Intuitive Interpretation of Non-Interactive Zero-Knowledge Cryptography

A jargon-free approach to understanding zk-SNARKs and zk-STARKs

Avneet Singh
Interplanetary Company UG
sshmatrix@proton.me

ABSTRACT

zk-SNARKs and zk-STARKs are relatively new concepts in cryptography, yet they are being touted as the next forefront in modern and future crypto tech. In blockchain space specifically, there is great interest in these subfields in the context of zk-Rollups to Layer 1 blockchains such as Ethereum, or as standalone decentralised ledgers with high rates of transactions per second (TPS), e.g. Aztec Network (zk-STARK), zkSync, Loopring, ZCash (zk-SNARKs) etc. Despite their great importance in cryptography, it is unfortunately difficult to understand zk-SNARKs and zk-STARKs due to limited literature and conceivably difficult mathematics conveyed through intensive jargon. This paper is an attempt to introduce zero-knowledge (zk) cryptography in an intuitive manner to garden-variety mathematicians, physicists, curious blockchain developers and perhaps even cryptographers.

INTRODUCTION

Zero-knowledge cryptography is presumably the next natural stage in cryptography's evolution toward post-quantum era. zk-SNARKs and zk-STARKs are specific implementations of zero-knowledge cryptography that are widely considered the most promising path toward post-quantum security. In order to deeply understand zero-knowledge cryptography, one must at least understand the current generation cryptographic systems such as RSA (Rivest–Shamir–Adleman) and Elliptic Curve Cryptography (ECC); this is admittingly a challenging task since the mathematics of such protocols is rather tedious and it only gets exponentially worse as one ventures in zero-knowledge protocols. Despite these challenges, it is nonetheless easier to understand at least the philosophy and intuition behind zero-knowledge protocols using the famous Alibaba Cave example [1] without requiring any prerequisite knowledge of RSA or ECC. We leave this as an exercise for the reader. In this paper, we will attempt to delve into the practical implementation of zero-knowledge protocols while retaining an intuitive understanding of the underlying mathematical processes.

> Through the course of this paper, we will leave additional comments in a grey box targeted at physicists, mathematicians and developers. These comments are anecdotes, comparisons or similariries noted across different fields that may help readers develop an intuitive understanding.

Cryptographic protocols at their core are motivated by the need to prove access to some information without necessarily revealing a part or the entirety of said information. In mathematical terms, there are several ways of achieving this functionality from an intelligently designed system.

PRIME NUMBERS

Cryptographers realised back in the day that prime numbers were one such system

that could provide such a functionality. For instance, consider two sufficiently large prime numbers **a** = 53781811 and **b** = 23252729, and their even larger product **a** × **b** = 1250573876312219. The product 1250573876312219 is a relatively difficult number to prime factorise back to **a** and **b** if both **a** and **b** are unknown. However, if either one of the two prime factors (**a** or **b**) are known, then it is straightforward to calculate the other unknown prime factor by simple division. To intuitively understand this system further, let's break it down into its principle components: we took a set of very large prime numbers of which **a** and **b** are members and defined an operation of multiplication[1] on the members of the set; such a finite field is called a Galois field. Additionally, we note that the product operation × is difficult to invert unless one of the two numbers is known; this kind of a system is usually called a trapdoor.

> In abstract mathematics, such a system is called a Group and the study of groups is called Group theory; a group is defined by a set of elements (called a Field, e.g. large prime numbers) along with the set of permitted operations between those elements (e.g. multiplication). The configuration of any trapdoor system is such that the permitted operations defined on the elements of the group are difficult to invert.

This trapdoor feature of our chosen system is essentially the backbone of all present day cryptography.

RSA CRYPTOGRAPHY

RSA protocol is one of the simplest implementations of the trapdoor feature which results in a keypair system – a public key and a private/secret key – typically utilised in conjunction for encrypting and decrypting information. The premise of the RSA protocol essentially lies in setting, i) public key as the qualitative equivalent of the product **a** × **b**, and ii) private key as the qualitative equivalent of either **a** or **b**, where **a** and **b** are restricted to a finite field of prime numbers (denoted by F_p). The security of such a system is encoded in the difficulty of prime factorising the product. It is unimportant to know the precise details of the protocol implementation in context of this paper. In nutshell, the RSA algorithm requires solving for the coefficients of Bezout's Identity using extended Euclidean algorithm [2] and employing modular arithmetic to wrap integer numbers when they fall outside the finite prime field F_p.

RSA SAFETY

While RSA is sufficiently safe to use today, it's safety will decrease over time as computational capacity of human civilisation increases. This is because the fastest and – arguably – maximally efficient algorithms capable of inverting the **a** × **b** product are iterative by construction and rely on optimised brute-forcing; Quadratic Sieve [3] and General Number Field Sieve [4] are two such well-known methods. With the advent of quantum computers, RSA algorithm's security will be definitively compromised; this is the so called SNDL problem (Save-Now-Decrypt-Later) facing the cryptography community today [5]. The core issue at hand here is that finite prime fields and the generic operation of multiplication on them – irrespective of the largeness of its elements – does not possess sufficient difficulty if the exploitor has relatively large computational resources at hand. In order to design better cryptosystems that are secure against brute-forcing, one must come with a new group with a better

[1] in addition to implicitly defining the operation of addition '+' and its inverse subtraction '−'

choice of finite field and a preferably a harder group operation which is resistant to such an attack.

RSA TO DIFFIE-HELLMAN

In very few lines, let us discuss the Diffie-Hellman Key Sharing Algorithm. Diffie-Hellman is another cryptosystem which in some sense is the intermediate step between RSA and ECC since it introduces finite fields over groups other than itself. In simple terms, this means that the group operation of product $\times$ remains the same but the group elements are now that of a finite cyclic group [6]. The non-invertible term securing the protocol in this case is of the form[2] g^a % **p**, instead of a simple product between two large prime numbers. Breaking the Diffie-Hellman method is thereby different from RSA since it involves inverting g^a % **p** instead of **a** $\times$ **b**; this is known as the discrete logarithm problem.

ELLIPTIC CURVE CRYPTOGRAPHY

The fundamental issue that leads to the breakdown of RSA algorithm is that the prime number field and the operation of multiplication is not complex enough in the face of large computational power. The effort to improve on this problem led to the advent of Elliptic Curve Cryptography (ECC). ECC functions by fixing the core illnesses in RSA, i.e. it proposes – similar to Diffie-Hellman – that instead of operating on the prime number field itself, we operate on another

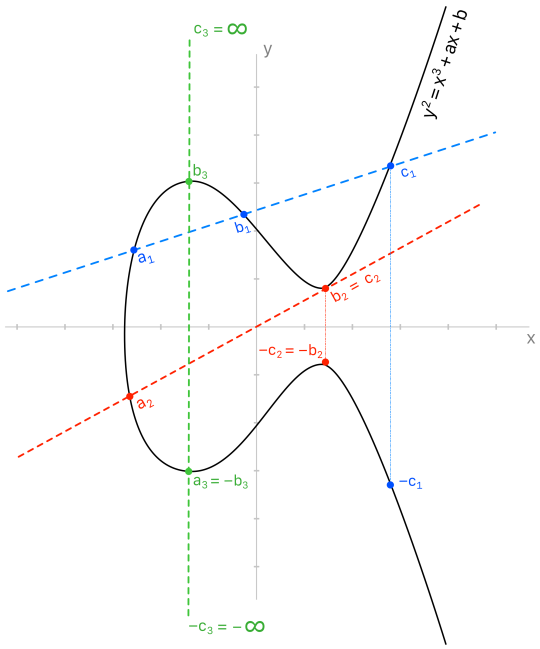

Figure 1: Elliptic curve over a continuous infinite field, aka a Lie group. We ignore the finite field contraint in this figure for simplicity.

field on which we define a newer set of operations that are much harder to invert. The premise of ECC thus lies in the introduction of finite prime fields over elliptic curves, a step up from finite cyclic groups used in Diffie-Hellman [7]. In context of cryptography, an Elliptic curve is simply the following relation between two finite prime fields (x, y),

$$y^2 = x^3 + px + q,$$
$$\text{given } 4a^3 + 27b^2 \neq 0$$

Typically, x is taken as the independent finite prime field and y is the so-called 'elliptic curve (evaluation) over finite field'; y are the elements of our desired new group. Ignoring the finite field constraint for a moment, the elliptic curve is shown in figure 1 when x is a continuous real variable. Our task now is to come up a hard-to-invert group operation on the continuous elements y of our new group; we will reinstate the finite prime field constraint afterwards.

Gigabrains realised few decades ago that the elliptic curves have some very cool properties in context of invertibile binary operations. To begin with, 1.

[2] g is called the group generator (integer), a is private key (integer) and **p** is a prime number

the elliptic curve is symmetric across the x-axis; this is a desirable property since it ensures the existence of both y and -y in our group. More importantly, 2. for any two points a and b on the elliptic curve, a line through them intersects the elliptic curve again at most - but not necessarily - 1 more point, denoted by c; the blue line in figure 1 is one such example. Note that it is possible for the line to not intersect the curve for the third time at all, and these singularities are in fact necessary for our group to function. Consider, the green line in figure 1; note that all vertical lines intersect the curve at exactly two points. It is also possible for a tangent to the curve to intersect the curve at exactly two points; see the red line in figure 1. The aforementioned gigabrains soon realised that one can use this intersectional property of a line with the elliptic curve (aka properties 1. and 2.) as the new group operation $\oplus$ such that

$$a \oplus b = -c$$

This is the same as saying that $\oplus$ operation between a and b is equivalent to taking reflection (-) of the third intersection point c across x-axis. Most literature phrases this $\oplus$ operation as the new "addition" operation on the group elements of y. However, we noted before that the third intersection c doesn't exist for all pairs a and b; in jargon-speak, we say that 'the group is not closed under the $\oplus$ operation'. We can solve this closeness problem by explicitely including the infinity element ∞ to the group. Once we have access to ∞, all intersections similar to the green line are assumed to intersect the curve again at $c = \infty$, such that

$$a \oplus \infty = a$$

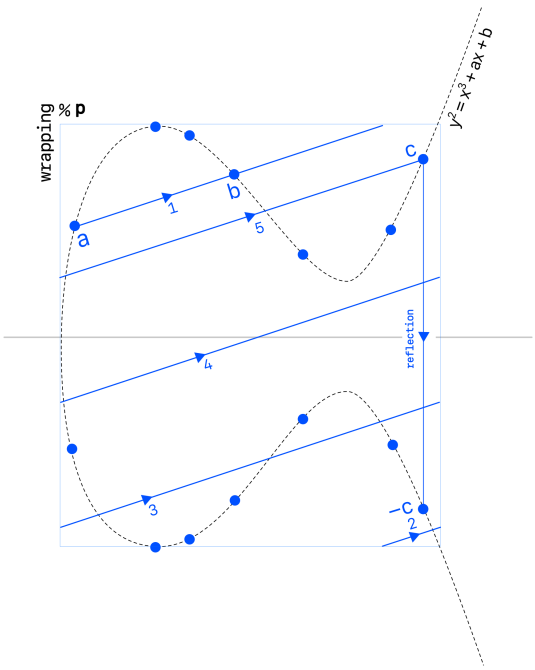

Figure 2: Elliptic curve over a finite prime field modulo **p**; the effect of the modulo operator % is to "wrap" the out-of-range values to within range. In the above example, it takes 5 iterations and 1 reflection to calculate -c.

Note that ∞ acts like the identity element $\hat{I}$ of our new group since it satisfies $a \oplus \hat{I} = a$. Lastly, the third case of the red line is not difficult since it is simply a special case of the blue line when b = c. Now we have all the ingredients required to construct a robust inverse problem on our new group y with the permitted operation $\oplus$. We reinstate the requirement of evaluating the elliptic curve on a finite prime field by explicitely defining a finite field range and wrapping the out-of-range evaluations over the elliptic curve with the modulo operator % **p**. The final form of the $\oplus$ operation is shown in figure 2.

In abstract mathematics, such a system is called a Group and the study of groups is called Group theory; a group is defined by a set of elements (called a Field, e.g. large prime numbers) along with the set of permitted operations between those elements (e.g. multiplication). The configuration of any trapdoor system is such that the permitted operations defined on the elements of the group are difficult to invert.

.

REFERENCES

[1] Quisquater, JJ. et al. (1990), How to Explain Zero-Knowledge Protocols to Your Children, Lecture Notes in Computer Science, Vol. 435, Springer, New York, NY
[2] Extended Euclidean Algorithm
[3] Quadratic Sieve Algorithm
[4] General Number Field Sieve Algorithm
[5] SNDL Problem
[6] Diffie-Hellman Key Exchange
[7] Elliptic Curve Cryptography

METADATA

Github:
Contracts:
Source:
SHA-1 Checksum:
Date: July 4, 2023