadratic Sieve and the General Number Field Sieve. These algorithms have demonstrated moderate success. Notably, they are computationally faster than the naive approach, which involves merely guessing pairs of known primes.

As discussed in the RSA section, the efficiency of these factoring algorithms decreases as the magnitude of the numbers increases. The difference in difficulty between factoring large numbers and multiplying large numbers diminishes as the number (i.e., the key's bit length) increases. Moreover, with the expanding resources for decrypting numbers, there is a need for the size of the keys to increase at an even faster rate. However, this convergence poses a challenge as larger keys become critical to maintaining cryptographic security. Unfortunately, this escalating key size requirement is impractical for resource-constrained devices, such as mobile and low-powered devices, which are limited in their computational capabilities. The diminishing gap between the difficulty of factoring and multiplying is unsustainable in the long run in the real world.

It thus becomes evident that systems like RSA may not be the optimal cryptographic system for the future. A future-proof public key system demands a more powerful Trapdoor Function.

3. Elliptic Curves

Following RSA and Diffie-Hellman, researchers delved into alternative mathematics-driven cryptographic solutions, seeking algorithms beyond factoring that could function as effective Trapdoor Functions. In 1985, cryptographic algorithms were introduced, grounded in the branch of mathematics known as elliptic curves. Now, what exactly is an elliptic curve, and how does it work cryptographically?

Definition 1. An elliptic curve $E(F)$ is defined as a set of points in a field F that satisfies an equation of the form:

$$y^2 + a_1xy + a_2y = x^3 + a_3x^2 + a_4x + a_5$$

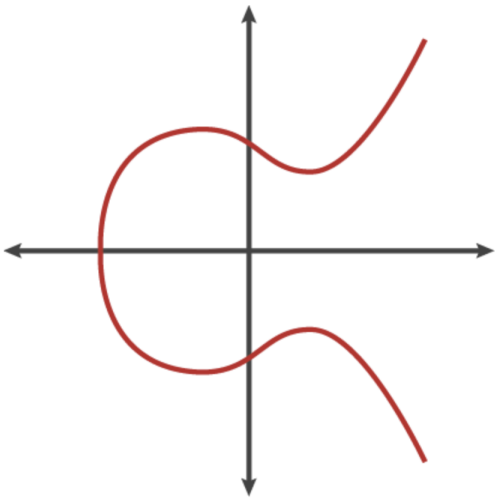
where $a_1, a_2, \ldots, a_5 \in F$. If we assume that the characteristic of the field is different than 2, this equation can be simplified to:

$$y^2 = x^3 + a_3x^2 + a_4x + a_5$$

Further simplification is possible if the characteristic of the field is also different than 3, leading to the more familiar equation, known as the Weierstrass normal form:

$$y^2 = x^3 + ax + b \quad \text{where} \quad a = a_4, \quad b = a_5.$$

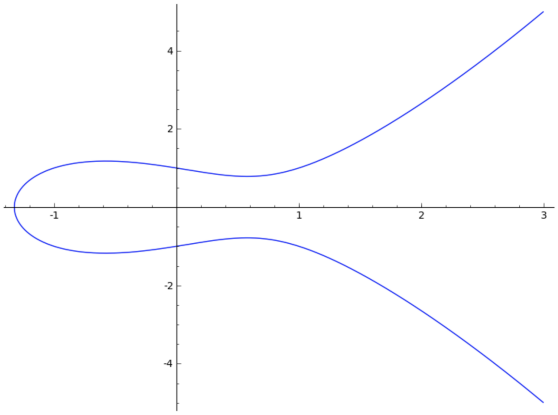
The following figure shows an elliptic curve in $\mathbb{R}^2$:



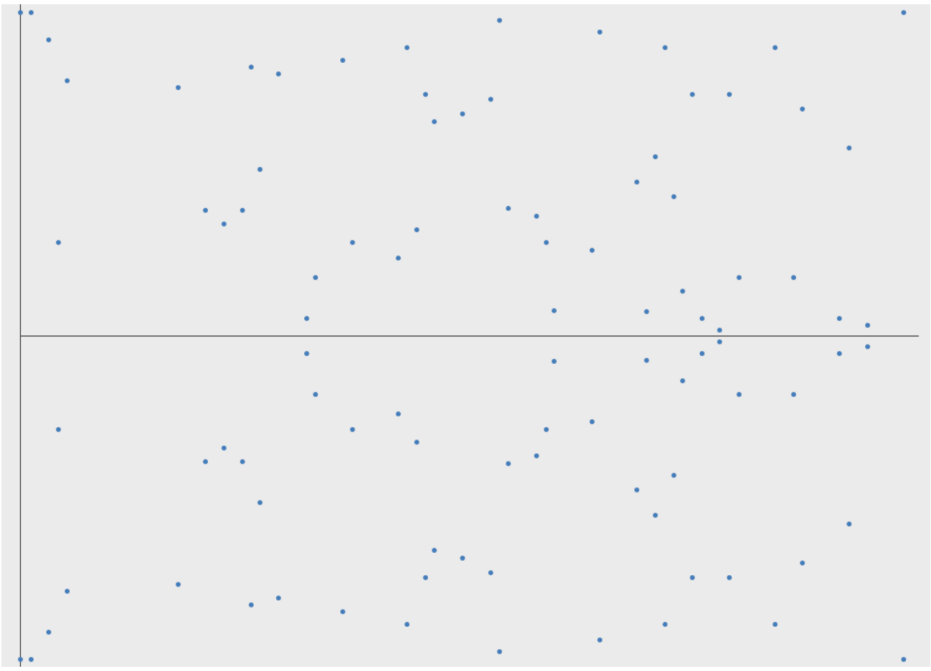
The simplified curve presented serves well for conveying and illustrating the fundamental concept of elliptic curves. However, it does not depict the typical appearance of curves employed in cryptographic applications.

To achieve this, a constraint is imposed on the values, similar to the approach in RSA. Instead of permitting any arbitrary value for the points on the curve, we confine ourselves to integers within a specified range. In the computation of the elliptic curve formula ($y^2 = x^3+ax+b$), we employ a technique of cycling through numbers when reaching the maximum. Opting for the maximum to be a prime number characterizes the elliptic curve as a prime curve, endowing it with robust cryptographic properties.

An illustrative instance of such a curve, expressed as ($y^2 = x^3 - x + 1$), is plotted for all possible integer values:



Here's the plot of the same curve with only the whole number points represented with a maximum of 97:



This doesn't quite resemble a conventional curve, yet it is one. It's as if the initial curve was folded around its edges, and only the segments that intersect with integer coordinates are shaded. The horizontal symmetry is still discernible.

Defining an elliptic curve cryptosystem involves selecting a prime number as a maximum, specifying a curve equation, and designating a public point on the curve. The private key is represented by a numerical value, denoted as "n," while the public key is derived by performing the dot product of the public point with itself "n" times. The process of computing the private key from the public key in this cryptographic system is termed the elliptic curve discrete logarithm function. Remarkably, this function serves as the Trapdoor Function we were searching for.

4. Applications

In the field of blockchain technology, elliptic curve cryptography plays a pivotal role in securing transactions and generating digital signatures. Cryptocurrencies like Bitcoin and Ethereum leverage ECC to ensure the integrity and authenticity of transactions within decentralized networks. The mathematical foundations of elliptic curves contribute to the creation of secure and efficient blockchain ecosystems.

Beyond cryptography, elliptic curves find application in error-correcting codes, particularly in the construction of algebraic geometry codes used for error detection and correction in data transmission and storage systems.

Moreover, in biometric authentication systems, elliptic curve cryptography enhances security, ensuring the protection of sensitive personal information. By providing a framework for generating secure digital signatures, ECC contributes to the development of secure biometric systems.

Euler Circle, Mountain View, CA 94040
Email address: josdreamr@gmail.com