

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 admittedly a challenging task since the mathematics of such protocols is rather tedious and it only gets exponentially worse as one ventures into 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 similarities 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 $a \times b = 1250573876312219$. 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 multiplication¹ on the members of the set. Additionally, we note that the product operation \times 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 $a \times b$, 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 $a \times b$ 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].

RSA TO DIFFIE-HELLMAN

Diffie-Hellman Key Sharing Algorithm 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

¹in addition to implicitly defining the operation of addition '+' and its inverse subtraction '-'

group operations of addition and multiplication remain 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 g^a modulo p ,² 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 modulo 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 field on which we define a newer set of operations that are much harder to invert. Enter Elliptic curves. In context of cryptography, an Elliptic curve is simply the following relation between two finite prime fields (x, y) ,





$$y^2 = x^3 + px + q.$$

The premise of ECC lies in the introduction of finite prime fields over elliptic curves, a step up from finite cyclic groups used in Diffie-Hellman [7]. Elliptic curves possess several highly desirable properties that make them an appropriate choice.

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 3, 2023

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