Introduction to Zero Knowledge Proofs

Introductory Maths

Numbers

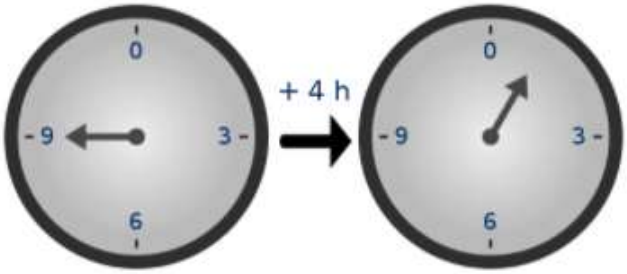
The set of Integers is denoted by $\mathbb{Z}$ e.g. $\{\cdots,-4,-3,-2,-1,0,1,2,3,4,\cdots\}$
The set of Rational Numbers is denoted by $\mathbb{Q}$ e.g. $\{\ldots 1, \frac{3}{2}, 2, \frac{22}{7} \ldots\}$
The set of Real Numbers is denoted by $\mathbb{R}$ e.g. $\{2, -4, 613, \pi, \sqrt{2}, \ldots\}$

Fields are denoted by $\mathbb{F}$, if they are a finite field or $\mathbb{K}$ for a field of real or complex numbers we also use $\mathbb{Z}_p^*$ to represent a finite field of integers mod prime p with multiplicative inverses.

We use finite fields for cryptography, because elements have “short”, exact representations and useful properties.

Modular Arithmetic

See this introduction



Because of how the numbers "wrap around", modular arithmetic is sometimes called "clock math"

When we write n mod k we mean simply the remainder when n is divided by k. Thus
25 mod 3 = 1
15 mod 4 = 3
The remainder should be positive.

Group Theory

Simply put a group is a set of elements {a,b,c,...} plus a binary operation, here we represent this as •
To be considered a group this combination needs to have certain properties

1. Closure

2. Associativity
   For all a, b and c in G, (a • b) • c = a • (b • c)
3. Identity element
   There exists an element e in G such that, for every element a in G, the equation e • a = a • e = a holds. Such an element is unique and thus one speaks of the identity element.
4. Inverse element
   For each a in G, there exists an element b in G, commonly denoted a^{-1} (or –a, if the operation is denoted "+"), such that a • b = b • a = e, where e is the identity element.

Fields

A field is a set of say Integers together with two operations called addition and multiplication.
One example of a field is the Real Numbers under addition and multiplication, another is a set of Integers mod a prime number with addition and multiplication.
The field operations are required to satisfy the following field axioms. In these axioms, a, b and c are arbitrary elements of the field $\mathbb{F}$.

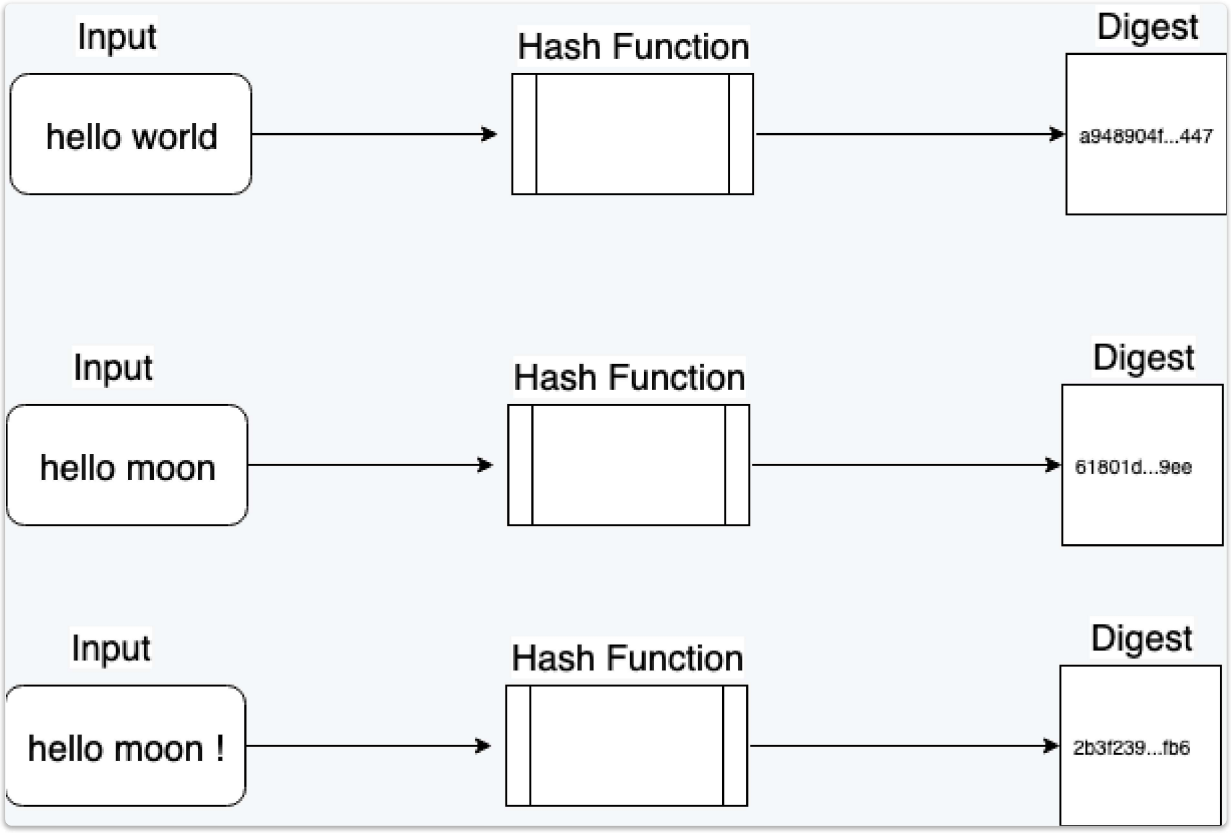
1. Associativity of addition and multiplication: a + (b + c) = (a + b) + c and a · (b · c) = (a · b) · c.
2. Commutativity of addition and multiplication: a + b = b + a and a · b = b · a.
3. Additive and multiplicative identity: there exist two different elements 0 and 1 in $\mathbb{F}$ such that a + 0 = a and a · 1 = a.
4. Additive inverses: for every a in F, there exists an element in F, denoted –a, called the additive inverse of a, such that a + (–a) = 0.
5. Multiplicative inverses: for every a ≠ 0 in F, there exists an element in F, denoted by a^{-1}, called the multiplicative inverse of a, such that a ·a^{-1} = 1.
6. Distributivity of multiplication over addition: a · (b + c) = (a · b) + (a · c).

To try out operations on finite fields, see https://asecuritysite.com/encryption/finite

---

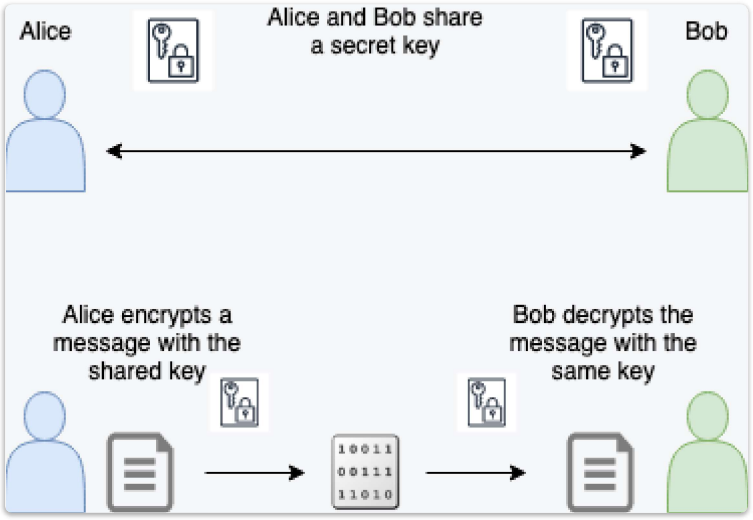
Cryptography Background

Hash Functions



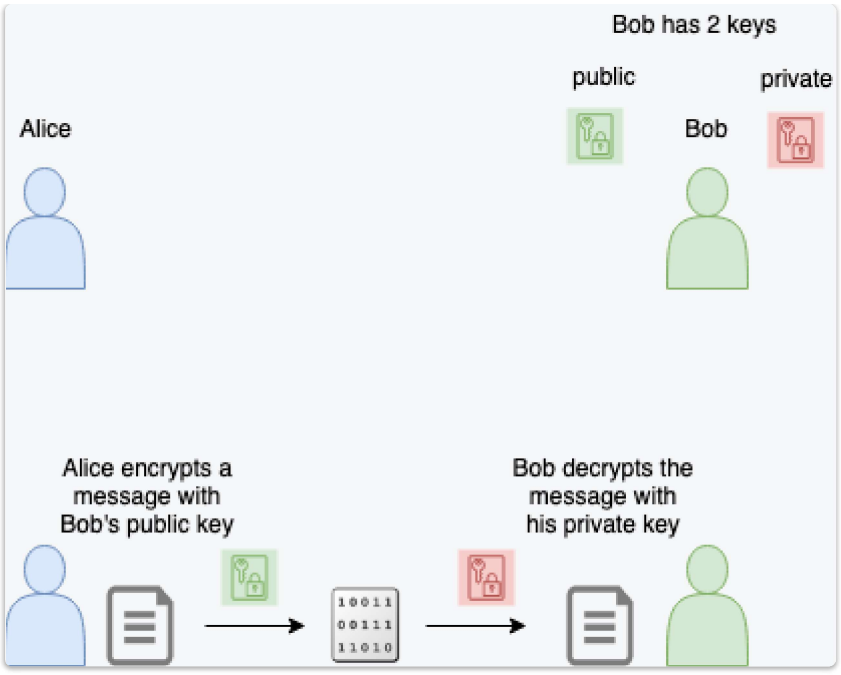
Encryption

SYMMETRIC ENCRYPTION

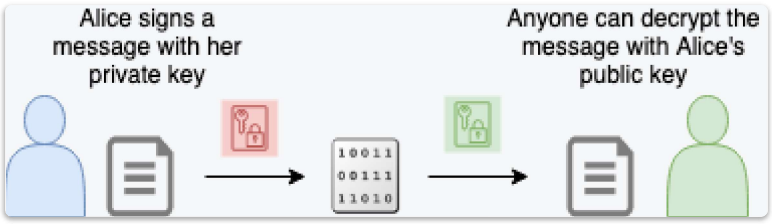


ASYMMETRIC ENCRYPTION

Sending a secret message



Proving ownership (knowledge of) of a private key



Intuitive grasp of Zero Knowledge Proofs

Introduction

It is difficult to find zero knowledge resources that avoid the extremes of either over simplyfying the subject, or presenting so much mathematical detail that the reader gets bogged down and loses interest.

We start with some examples to show how zero knowledge proofs can be achieved.

What is a zero knowledge proof

A loose definition
It is a proof that there exists or that we know something, plus a zero knowledge aspect, that is the person verifying the proof only gains one piece of information - that the proof is valid or invalid.

Actors in a Zero Knowledge Proof System

- Creator - optional, maybe combined with the prover
- Prover - I will call her Peggy
- Verifier - I will call him Victor

Examples to give an Intuitive grasp of zero-knowledge proofs

1. Colour blind verifier
   This is an interactive proof showing that the prover can distinguish between a red and a green billiard ball, whereas the verifier cannot distinguish them.

```
• The prover wants to show the verifier that they have different colours
but does not want him to learn which is red and which is green.

• Step 1: The verifier takes the balls, each one in each hand, holds them
in front of the prover and then hides them behind his back. Then, with
probability 1/2 either swaps them (at most once) or keeps them as they
are. Finally, he brings them out in front.

• Step 2: The prover has to say the verifier switched them or not.

• Step 3: Since they have different colours, the prover can always say
whether they were switched or not.
But, if they were identical (the verifier is inclined to believe that),
the prover would be wrong with probability 1/2.

• Finally, to convince the verifier with very high probability, the prover
could repeat Step 1 to Step 3 k times to reduce the probability of the
prover being successful by chance to a extremely small amount.
```

2. Wheres Wally
   Based on the pictures of crowds where Wally is distinctivly dressed, the aim being to find him within a sea of similar people.
   The proof procedes as follows :
   Imagine the Peggy has found Wally in the picture and wants to prove this fact to Victor, however if she just shows him, Victor is liable to cheat and claim he also found Wally.
   In order to prove to Victor that she has indeed found Wally, without giving away his location in the picture

```
1. Peggy cuts a hole in a (very) large sheet of paper, the hole should be
the exact shape of Wally in the underlying picture.
2. Peggy places the paper sheet over the original picture, so that the
location of the picture beneath the paper is obscured.
3. Victor can then see throught he hole that Wally has indeed been found,
but since the alignment with the underlying picture cannot be seen, he
doesn't gain any information about the location of Wally.
```

Quote from Vitalik Buterin
"You can make a proof for the statement "I know a secret number such that if you take the word ‘cow', add the number to the end, and SHA256 hash it 100 million times, the output starts with `0x57d00485aa`". The verifier can verify the proof far more quickly than it would take for them to run 100 million hashes themselves, and the proof would also not reveal what the secret number is."

Changes have occurred because of

- Improvements to the cryptographic primitives (improved curves or hash functions for example)
- A fundamental change to the approach to zero knowledge

From [Matthew Green](https://blog.cryptographyengineering.com/2014/11/27/zero-knowledge-proofs-illustrated-primer/)

Prior to Goldwasser et al., most work in this area focused the soundness of the proof system. That is, it considered the case where a malicious Prover attempts to 'trick' a Verifier into believing a false statement. What Goldwasser, Micali and Rackoff did was to turn this problem on its head. Instead of worrying only about the Prover, they asked: what happens if you don't trust the Verifier?

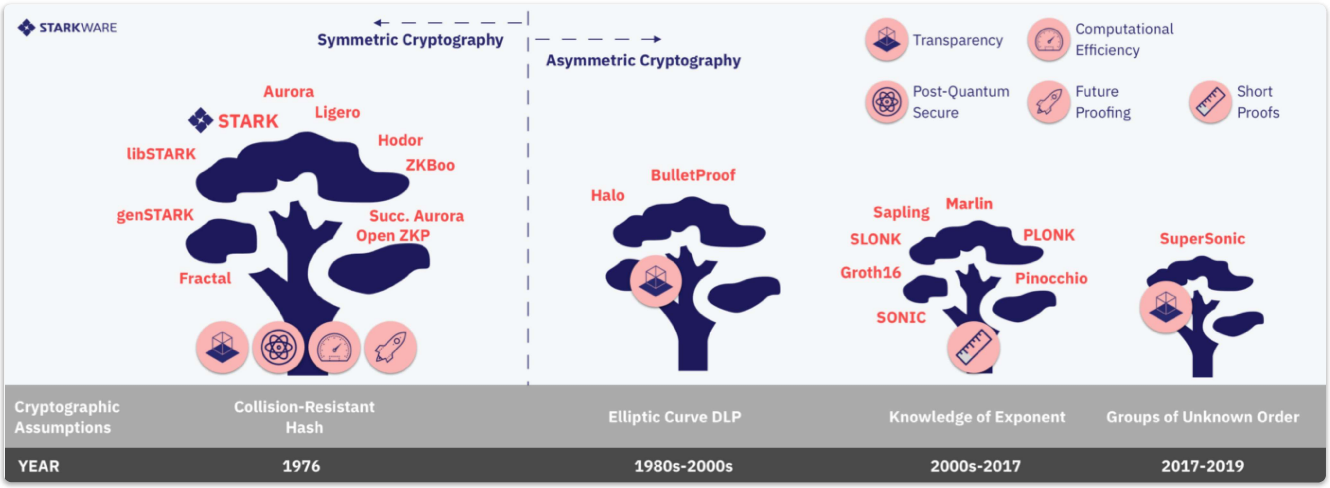
EARLY PAPERS

1984 : Goldwasser, Micali and Rackoff - Probabilistic Encryption.
1989 : Goldwasser, Micali and Rackoff - The Knowledge Complexity of Interactive Proof Systems

LATER PAPERS

2016 : Jens Groth - On the Size of Pairing-based Non-interactive Arguments

ZKP ECOSYSTEM



from
The Cambrian Explosion

Types of ZK System

There are many different ways to create zero knowledge proofs, the 2 main technologies are

1. ZKSNARKS - These have small proofs but require an initial setup and are not quantum resistant.
2. ZKSTARKS ((Scalable Transparent ARguments of Knowledge) - These have larger proofs, but are simpler to implement and are resistant to attacks using quantum computers.

ZKSTARKS were developed by Starkware and are the type used on Starknet.

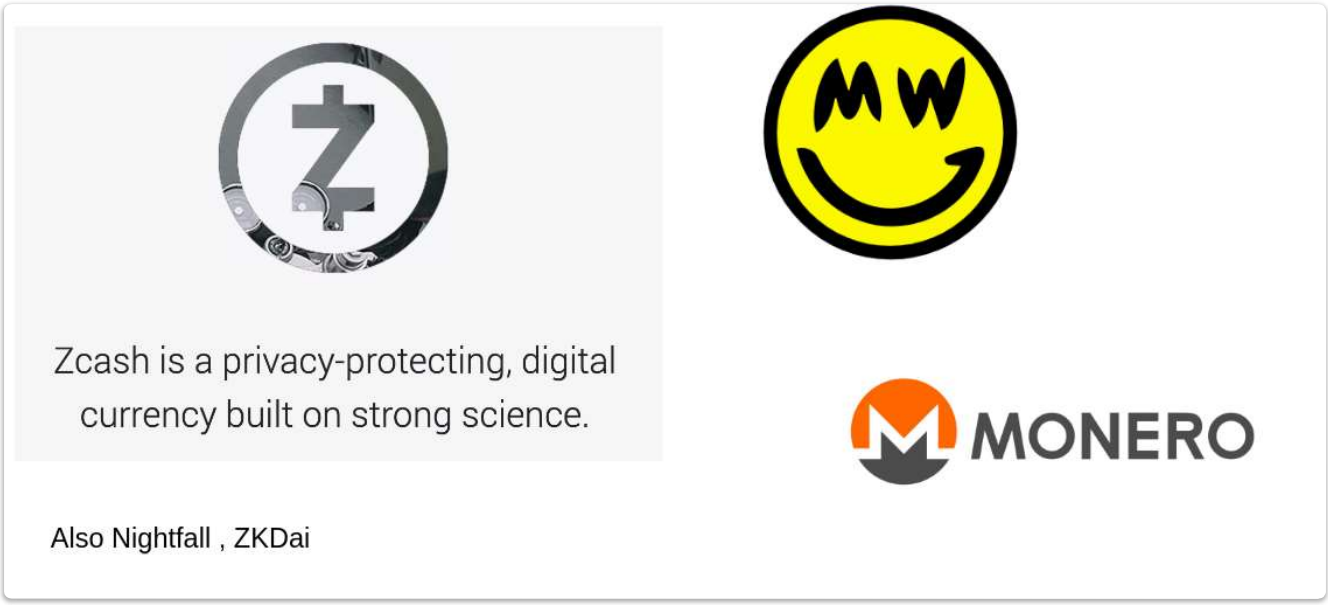
PROVING COMPUTATION WITH STARKS

A computation is a set of steps, and this can be represented as a set of polynomials - the Algebraic Intermediate Representation (AIR).
A Stark proof is essentially proving that the sequence of steps, or trace has been done correctly.

Cairo pulls together multiple AIRS to provide arbitrary computation.
According to this excellent blog, the name comes from:
a CPU built from AIRs (CPU-AIR, Oh nice –> CAIRO).

Zero Knowledge Proof Use Cases

Privacy preserving cryptocurrencies



Blockchain Scalability

For example
Rollups on Ethereum

"The scalability of ZK rollup will increase by up to 4x, pushing theoretical max TPS of such systems well over 1000." - Vitalik

From Starkware article

"Layer 2 (L2) offers dApp developers a computational greenfield, free of the gas glass ceiling. We believe that the vast majority of dApps will be L2-native within a couple of years: they will have been built from the ground up on L2 to benefit from this computational degree of freedom."

Rollups

Rollups are solutions that have

- transaction execution outside layer 1
- data or proof of transactions is on layer 1
- a rollup smart contract in layer 1 that can enforce correct transaction execution on layer 2 by using the transaction data on layer 1

The main chain holds funds and commitments to the side chains
The side chain holds state and performs execution
There needs to be some proof, either a fraud proof (Optimistic) or a validity proof (zk)

Rollups require "operators" to stake a bond in the rollup contract. This incentivises operators to verify and execute transactions correctly.