Introduction to Digital Cash

Emily Willson

Women and Mathematics Program

May 24, 2018

The Evolution of Digital Cash

How did we get to Bitcoin?

1983

Researcher David Chaum publishes a paper on digital cash. Founds company DigiCash, which folds in 1998.

1996

E-gold, the first digital currency, founded. It was backed by gold. Dies in Dot-com bubble.

2005

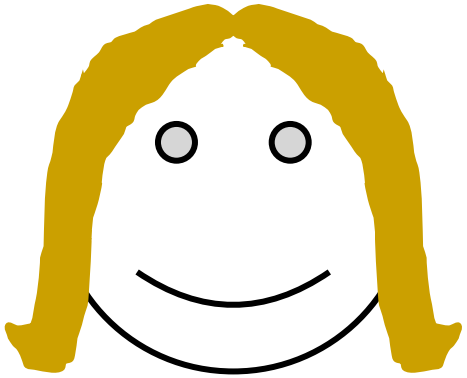
Tencent QQ introduces Q coins on platform in China, briefly destabilizing the yuan.

2009

Author Satoshi Nakamoto publishes Bitcoin paper, proposing a decentralization digital currency system.

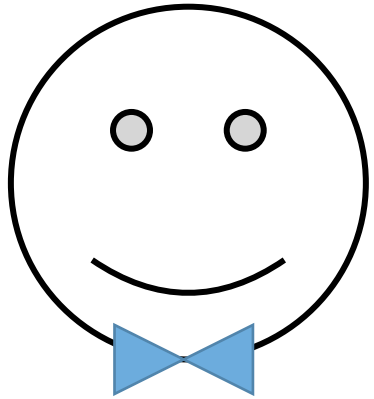
The Basics of Bitcoin

How Bitcoin Works



Alice

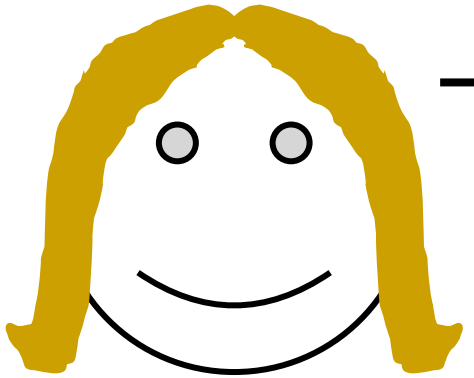
Alice wants to pay Bob 5 BTC.



Bob

How Bitcoin Works

Participants

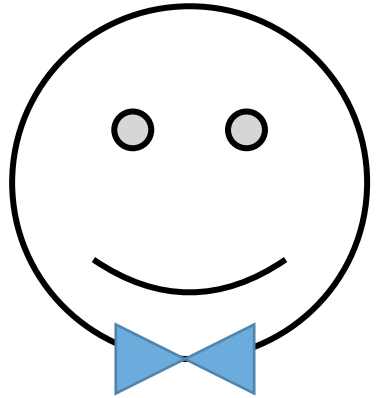


Alice

Address: *1slkj53ce58ck1359*

pays

Address: *1abck50spk59sc20*

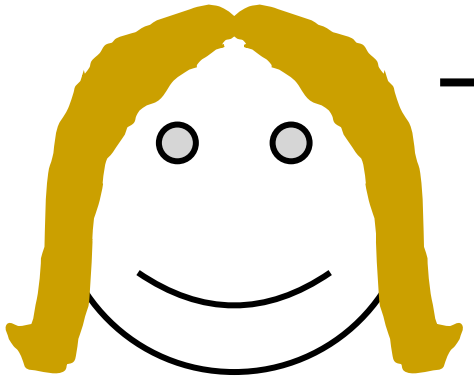


Bob

5 BTC

How Bitcoin Works

Participants

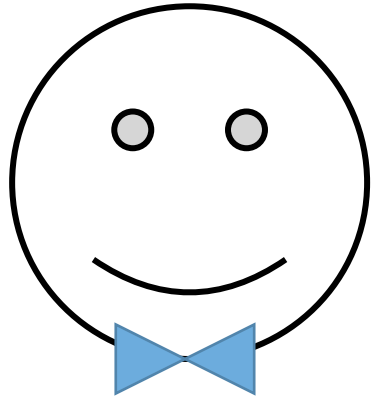


Alice

Address: *1slkj53ce58ck1359*

pays

Address: *1abck50spk59sc20*

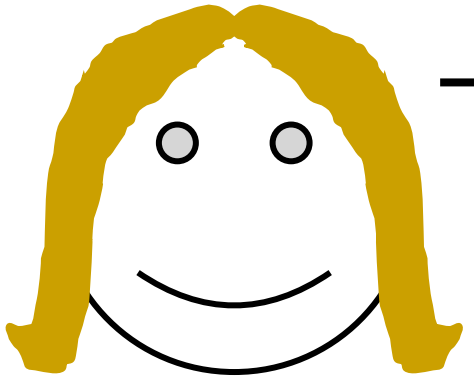


Bob

5 BTC

How Bitcoin Works

Participants

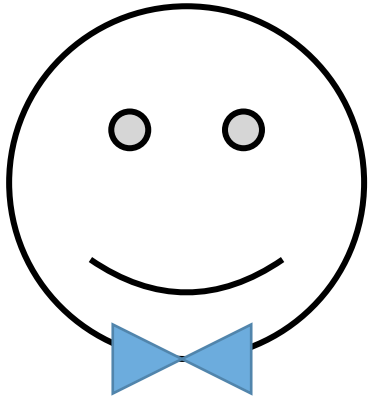


Alice

Address: *1slkj53ce58ck1359*

pays

Address: *1abck50spk59sc20*

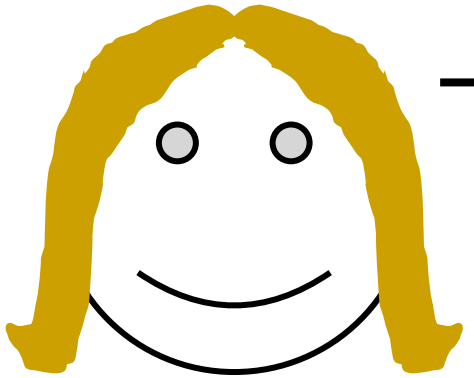


Bob

5 BTC

How Bitcoin Works

Participants

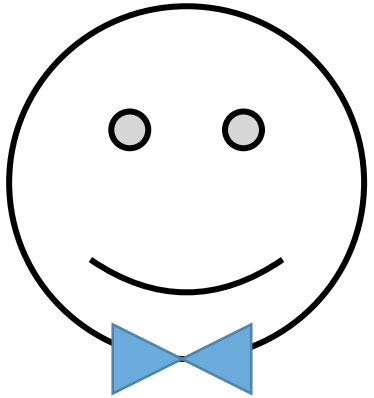


Alice

Address: *1slkj53ce58ck1359*

pays

Address: *1abck50spk59sc20*



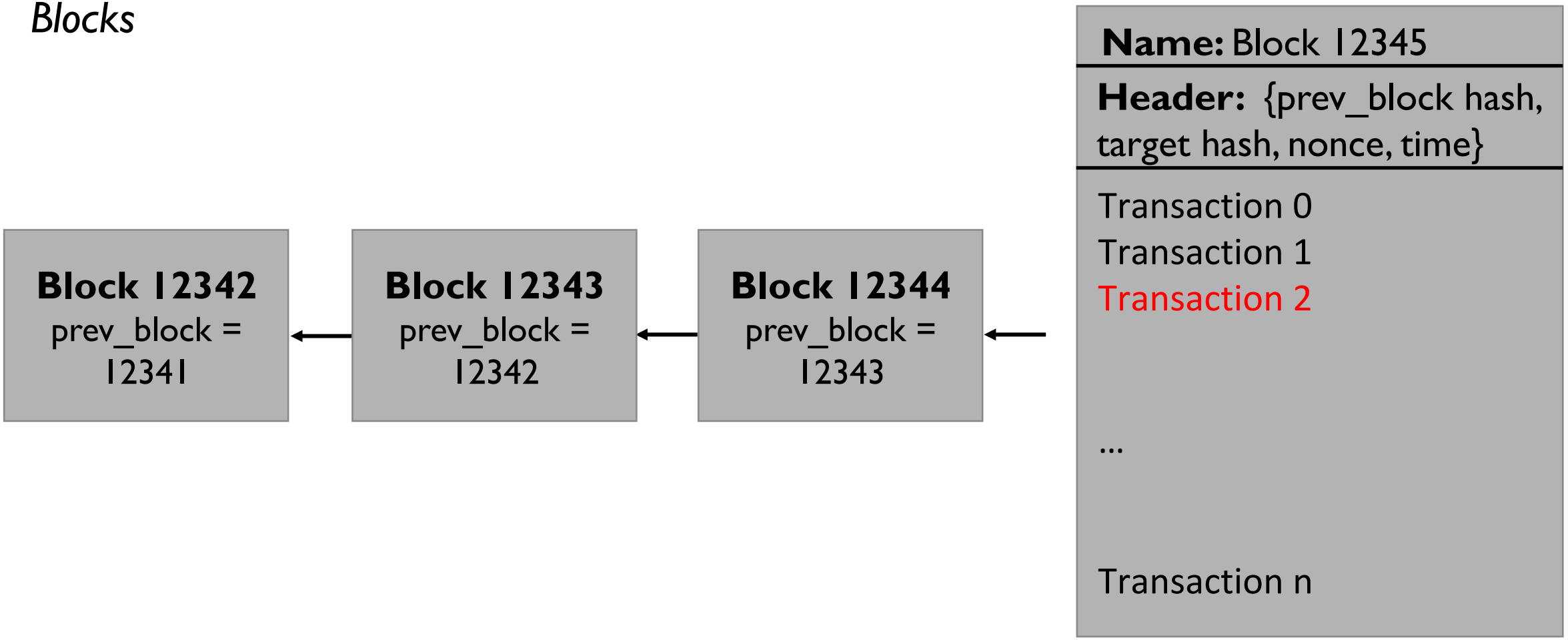
Bob

5 BTC

Transaction 2

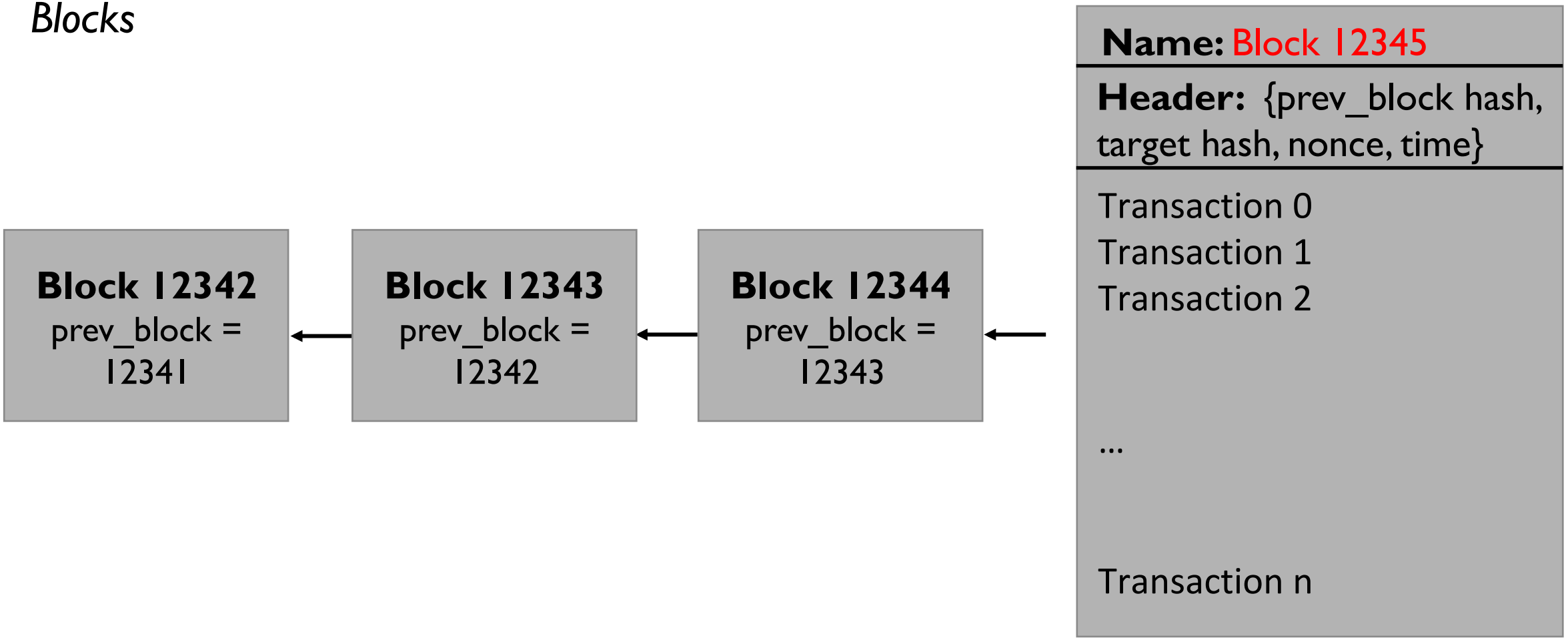
How Bitcoin Works

Blocks



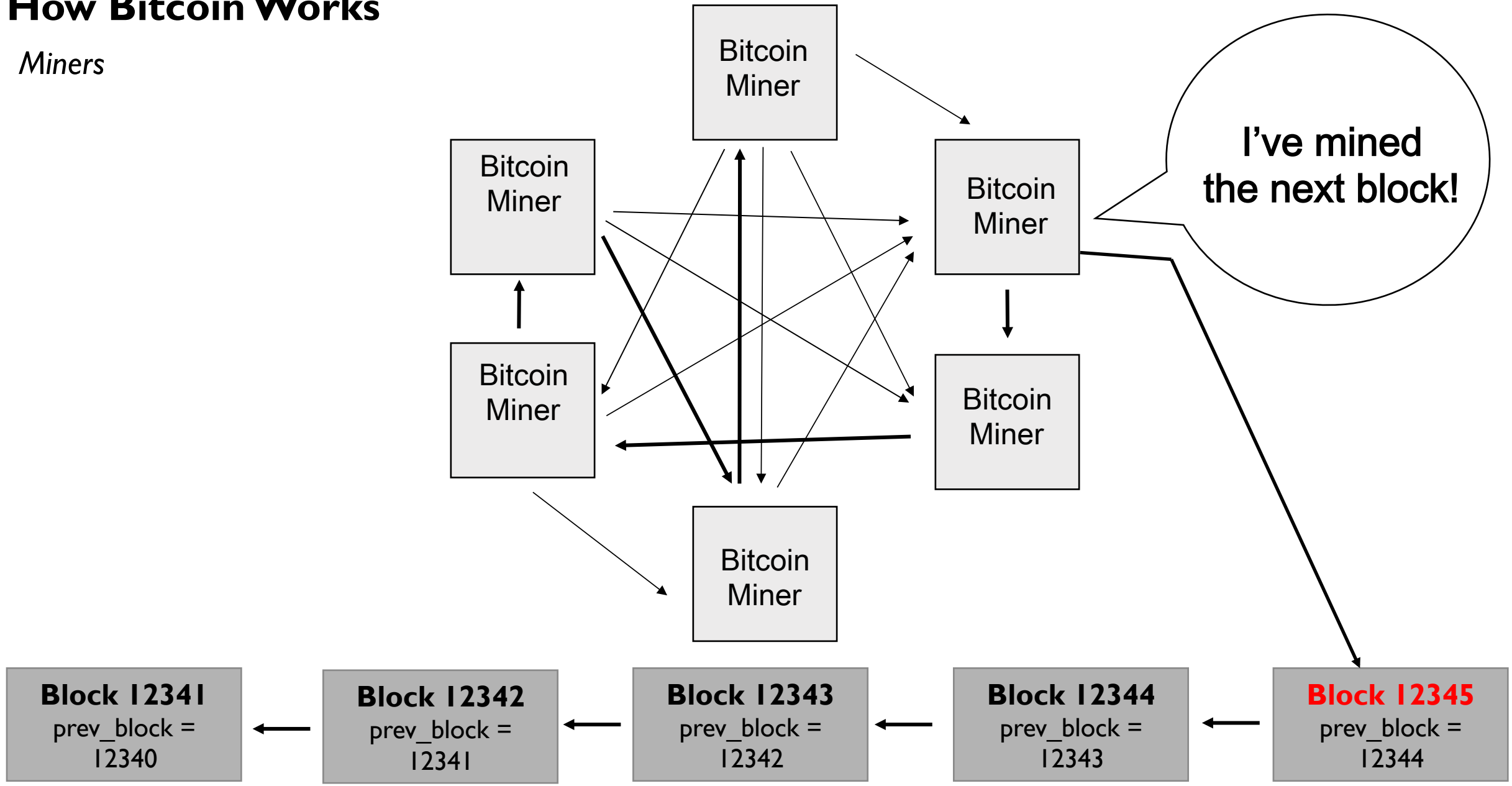
How Bitcoin Works

Blocks



How Bitcoin Works

Miners



How Bitcoin Works

Proof of Work

while no one else has mined a new block:

blockhash = **SHA256**{ all valid transactions you see + previous block information + nonce}

if blockhash < target threshold*:
    you win!
    broadcast new block to peers
    receive 12.5 BTC
    start mining the next block
else:
    increment nonce
    return to top

Examples

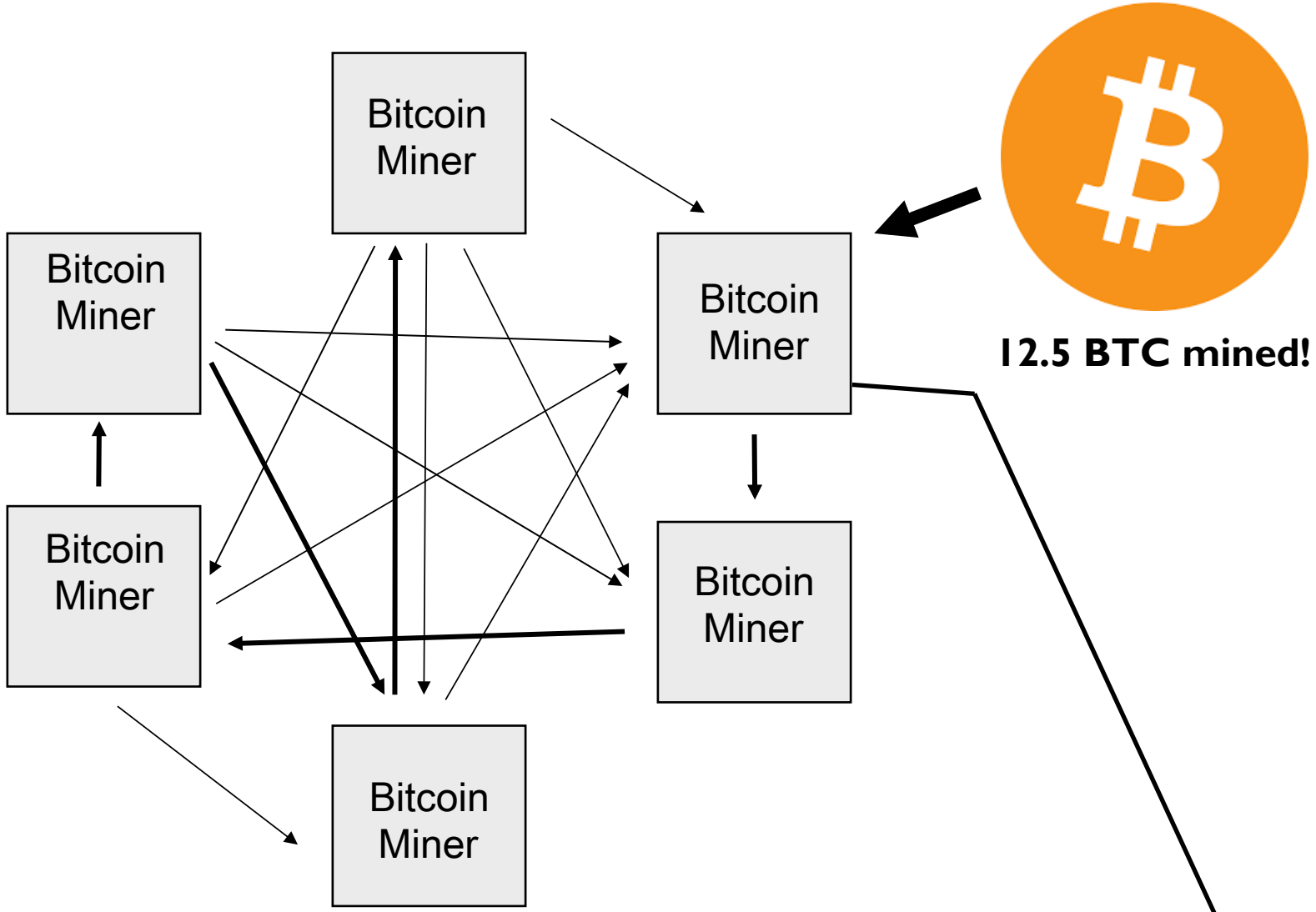
Blockhash: 0x0000005
Target Threshold: 0x0003245

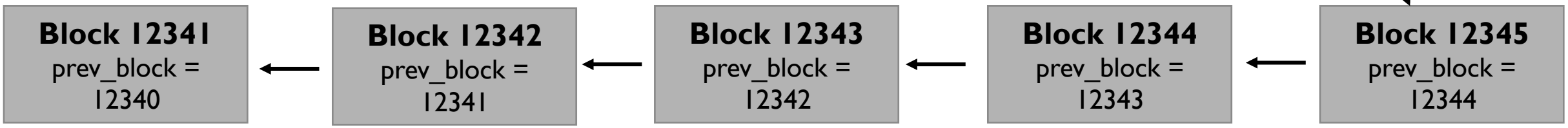
Blockhash: 0x0115251
Target Threshold: 0x0001523

* target threshold = a certain leading number of 0s in hash (i.e. 0x00000000000000000000abcde)

How Bitcoin Works

Miners





Where’s the crypt?

Case Study: Zcash

What's all this fuss about zk-SNARKs?*

Zcash: The Basics

Designed to have better **scalability**, **efficiency**, **democracy**, and **secrecy** than Bitcoin.

- **Scalability:** larger block sizes and more frequent block mining.
- **Efficiency:** less compute-intensive proof of work saves electricity.
- **Democracy:** memory-intensive proof of work avoids "elite mining."
- **Secrecy:** shielded transactions designed to provide true blockchain anonymity.

zk-SNARKs: Definitions

Snark: fictional animal species featured in Lewis Carroll's poetry.

zk-SNARK: zero-knowledge succinct non-interactive argument of knowledge.

zk-SNARKs: Definitions

zk-SNARKs are used to verify
shielded transactions in Zcash.

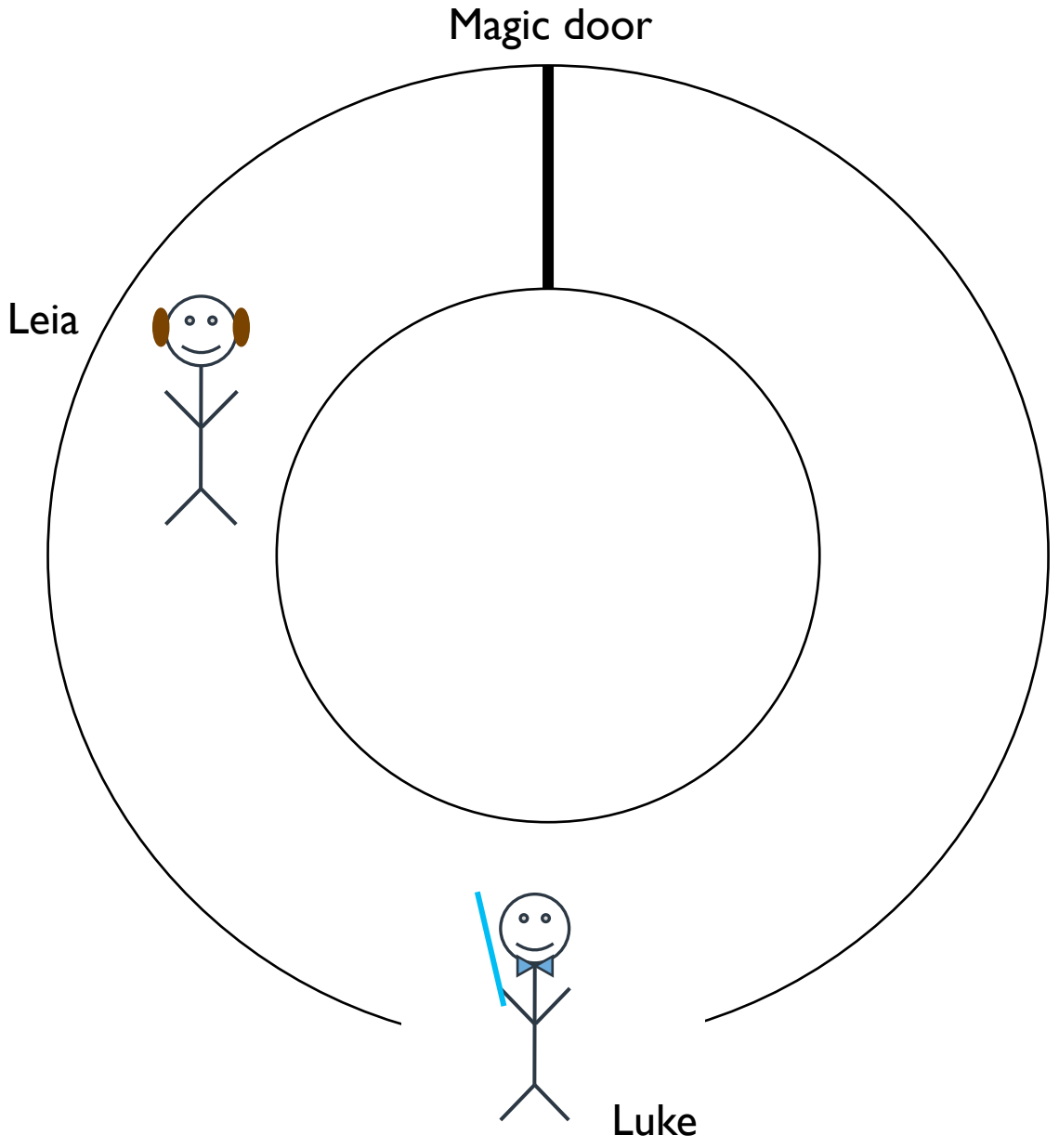
shielded transactions: fully
encrypted transactions
stored on the Zcash blockchain.

zk-SNARKs: Definitions

zk-SNARK: zero-knowledge succinct non-interactive argument of knowledge.

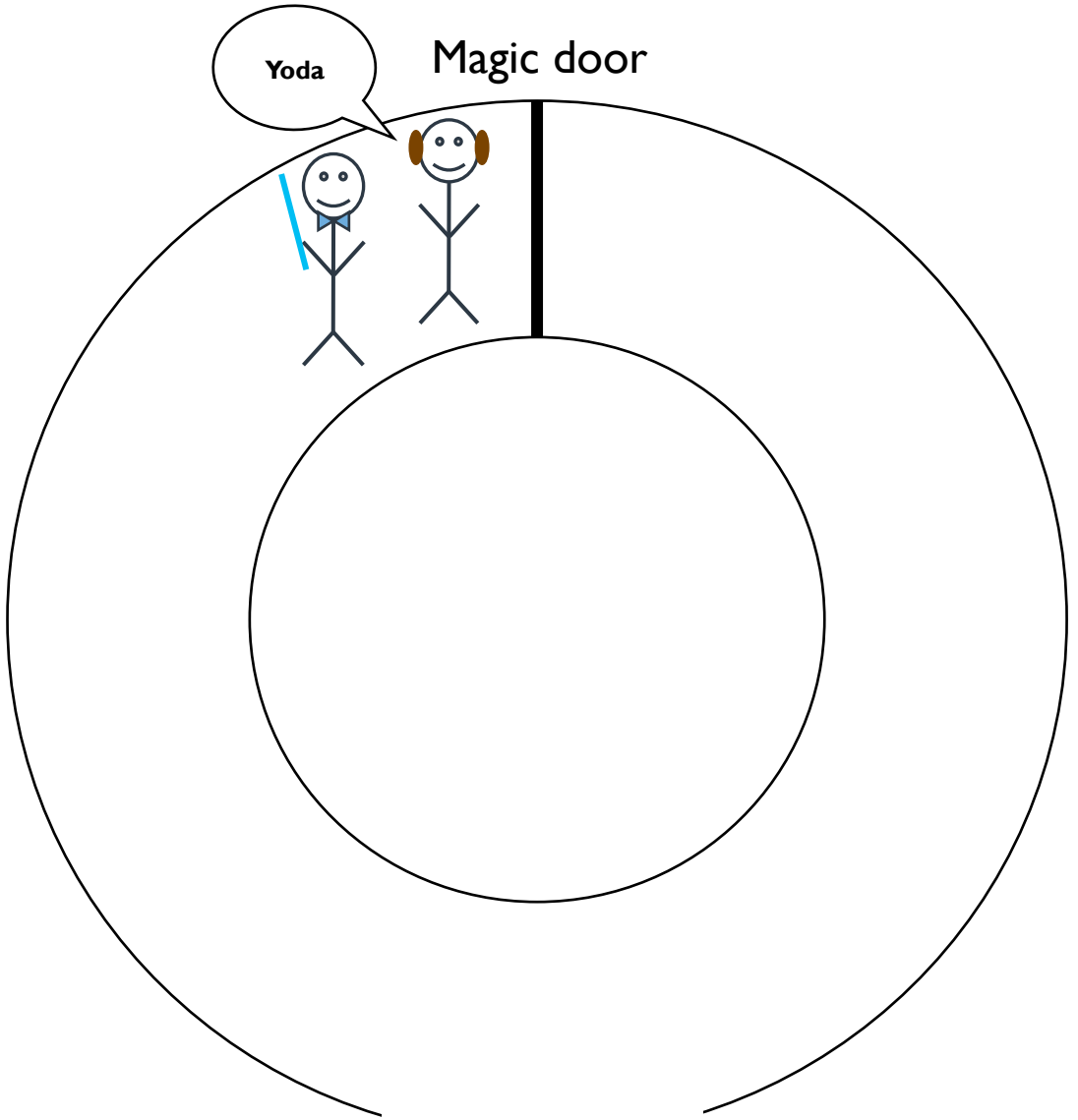
- **Zero knowledge:** transactions don't reveal information about the participants.
- **Succinct:** proofs are (relatively) short and cheap to verify.
- **Non-interactive:** proofs live on blockchain, and anyone can verify them.
- **Argument of knowledge:** anyone can prove transactions are valid.

The Magic Cave



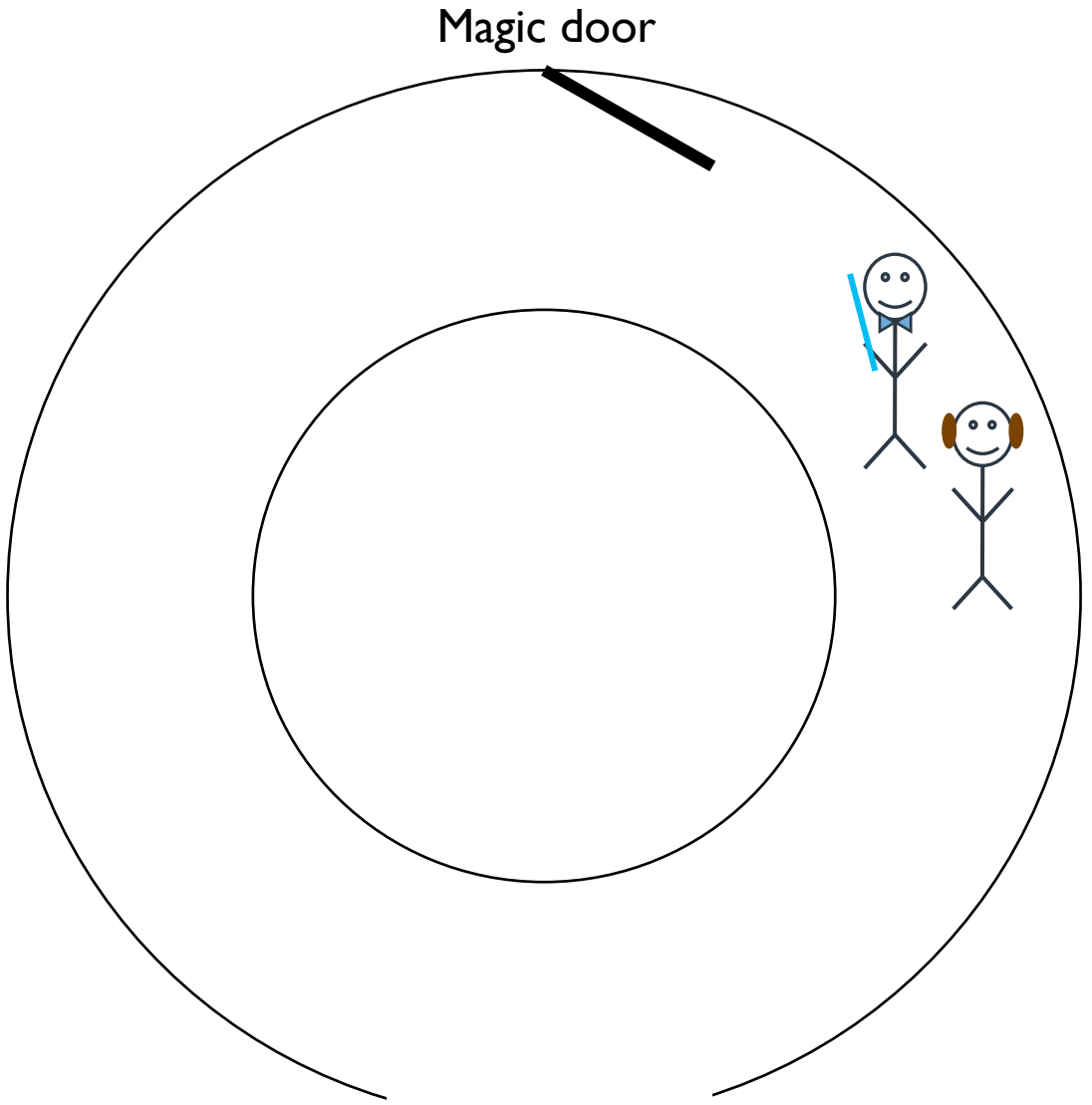
Goal: Leia proves to Luke that she knows the magic word to open the magic door.

The Magic Cave



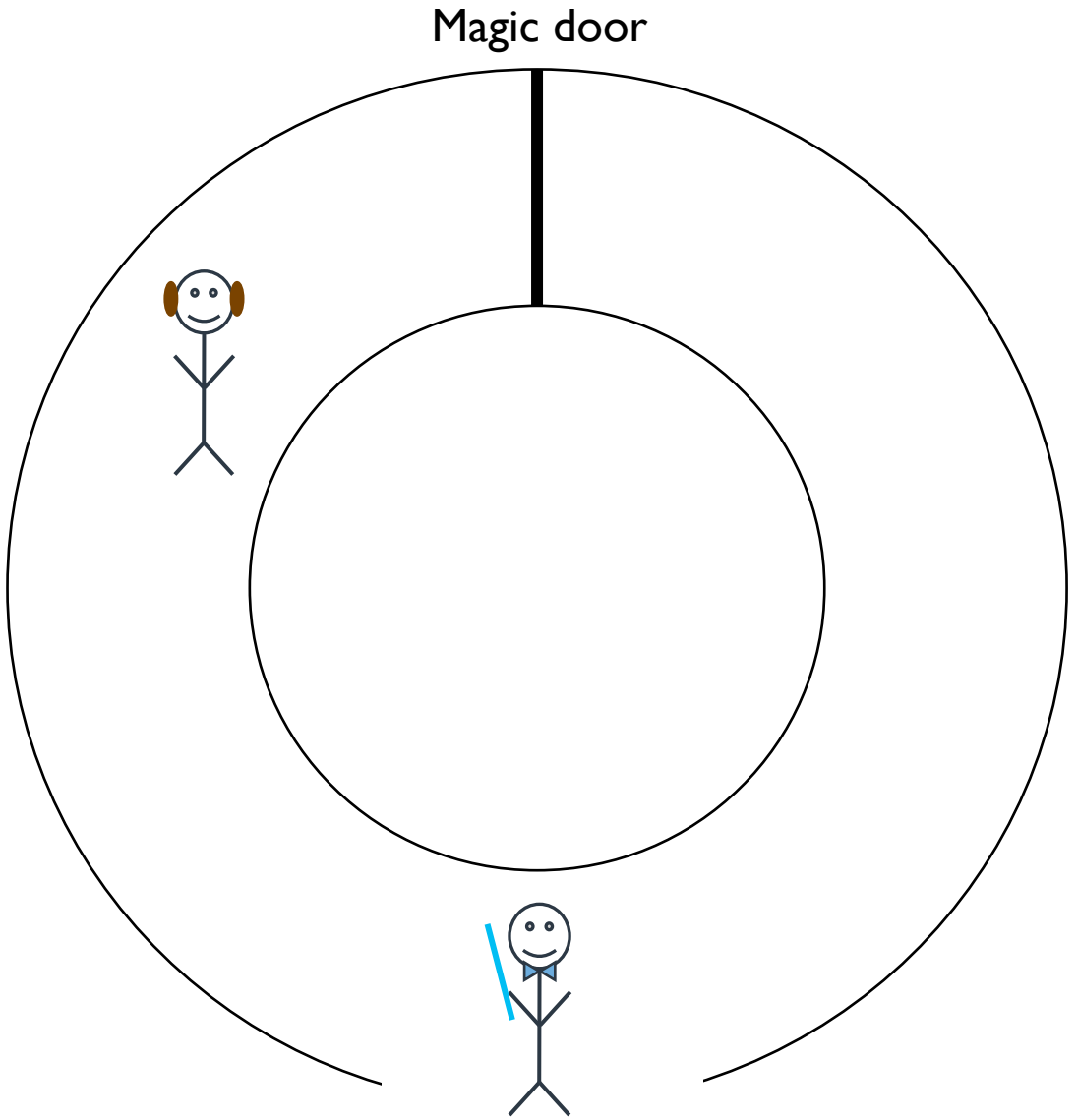
Attempt 1: Leia takes Luke to the door and opens it with the magic word.

The Magic Cave



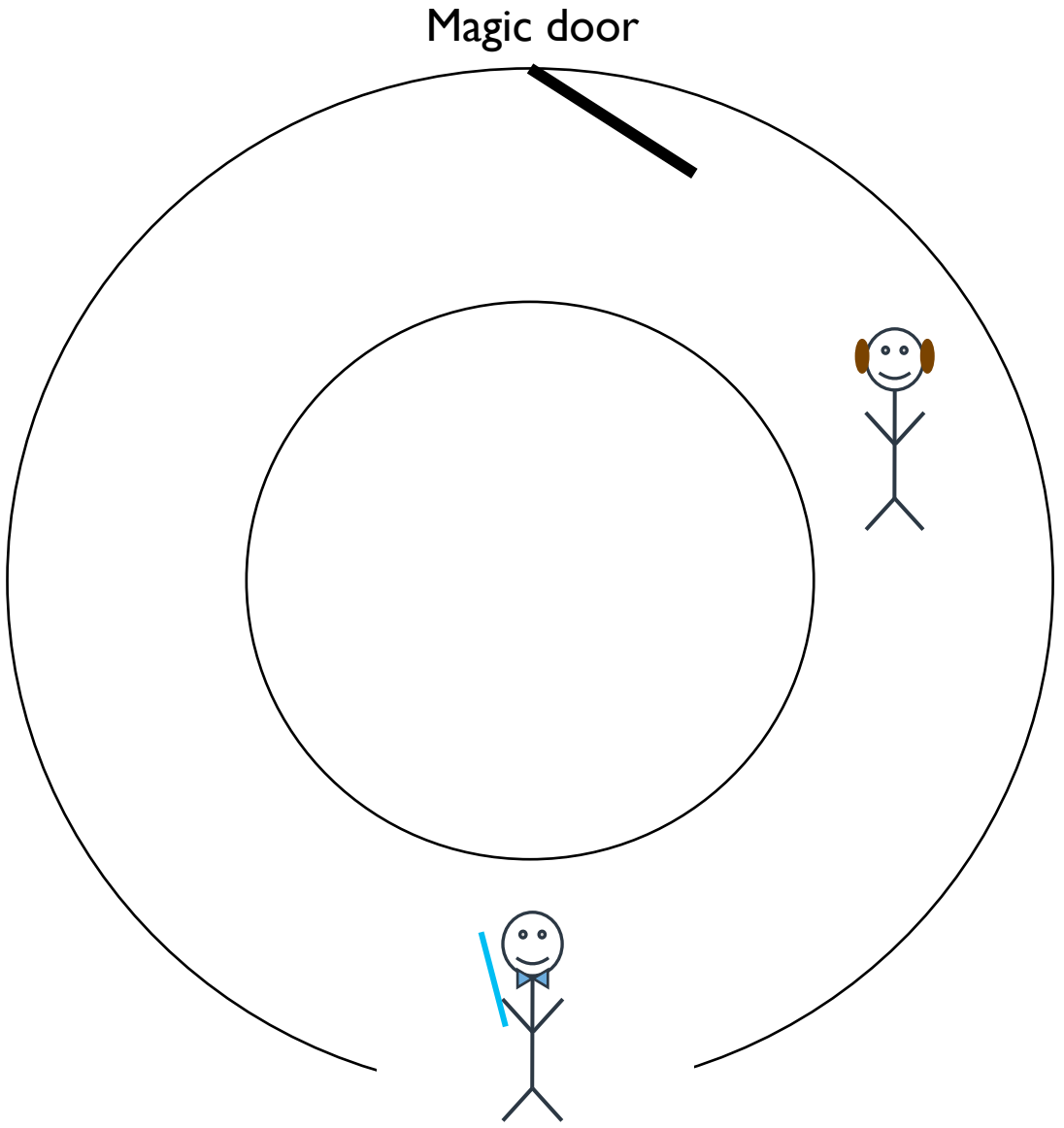
Bad: Luke knows that Leia can get through the door, but now Luke knows the magic word.

The Magic Cave



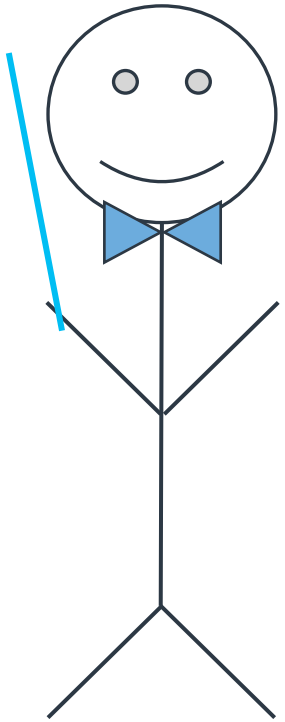
Attempt 2: Leia starts on the left side of the cave, while Luke remains outside.

The Magic Cave

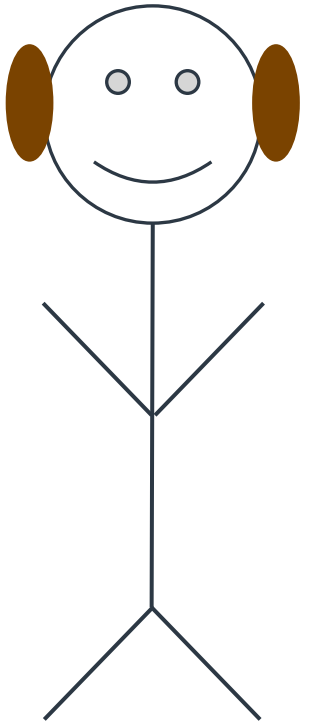


Good: Leia appears on the right side of the cave, implying to Luke that she must know the magic word to get through the door.

The Magic Cave



Success! Luke believes that Leia knows how to get through the door, but Leia has not revealed the magic word.



zk-SNARKs: Overview

In a nutshell . . .

zk-SNARKs transform the process of proving your transaction is valid into showing you know the solution to a set of algebraic equations without revealing the solution or the equations.

zk-SNARKs: Overview

zk-SNARKs show:

1) The sum of transaction inputs matches sum of transaction outputs.

2) Sender holds private spending keys of input notes.

3) Private spending keys can be linked to signature over transaction.

4) For each input note, a revealed commitment* exists.

* Commitment = shows unspent transaction output = HASH(recipient address, amount, ρ, nonce)

The Real Math

A zk-SNARK evaluates a polynomial $P(X)$ created by a specially constructed circuit at a randomly chosen point s in a way that prevents both the Leia, the prover who knows $P(X)$, and Luke, the verifier who knows s, from learning either other's secrets.

The future of digital cash

Why should I care about this technology?

- Privacy-centric coins such as Monero and Zcash will continue to push cryptographic boundaries.

- Additional growth opportunities exist for other distributed ledger-based technologies.

- As long as products ***solve real human needs***, they will last.

Questions?

Additional Slides: zk-SNARKs

How exactly do these things work?

zk-SNARKs: Overview

zk-SNARKs show:

1) The sum of transaction inputs matches sum of transaction outputs.

2) Sender holds private spending keys of input notes.

3) Private spending keys can be linked to signature over transaction.

4) For each input note, a revealed commitment* exists.

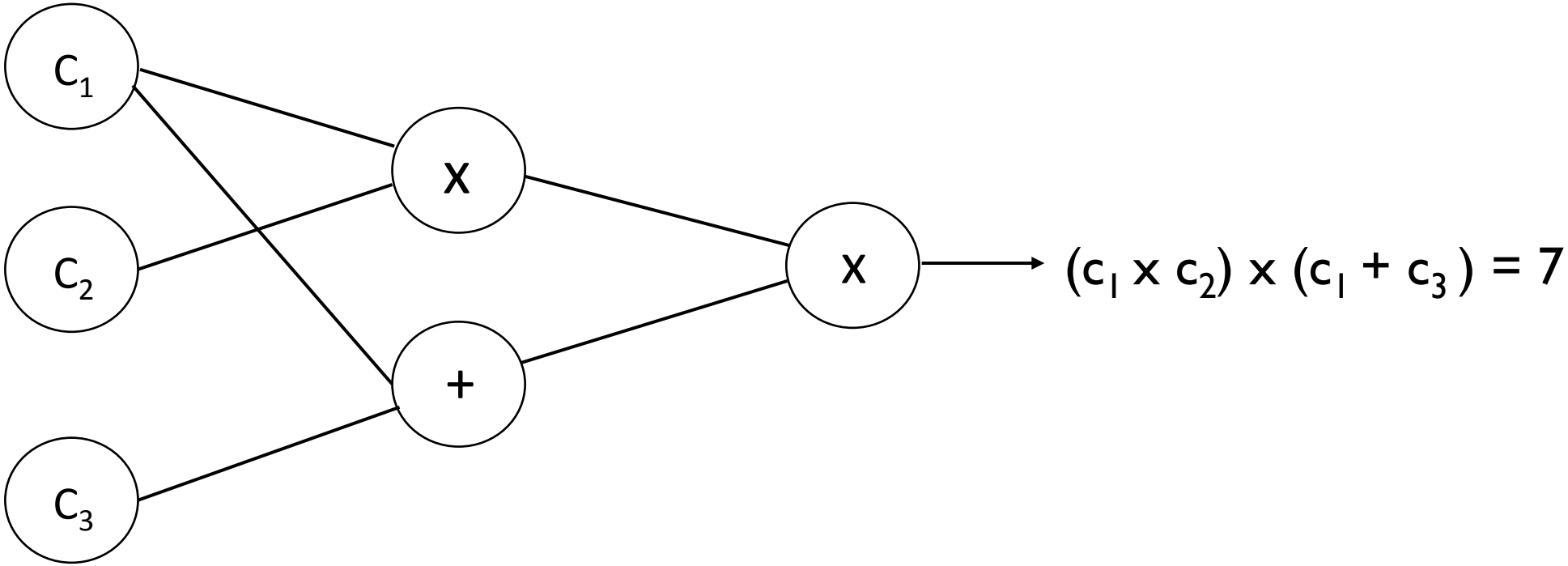
* Commitment = shows unspent transaction output = HASH(recipient address, amount, ρ, nonce)

The Challenge

Leia (the **prover**) wishes to prove to Luke (the **verifier**) that she knows a set of constants (c_1, c_2, c_3) such that $(c_1 x\ c_2) \times (c_1 + c_3) = 7$ without revealing what the constants are.

Simplifying the Problem

Step 1: Construct arithmatic circuit.



Simplifying the Problem?

Step 2: Construct verifying circuit

Rank 1 Constraint Circuit (R1C2): A method to verify that values are travelling correctly through arithmetic circuit. It is possible to only check values at multiplicative gates using an R1C2 circuit.

Quadratic Arithmatic Program (QAP): This method, a refinement of R1C2, verifies circuit correctness by only checking if polynomials match expected values at a randomly chosen point in the circuit. If the prover is telling the truth (i.e., knows constants (c_1, c_2, c_3) s.t. $(c_1 x\ c_2) \times (c_1 + c_3) = 7$) the polynomials will match regardless of the point chosen.

The Real Math

Step 3: Use homomorphic hiding and elliptic curve pairings to do polynomial checking.

A zk-SNARK evaluates a polynomial $P(X)$ created by a specially constructed circuit at a randomly chosen point s in a way that prevents both the Leia, the prover who knows $P(X)$, and Luke, the verifier who knows s, from learning either other's secrets.

What's a Homomorphic Hiding?

Definition

A homomorphic hiding $E(x)$ of a number x satisfies:

- For most x, given $E(x)$, it is difficult to recover x.
- If $x \neq y$, then $E(x) \neq E(y)$.
- $E(x)+E(y)=E(x+y)$

What's a Homomorphic Hiding?

Definition

A homomorphic hiding $E(x)$ of a number x is a function satisfying the following properties:

- For most x, given $E(x)$, it is difficult to recover x.
- If $x \neq y$, then $E(x) \neq E(y)$.
- $E(x)+E(y)=E(x+y)$

Example

Consider the group $\mathbb{Z}\downarrow p$, where p is prime. Define $E(x)=g\uparrow x$, where g is a generator of $\mathbb{Z}\downarrow p$.

- Taking discrete logs is hard in $\mathbb{Z}\downarrow p$.
- Because g is a generator of $\mathbb{Z}\downarrow p$, each $E(x)$ element is unique.
- $E(x+y)=g\uparrow x+g\uparrow y=E(x)+E(y)$

Restating the Goal

Goal

Given that Leia knows a polynomial P of degree d over field $\mathbb{F}_p$, Luke would like to know a solution $P(s)$ for a point $s \in \mathbb{F}_p$ without Leia revealing P or Luke revealing s.

Baby Steps

Knowledge of Coefficient Test

Goal

Leia must prove to Luke that she knows a number γ without revealing the number's value.

Definition

For $\alpha \in \mathbb{F}_p$, call a pair (a, b) an α-pair if $a, b \neq 0$ and $b = \alpha a$.

Baby Steps

Verifiable, Blind Polynomial Evaluation

Goal

Verifiable, blind evaluation of $P(x)$ known by Leia at a point $s \in \mathbb{F}_p$ provided by Luke.

Definition

If Luke sends multiple $\propto$ pairs to Leia, Leia can provide (a', b') via a linear combination of Luke's α pairs. She chooses coefficients c_i s.t.

$$a' = \sum_{i=1}^{d} c_i \, a_i \; .$$

Verifiable Blind Polynomial Evaluation Protocol

1. Luke chooses $\alpha \in \mathbb{F}_p^*$.

2. Luke sends Leia the hiding $(g, gs, ..., gs^d)$ and $(\alpha g, \alpha gs, ..., \alpha gs^d)$, where s is the evaluation point.

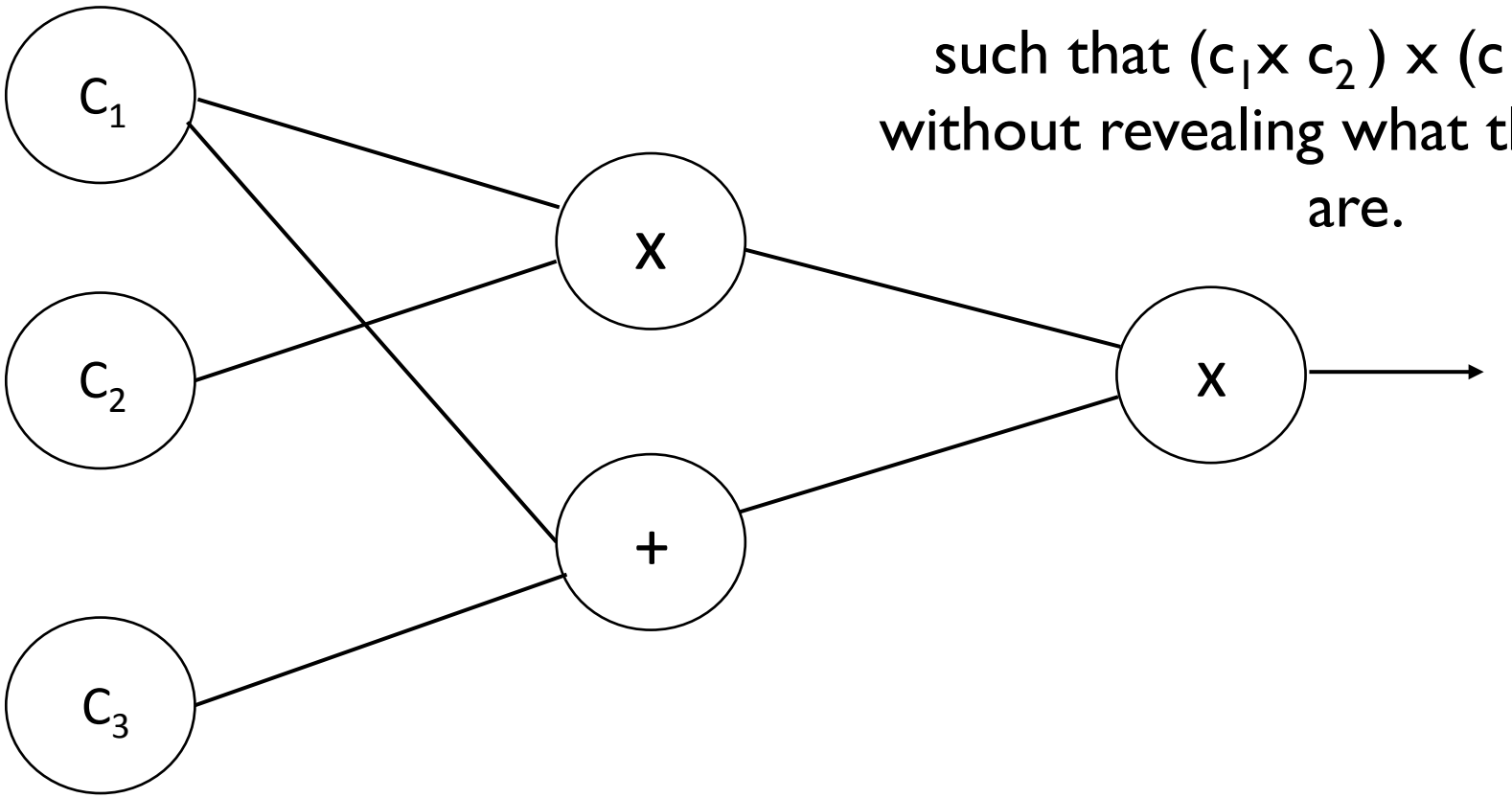
3. Leia computes $a' = g^{P(s)}$ and $b' = g^{\alpha P(s)}$, where $P(s)$ is a linear combination of $(g, gs, ..., gs^d)$.

4. Luke checks that $b' = \alpha a'$.

5. If this is true, then by the d-Knowledge of Coefficient Test, Leia must know a set of coefficients c_i s.t. $\prod_{i=1}^{d} c_i s^i g = a'$. Therefore, $a' = P(s)g$, where $P(X) = \sum^{\square} c_i X^i$ known to Leia.

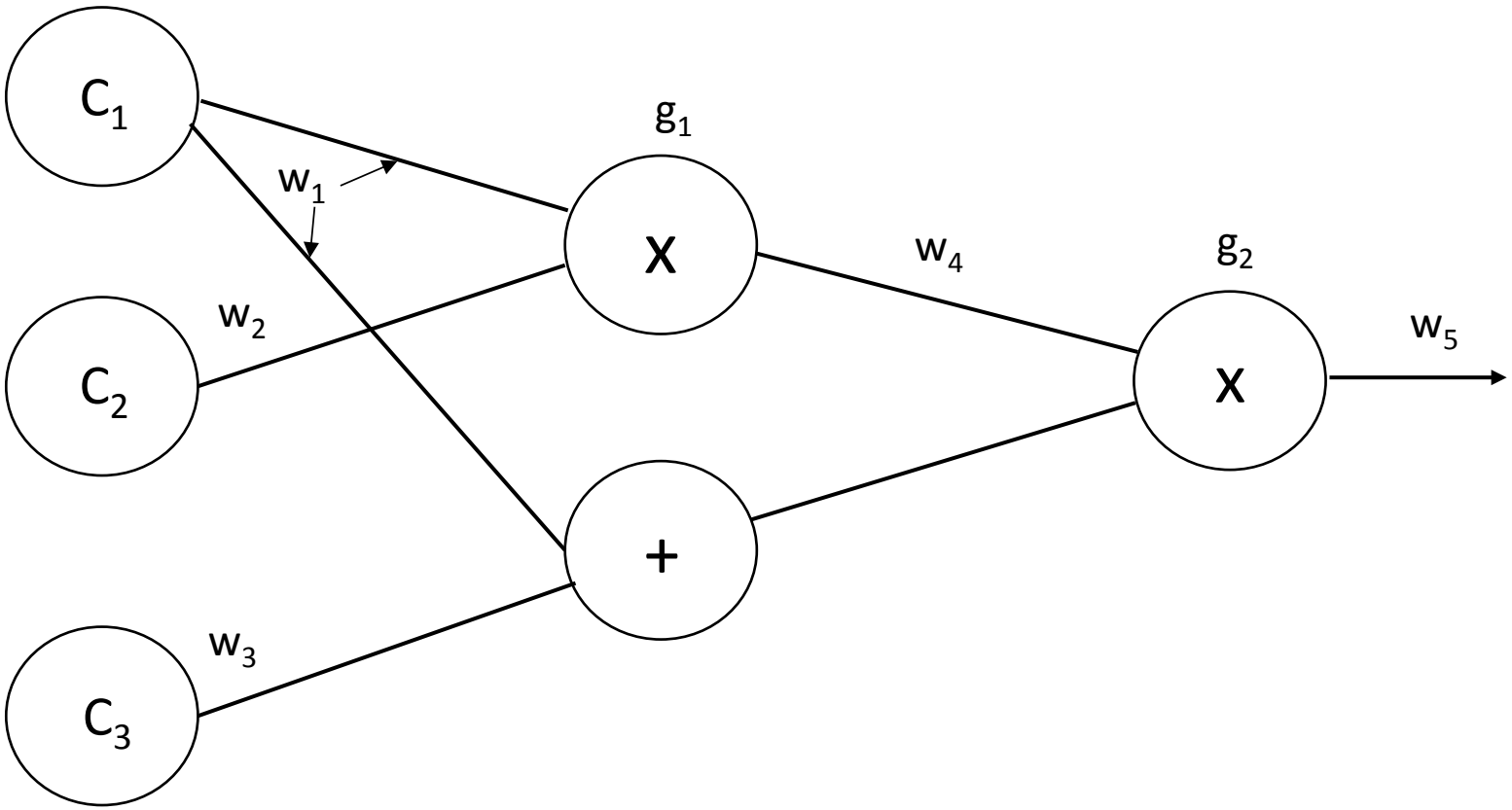
Quadratic Arithmetic Polynomial

Recall: Leia (the **prover**) wishes to prove to Luke (the **verifier**) that she knows a set of constants (c_1, c_2, c_3) such that $(c_1 x\ c_2) \times (c_1 + c_3) = 7$ without revealing what the constants are.



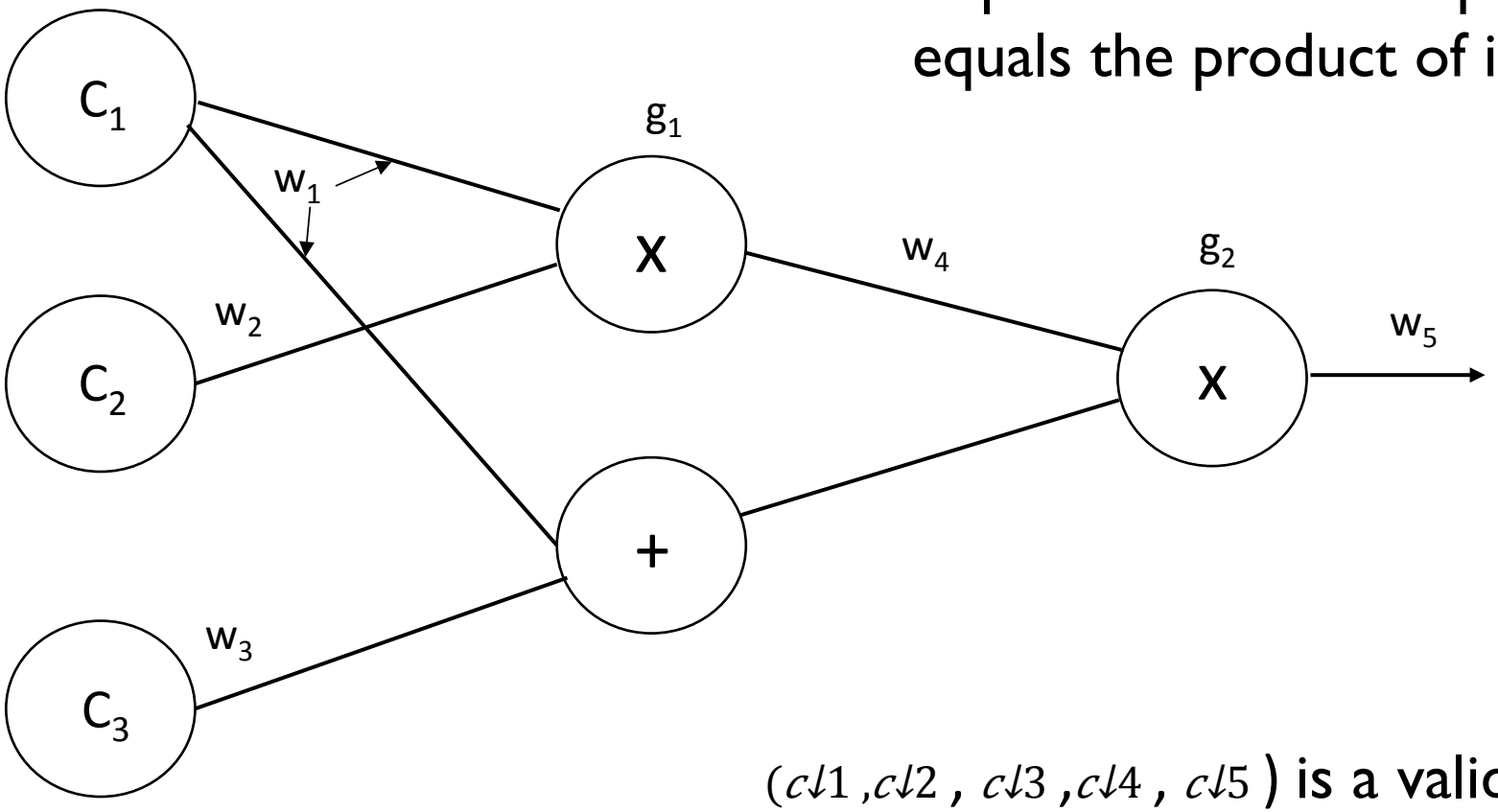
Quadratic Arithmetic Polynomial

Step 1: Assign values to wires and gates in the circuit.



Quadratic Arithmatic Polynomial

Legal assignment: Assignment of values to labelled wires such that the output values of multiplication gates equals the product of input values.

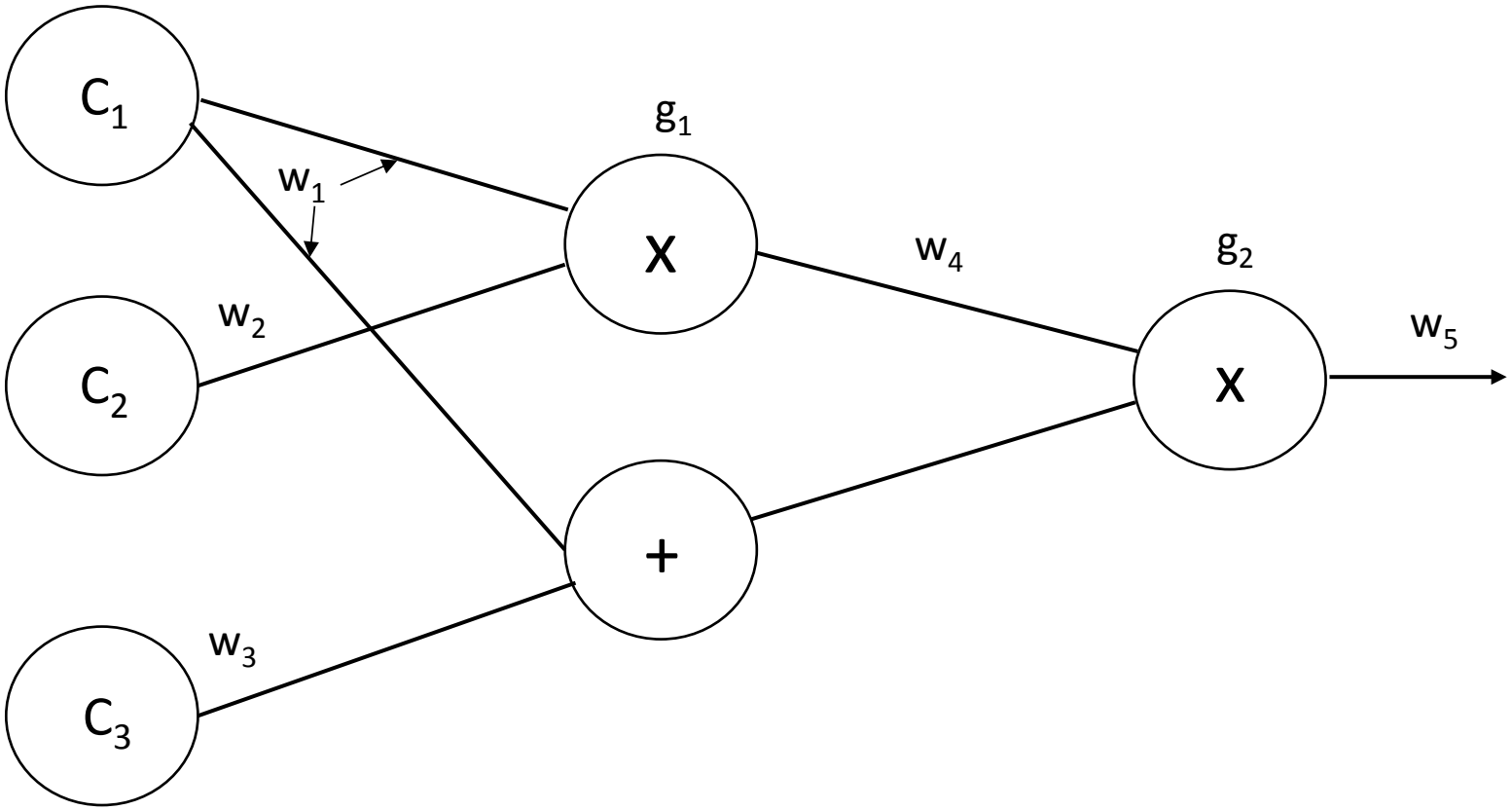


$(c\downarrow 1, c\downarrow 2, c\downarrow 3, c\downarrow 4, c\downarrow 5)$ is a valid assignment

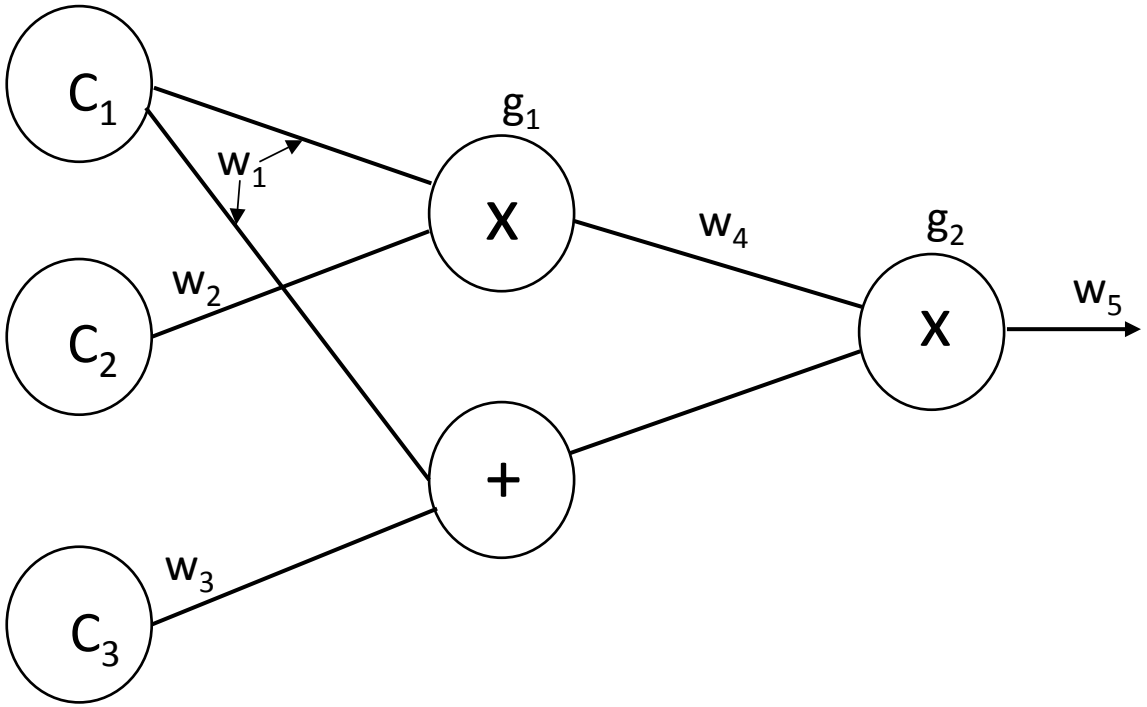
$iff\ c\downarrow 4 = c\downarrow 1 \times c\downarrow 2$ and $c\downarrow 5 = c\downarrow 4\,(c\downarrow 1 + c\downarrow 3)$

Quadratic Arithmatic Polynomial

Leia's task is to prove she knows $(c\downarrow 1$, $c\downarrow 2$, $c\downarrow 3$, $c\downarrow 4$, $c\downarrow 5$) s.t. $c\downarrow 5 = 7$.



Quadratic Arithmatic Polynomial

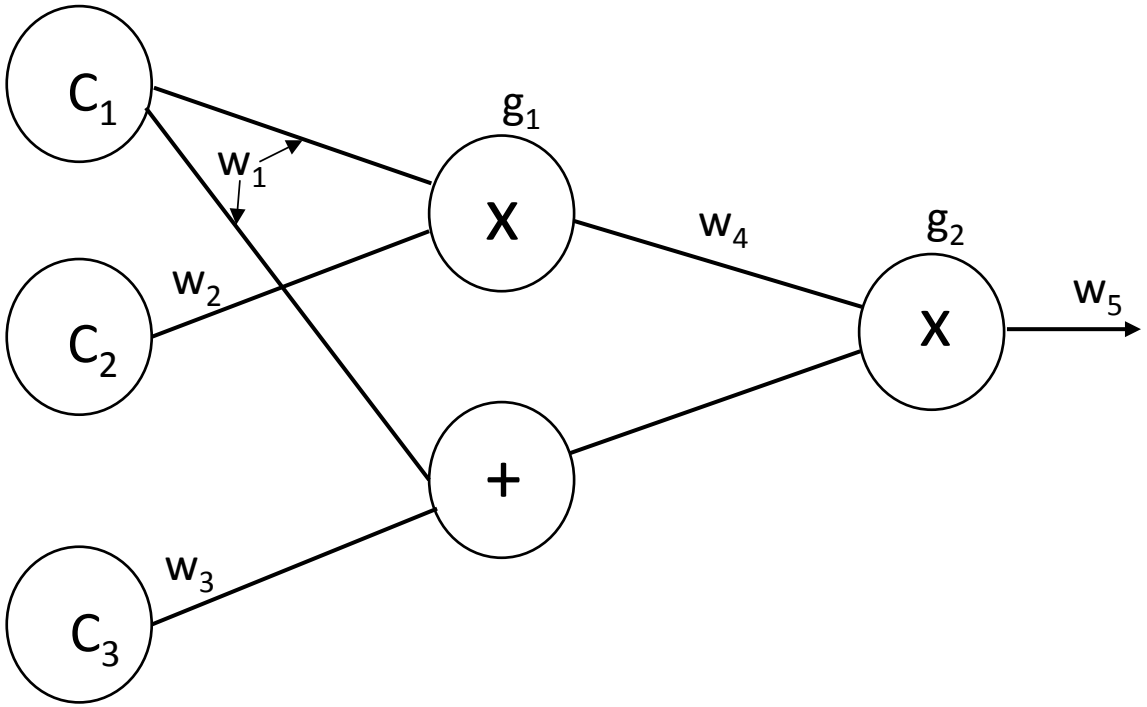


Step 1: Construct wire polynomials

- Assign group elements to gates:
  - $g\downarrow 1 = 1 \in \mathbb{F}\downarrow p$
  - $g\downarrow 2 = 2 \in \mathbb{F}\downarrow p$

(call these the target points)

Quadratic Arithmatic Polynomial



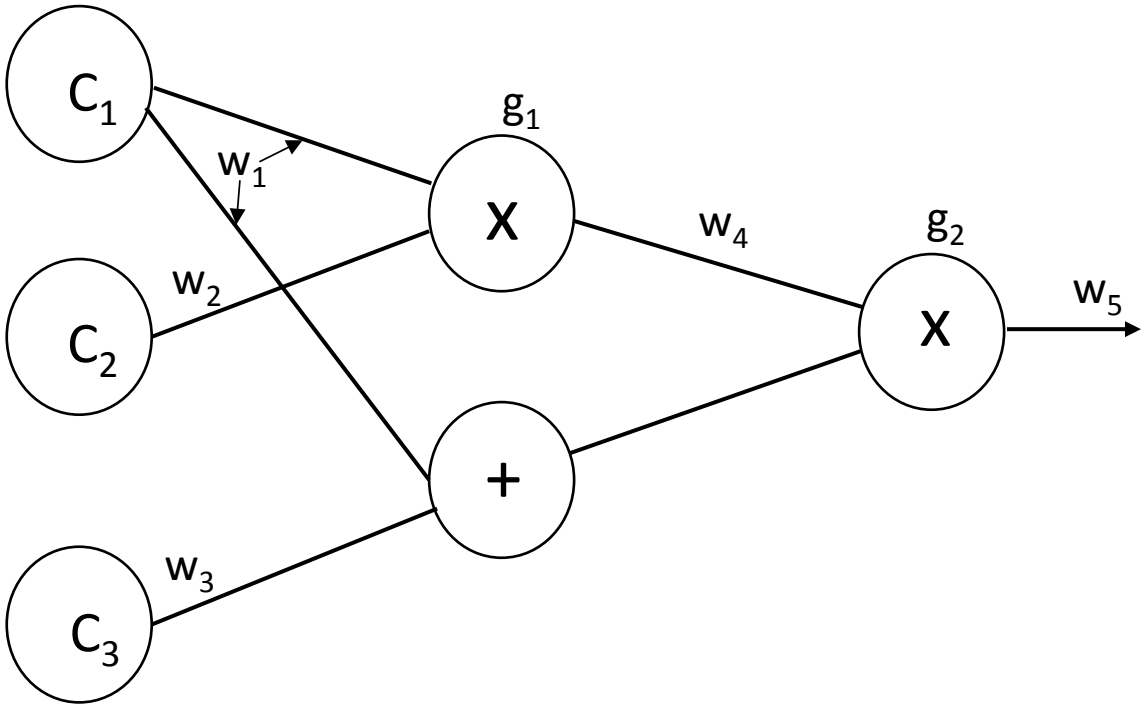
Step 1: Construct wire polynomials

- Assign group elements to gates:
  - $g\downarrow 1 = 1 \in \mathbb{F}\downarrow p$
  - $g\downarrow 2 = 2 \in \mathbb{F}\downarrow p$

  (call these the target points)

- Now create L, R, and O wire polynomials. These should be 0 everywhere except when the wires are part of a target point multiplicative gate.

Quadratic Arithmatic Polynomial



Step 1: Construct wire polynomials

- Assign group elements to gates:
  - $g\downarrow 1 = 1 \in \mathbb{F}\downarrow p$
  - $g\downarrow 2 = 2 \in \mathbb{F}\downarrow p$

  (call these the target points)

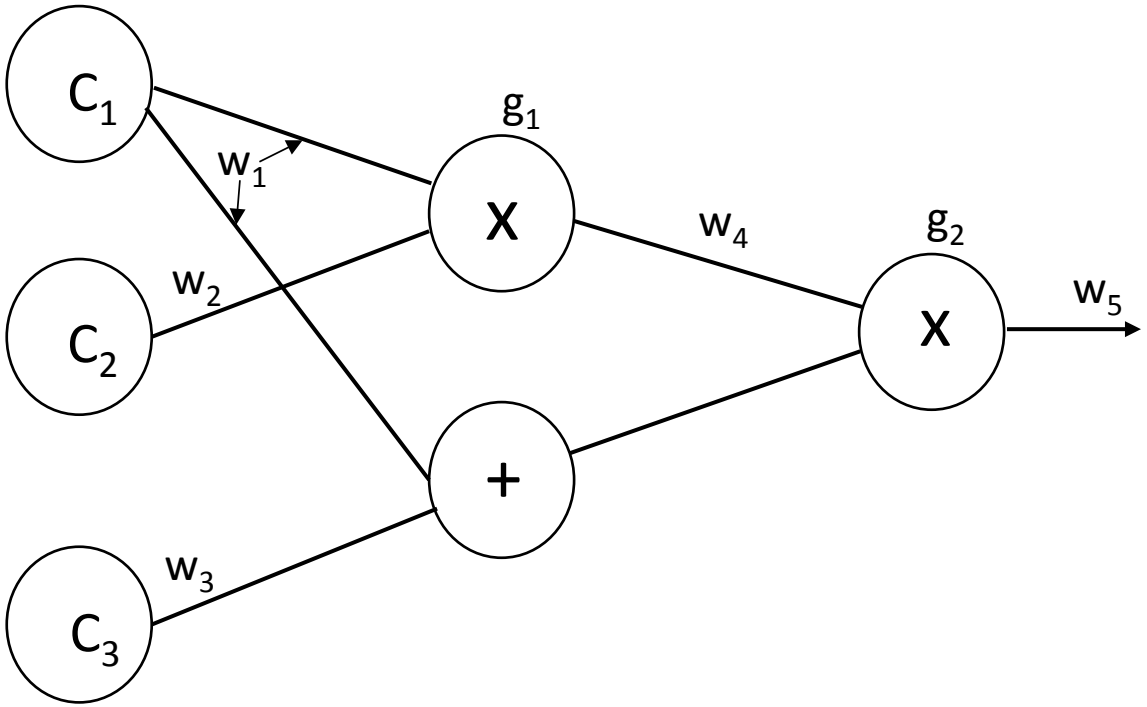
- Now create L, R, and O wire polynomials. These should be 0 everywhere except when the wires are part of a target point multiplicative gate. We pretend addition gates don't exist.

$w\downarrow 1$, $w\downarrow 2$, $w\downarrow 4$ wires generate the polynomials:

$$L\downarrow 1 = R\downarrow 2 = O\downarrow 4 = 2 - X$$

$$L\downarrow 4 = R\downarrow 1 = R\downarrow 3 = O\downarrow 5 = X - 1$$

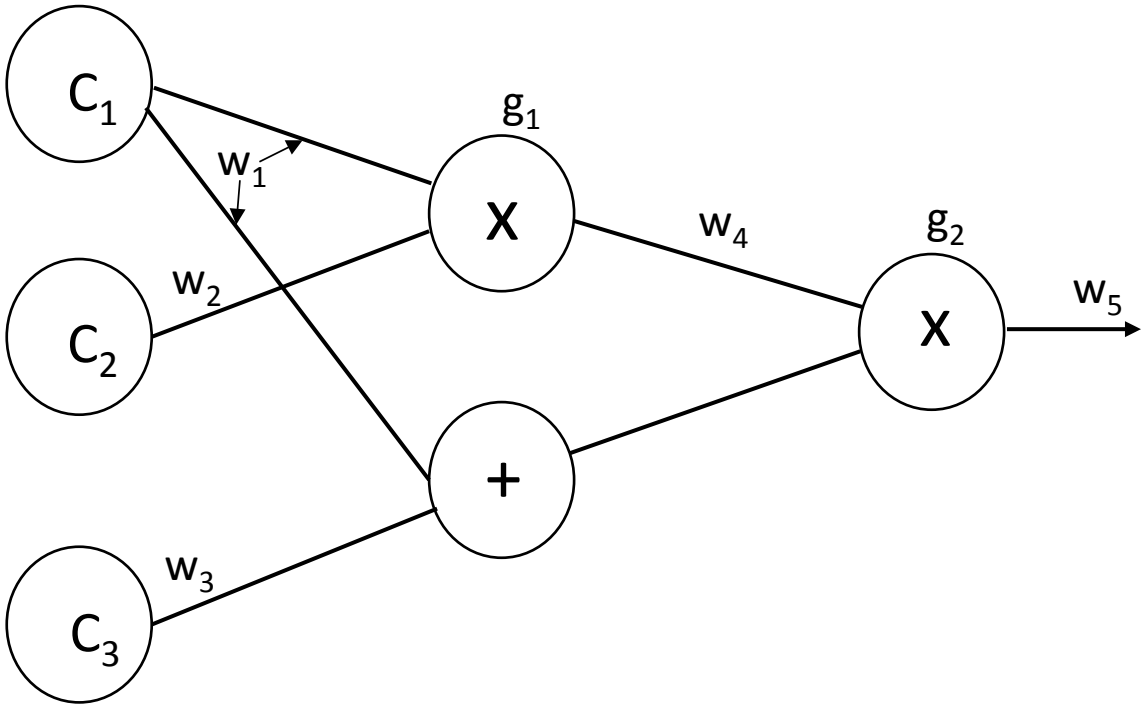
Quadratic Arithmatic Polynomial



Step 2: Create overall polynomial

- Use c_i as coefficients to combine wire polynomials:
  - L = $\sum_i c_i L_i$
  - R = $\sum_i c_i R_i$
  - O = $\sum_i c_i O_i$

Quadratic Arithmatic Polynomial



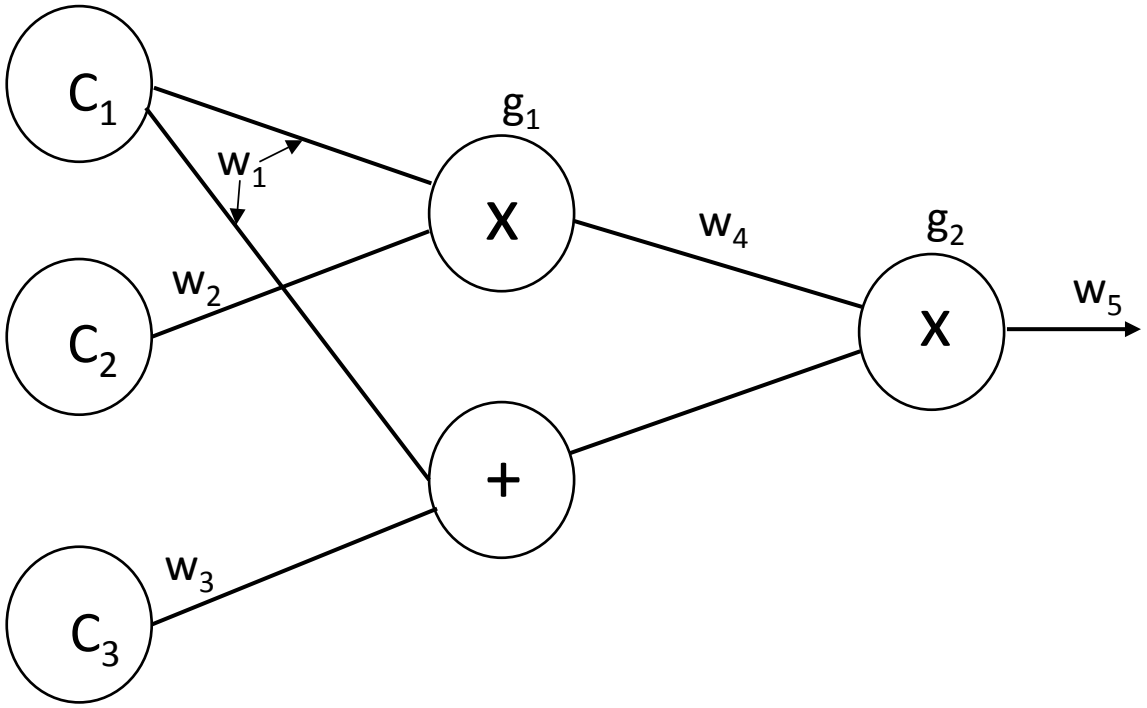
Step 2: Create overall polynomial

- Use c_i as coefficients to combine wire polynomials:
  - L = $\sum_i c_i L_i$
  - R = $\sum_i c_i R_i$
  - O = $\sum_i c_i O_i$
- Let $P = LR - O$

Definition

We say (c_1 , c_2 , c_3 , c_4 , c_5) is a valid assignment *iff P* vanishes on all target points.

Quadratic Arithmatic Polynomial



Step 2: Create overall polynomial

- Use c_i as coefficients to combine wire polynomials:
  - L = $\sum_i c_i L_i$
  - R = $\sum_i c_i R_i$
  - O = $\sum_i c_i O_i$
- Let $P = LR - O$

Definition

We say (c_1 , c_2 , c_3 , c_4 , c_5) is a valid assignment *iff P* vanishes on all target points.

Example

$P(1) = L_1 R_2 - O_4 = (2-1)(2-1) - (2-1) = 0$

Algebra flashback theory

For a polynomial P and $a \in \mathbb{F}\downarrow p$, $\mathrm{P}(\mathrm{a})=0$ iff $X-a$ divides P (i.e. $P=(X-a)H$)

Quadratic Arithmetic Polynomial

Definition

A Quadratic Arithmetic Polynomial Q of degree d and size m consists of polynomials $L↓1$, ..., $L↓m$,$R↓1$, ...,$R↓m$,$O↓1$, ...,$O↓m$ and target polynomial T also of degree d.

An assignment $(c↓1$,$c↓2$, $c↓3$,$c↓4$, $c↓5$) satisfies Q if, letting L $=∑i=1↑d▒c↓i L↓i$, similarly for R and O, and $P=LR−O$, we have that $T \mid P$.

Pinocchio Protocol

Definition

If Leia has an assignment $(c_1, c_2, c_3, c_4, c_5)$ that satisfies Q, then, defining L, R, O, P as before, there exists a polynomial H such that $P=HT$ and for all $s \in \mathbb{F}_p^*$, $P(s)=H(s)T(s)$.

Pinocchio Protocol

1) Leia chooses L, R, O, H of degree at most d.

2) Luke chooses $s \in \mathbb{F}\downarrow p \uparrow *$ and computes $E(T(s))$.

3) Leia computes the hidings of all polynomials at s (i.e. $E(L(s)), E(R(s)), E(O(s)), E(H(s))$).

4) Luke evaluates $E(L(s)R(s) - O(s)) = E(T(s)H(s))$.

Four Final Problems

Problem 1

Leia must form her polynomials correctly, otherwise she can provide a satisfactory solution that doesn't fulfill the necessary QAP constraints.

Solution

Require Leia to prove that $F=\sum i=1\uparrow d \square c\downarrow i\, L\downarrow i$, where $F=L+X\uparrow d+1\, R+X\uparrow 2(d+1)\, O$ because the coefficients of L, R, O do not mix.

Four Final Problems

Problem 2

Leia's polynomial assignment is not fully concealed. Leia revealing $E(L(s)), E(R(s)), E(O(s)), E(H(s))$ allows Luke to choose a different assignment $(c'\downarrow 1, c'\downarrow 2, \ldots, 'c\downarrow d)$ that yields the hidings $(E(L'(s)), E(R'(s)), E(O'(s)), E(H'(s)))$. If these do not match Leia's hidings, Luke knows Leia's coefficients are not $(c'\downarrow 1, c'\downarrow 2, \ldots, 'c\downarrow d)$.

Solution

Leia adds a random T-shift to each L, R, O polynomial.

(i.e. $L\downarrow i = L + \delta\downarrow i\, T$).

Four Final Problems

ELLIPTIC CURVES AT LONG LAST

Problem 3

The Pinocchio protocol requires a homomorphic hiding that supports multiplication. Luke must evaluate $E(L(s)R(s)-O(s))=E(T(s)H(s))$

Solution

Use elliptic curves!

Four Final Problems

Problem 4

Zk-SNARKS should be non-interactive.

Solution

Use a common reference string (CRS) for verification.

Common Reference String

1. Choose $\alpha \in \mathbb{F}_{\downarrow r \uparrow *}$, $s \in \mathbb{F}_{\downarrow r}$ at the beginning and publish CRS = $(E_{\downarrow 1}(1), E_{\downarrow 1}(s), \ldots, E_{\downarrow 1}(s^{\uparrow d}), E_{\downarrow 2}(\alpha), E_{\downarrow 2}(\alpha s), \ldots, E_{\downarrow 2}(\alpha s^{\uparrow d}))$.

2. To prove, Leia computes $a = E_{\downarrow 1}(P(s))$ and $b = E_{\downarrow 2}(P(s))$.

3. To verify, fix $x, y \in \mathbb{F}_{\downarrow r}$ s.t. $a = E_{\downarrow 1}(x)$ and $b = E_{\downarrow 2}(y)$. Luke computes $E(\alpha x) = \text{Tate}(E_{\downarrow 1}(x), E_{\downarrow 2}(\alpha))$ and $E(y) = \text{Tate}(E_{\downarrow 1}(1), E_{\downarrow 2}(y))$.

4. If these are equal, implies that $y = \alpha x$ and that Leia has provided a valid solution.

Wrapup

Zk-SNARKs enable the verification of encrypted transactions on the Zcash blockchain by transforming the problem of verifying a transaction into the problem of proving you know the value of a polynomial at a random point.

Wrapup

Questions?