Lecture 11:
Privacy in blockchains

Yury Yanovich

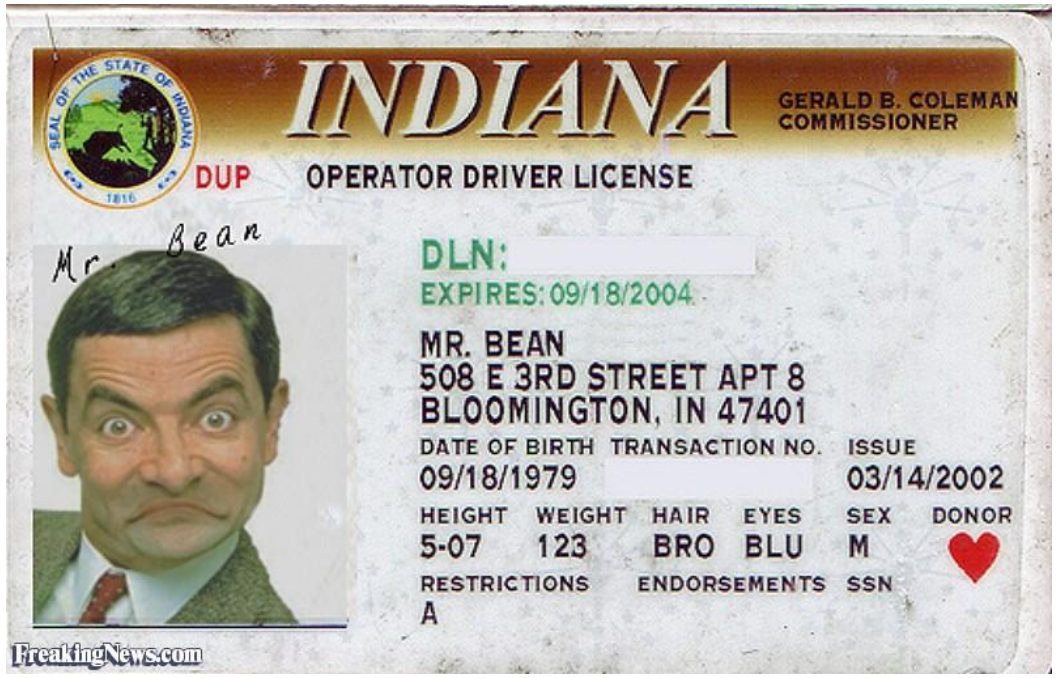
December 4, 2018

1. Privacy issues in blockchains

Objects

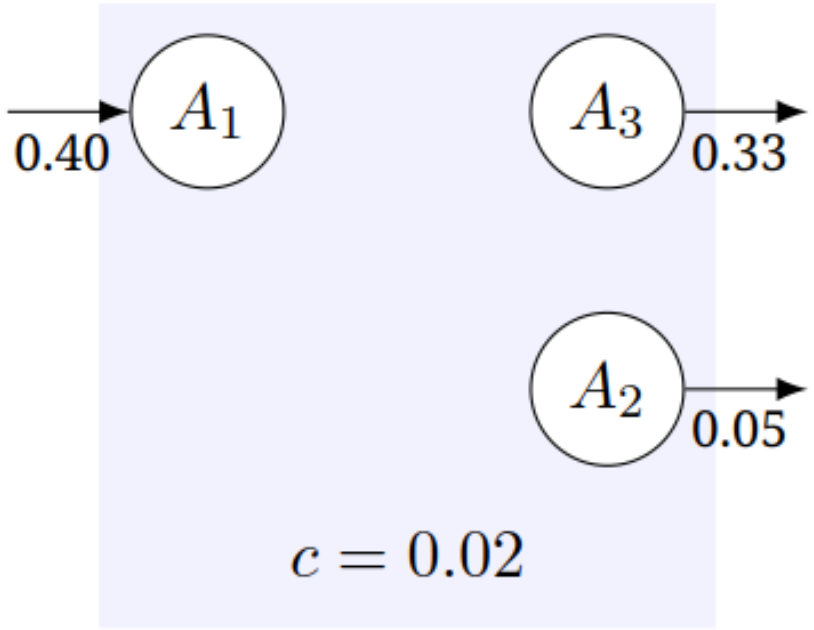
- **People and organizations**
  - Storage and processing of personal data
    - General Data Protection Regulation (GDPR)
  - Money: invariants instead of amounts, senders and recipients
- **(Smart) contracts**
  - check and process conditions without disclosing them

Example 1: road police and driving license

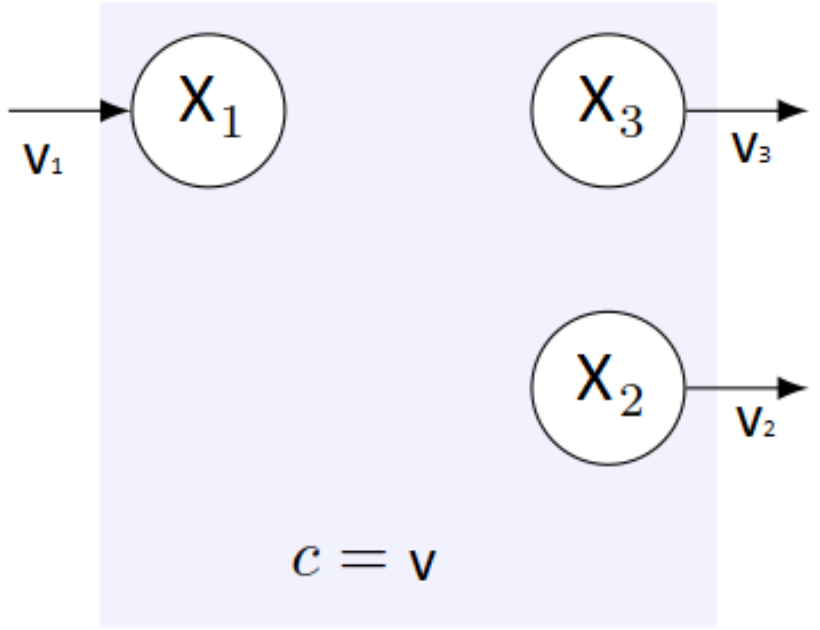


vs

Example 2: money transactions



vs



where

- X_1 is unspent
- $v_1+v=v_2+v_3$
- X_1, X_2, X_3 are unknown

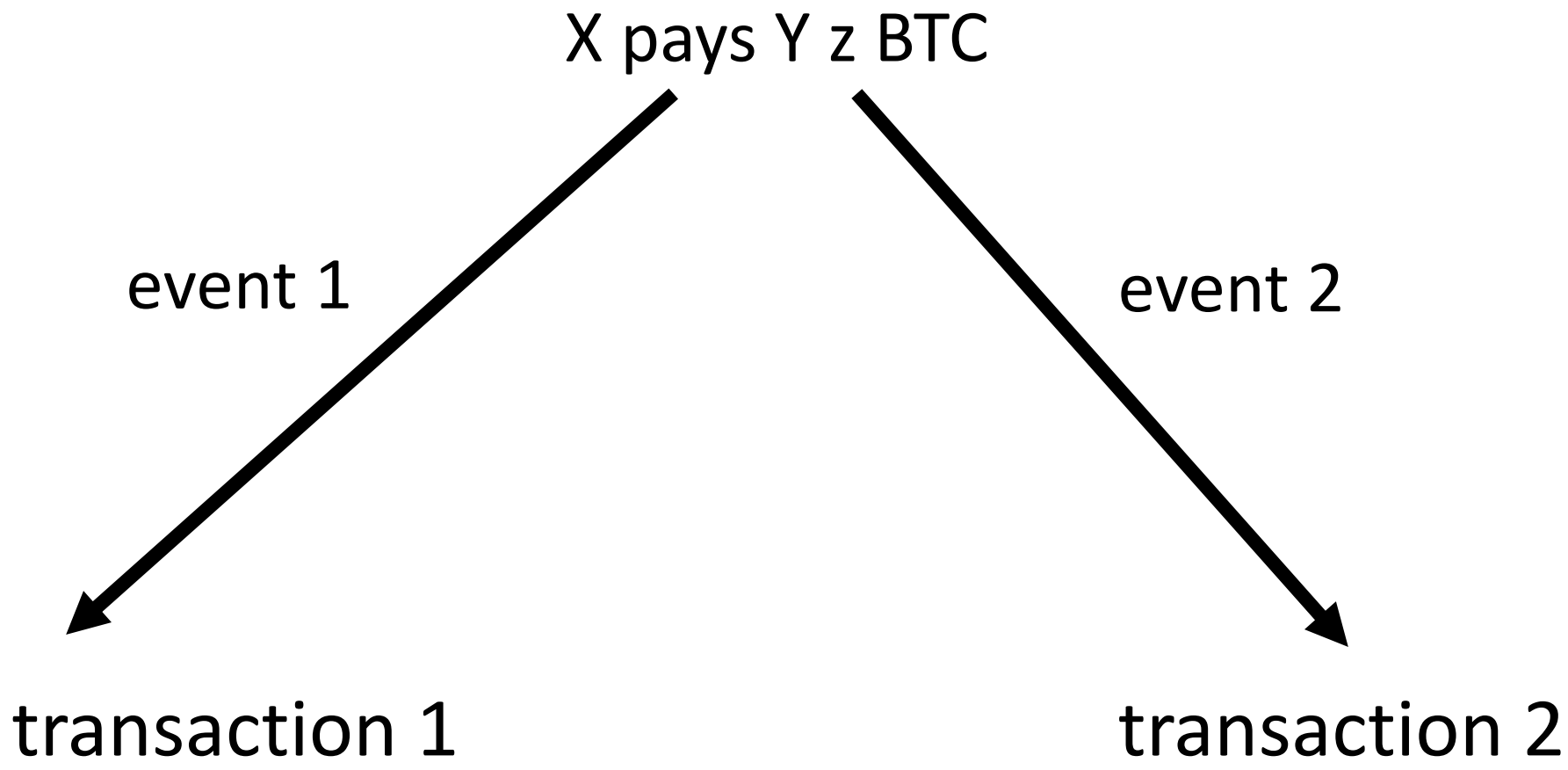
Example 3: insurance

Bob pays Alice 0,01 BTC

Alice pays Bob 1 BTC

nothing

Example 3: insurance (2)



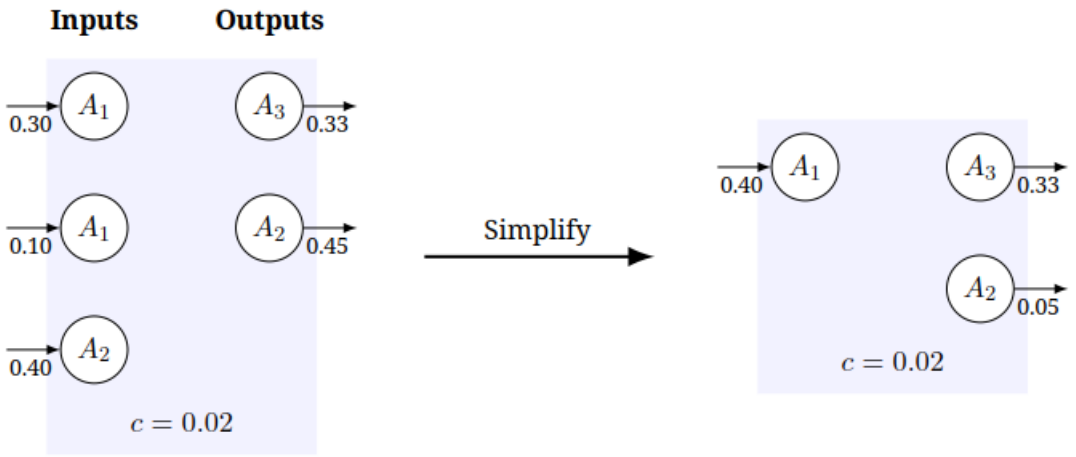
Privacy arms race

“Sword”

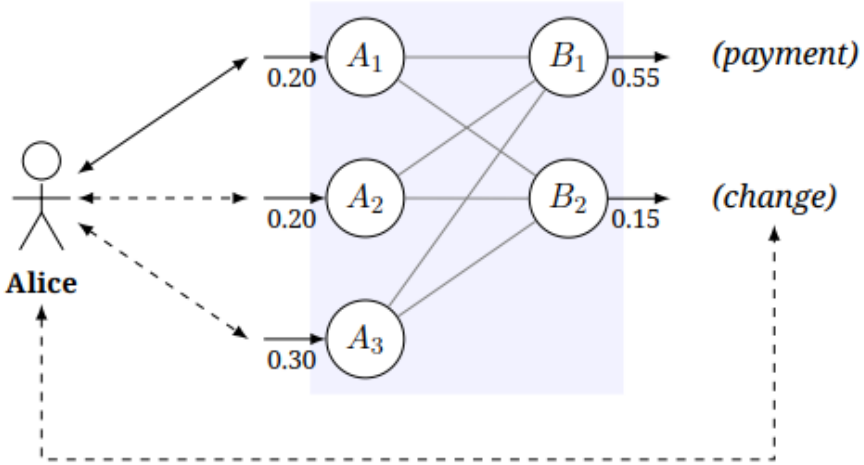
“Shield”

2. Sword: Bitcoin case study

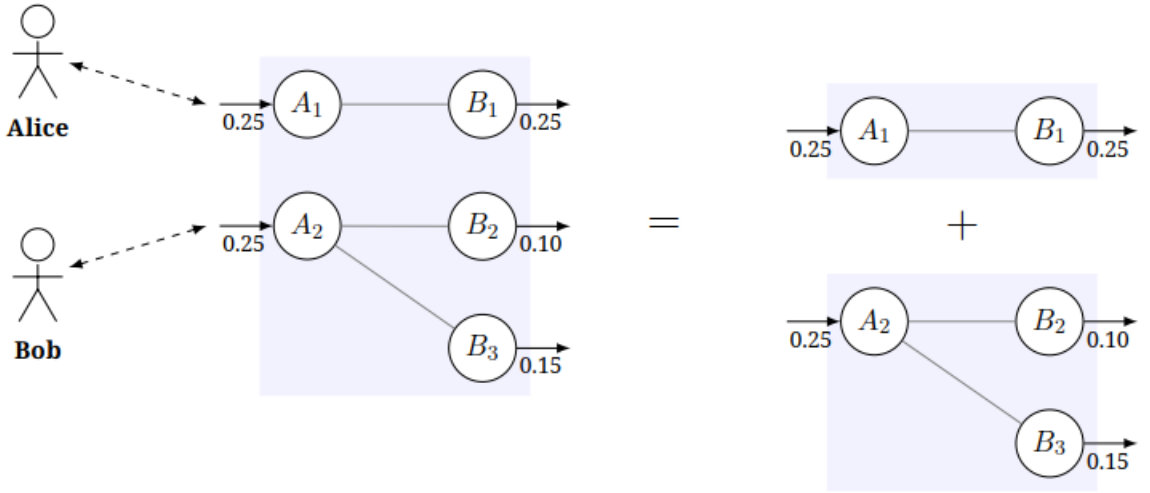
Transactions



Structure of a typical non-mixing transaction

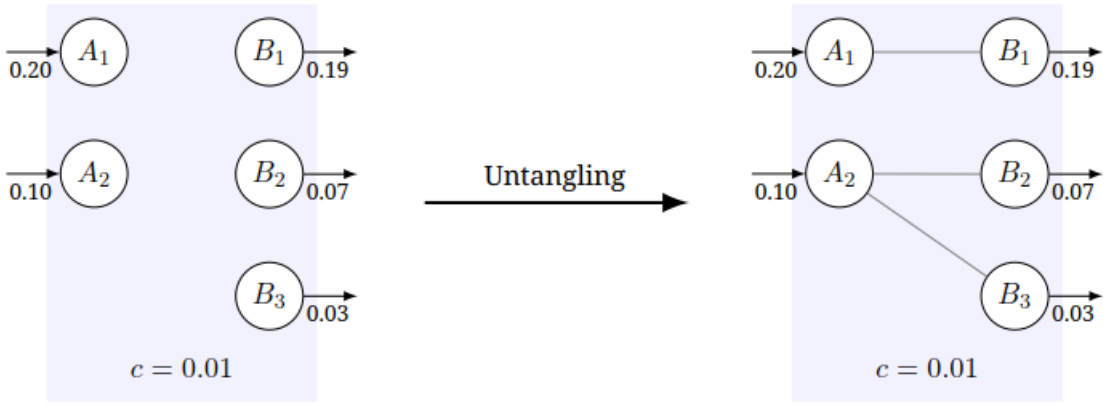


Shared send mixing transaction with two participants

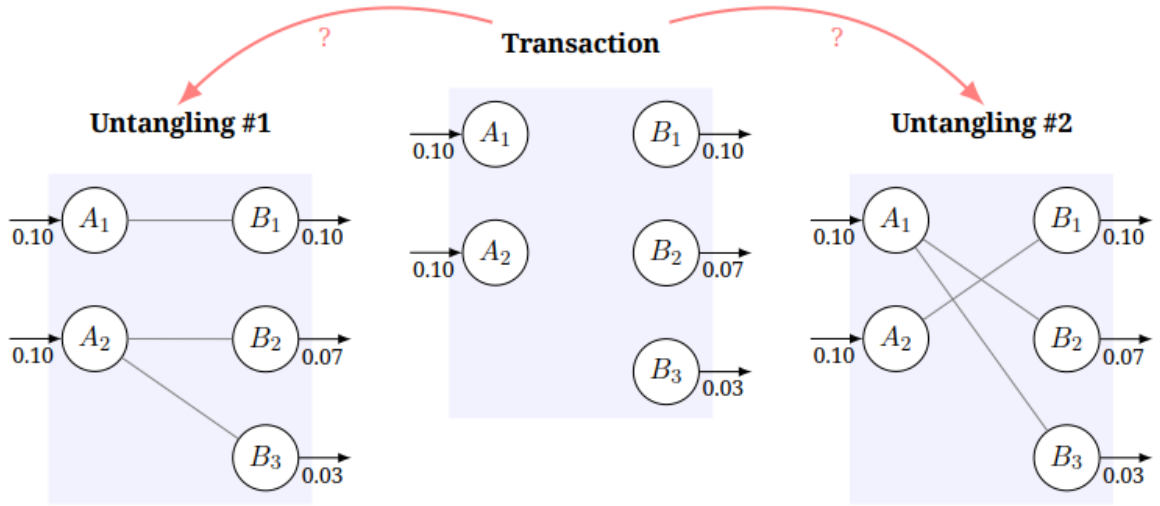


Transactions (2)

Separable transaction

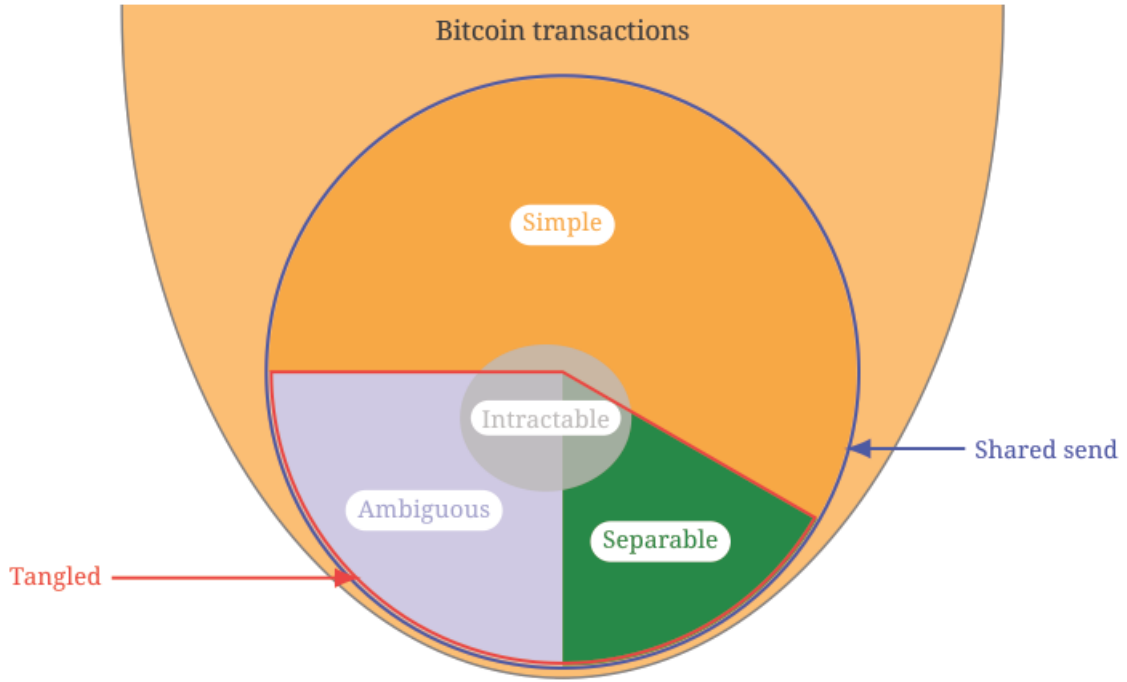


Ambiguous transaction (admits two different untangling outcomes)

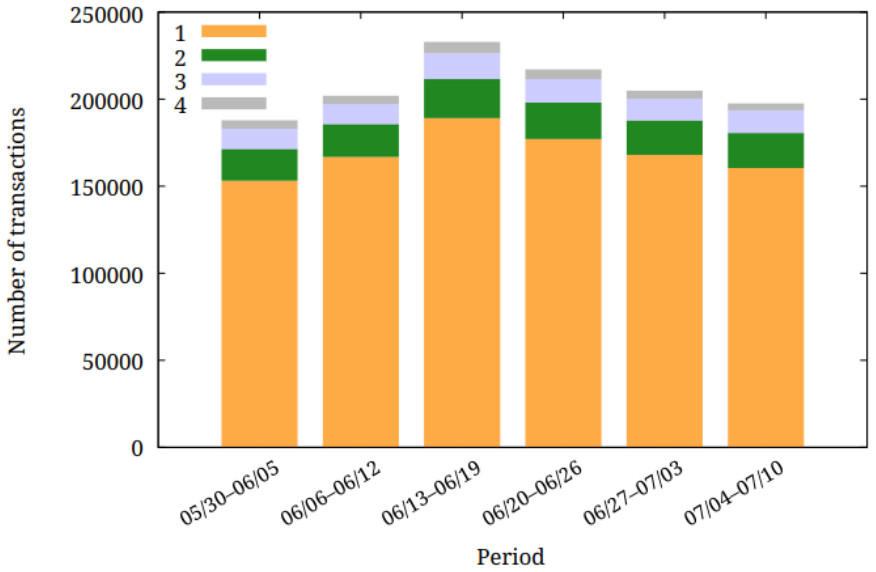


Transactions (3)

Theorem. The problem of detection of ambiguous shared send transactions is NP-complete (under some strict Definitions and Assumptions).



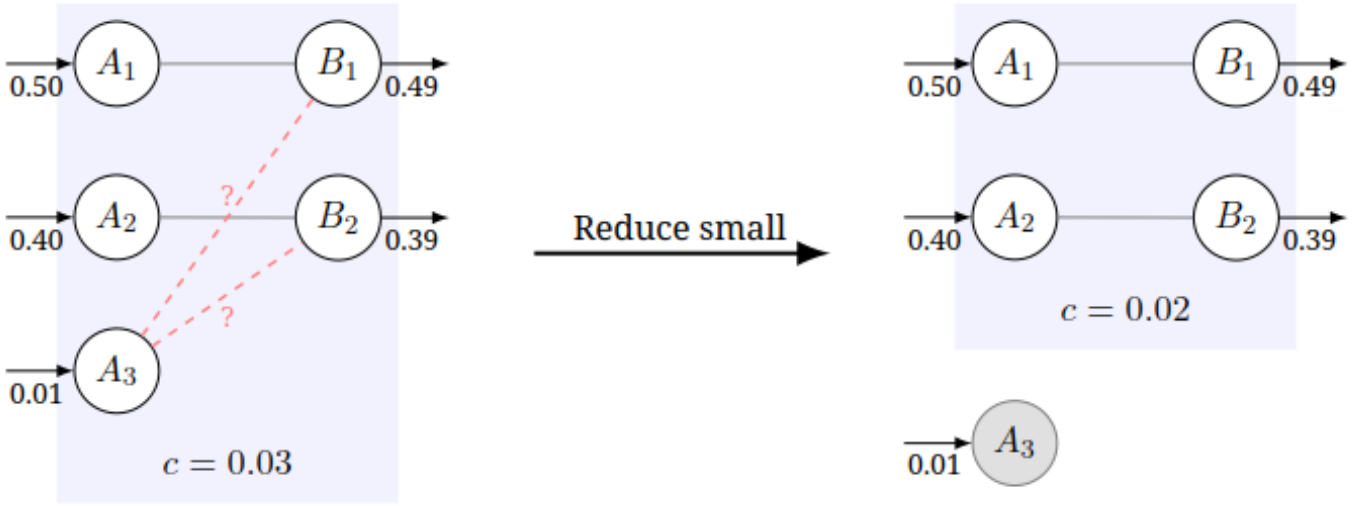
1 – simple;
2 – separable;
3 – ambiguous;
4 – intractable



Transactions (4)

Reality is more complicated!

- Grouping Addresses
- Discarding Small Inputs and Outputs
- Approximate Equalities



Cluster analysis

Assumptions

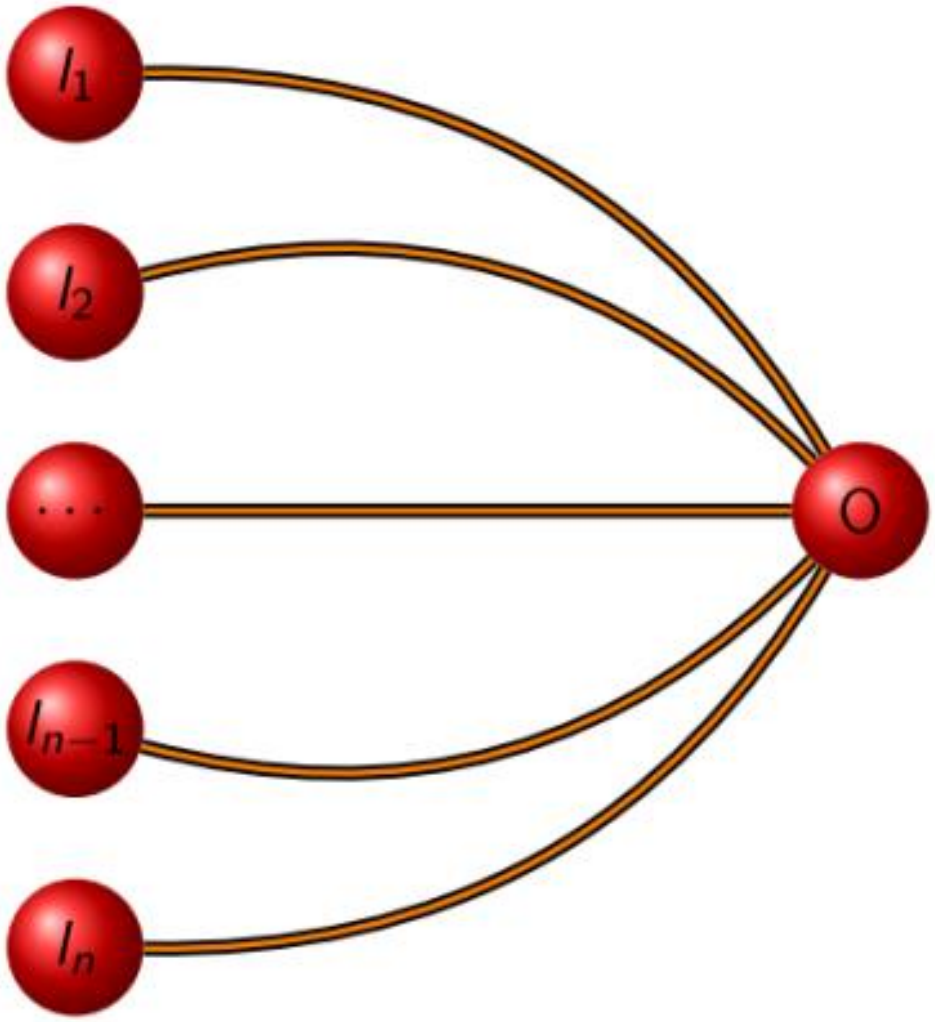
- Each Bitcoin address is controlled by a single real-world entity. Thus, we will ignore those sufficiently rare cases in which a multi-signature address is used for joint ownership of bitcoins, and not for multi-factor authentication.
- A single entity may control more than one address.

Goal: Bitcoin clustering algorithm

- minimal number of clusters
- all the addresses in each cluster are controlled by the same user.

Common Spending

If two or more addresses are inputs of the same transaction with one output, then all these addresses are controlled by the same user.

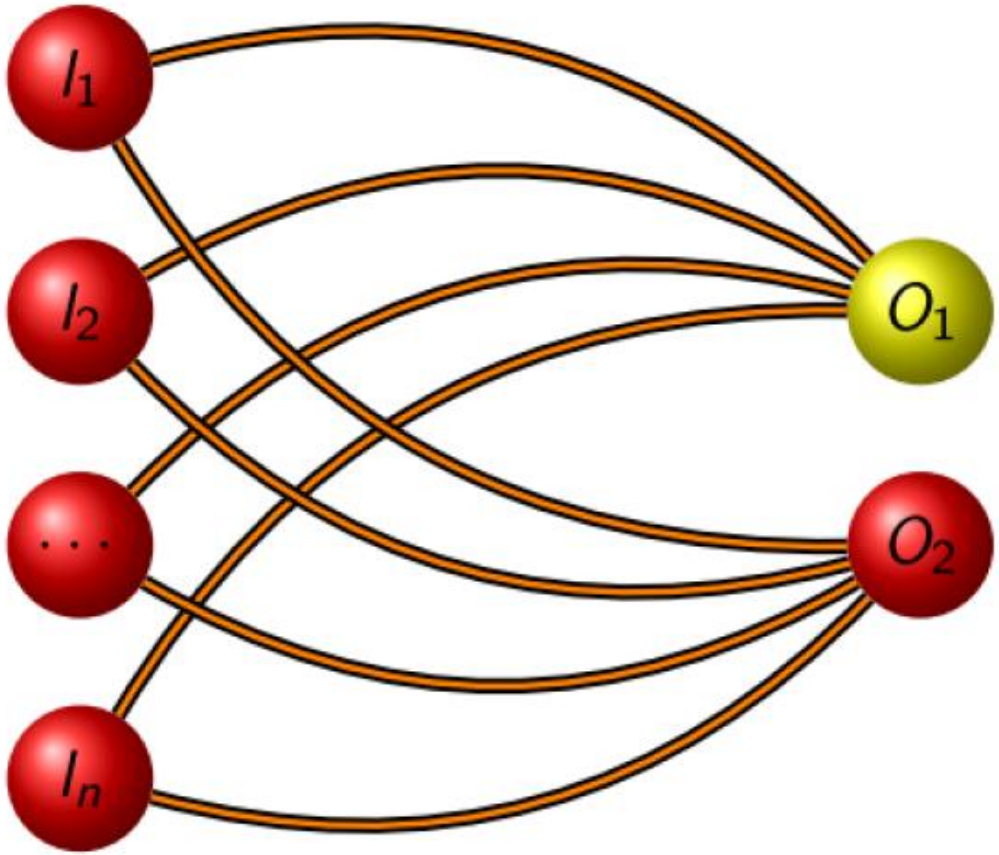


One-time Change

We say that the transaction t = (A, B, c) satisfies the condition of a one-time change if the following conditions hold.

1. #Addr(B) = 2, i.e. the transaction t has exactly ewo outputs.
2. #Addr(A) ≠ 2, i.e. the number of t inputs is not equal to two. If #Addr(A) = #Addr(B) = 2 the ransaction is most likely shared send mixer.
3. Both outputs of transaction t, B1 and B2, are not self-change addresses.
4. One output of the transaction B1 did not exist before transaction t and decimal representation of the value b1 has more than 4 digits after the dot.
5. The other output of the transaction B2 was previously part of the Bitcoin network and has not been OTC addressed in previous transactions.

The OTC output and all the inputs of the transaction arecontrolled by the same user.



Off-Chain information

Much public information can be found on the Internet (off-chain information).

If the Bitcoin address is mentioned in the same data frame with the tag (key phrase-entity, for example, company name or username), then it is said say that the address has such a tag.

Tag collection

- **passive approach:** web crawling of public forums and user profiles (for example, Bitcointalk.com, Twitter and Reddit) and Darknet markets (for example, Silkroad, The Hub Ьarketplace and Alphabay)
- **active approach:** manual analysis of Bitcoin companies and data actualization procedures (Satoshi Bones casino uses 1change and 1bones prefixes and BTC-E exchange uses 1eEUR and 1eUSD prefixes, etc.).

Possible categories of Bitcoin **organizations**

- mining pools (pools)
- exchanges
- Darknet markets (dnm)
- mixers
- gambling
- other services (services).

services	57
gambling	80
mixer	3
dnm	16
exchange	98
pool	52

TABLE I: Unique clean tags per category.

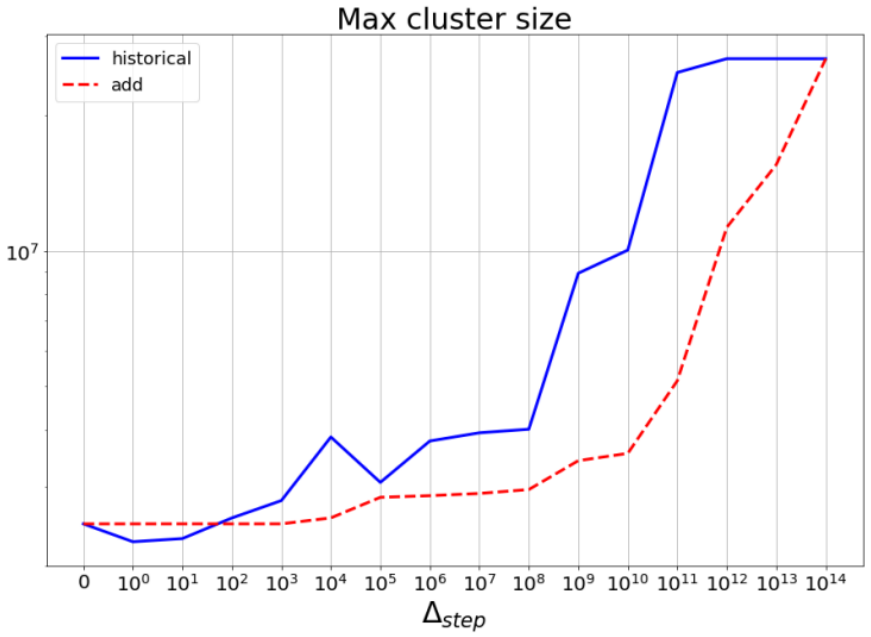
Cluster analysis (2)

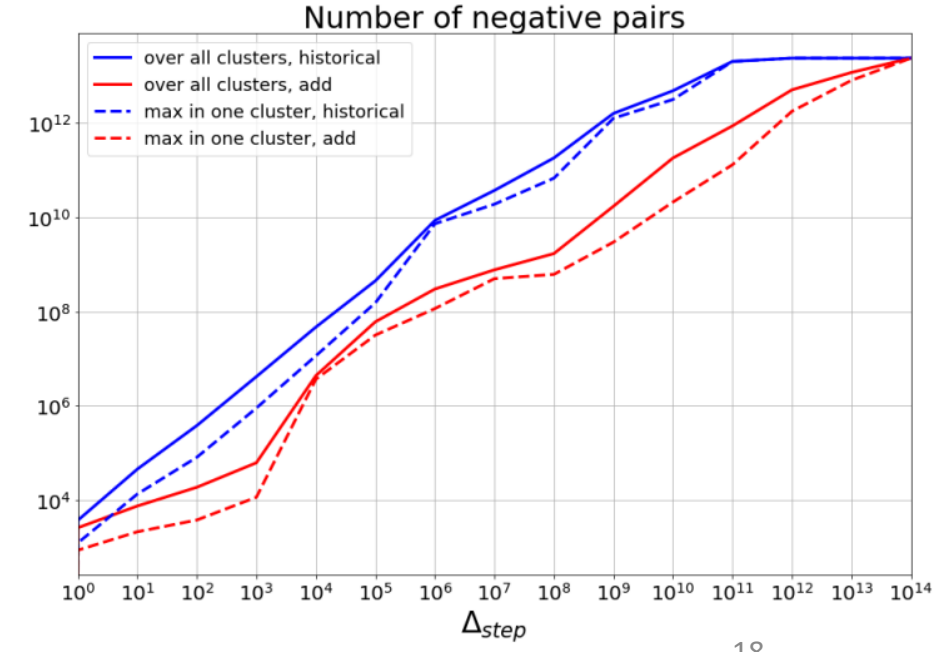
Experimental Setup

- Bitcoin blockchain data from 3d January of 2009 to 9th March of 2017: 211, 789, 876 transactions which cover 244, 030, 115 unique addresses.
- CS heuristic condition is satisfied for 8, 161, 086 transactions with 28, 416, 034 unique addresses.
- OTC heuristic condition holds for 35, 844, 487 OTC transactions with 69, 520, 194 unique addresses.
- Both conditions give a total of 44, 005, 573 covered transactions with 95.250.167 unique addresses (the overlap is 2, 686, 061 addresses).

Category	Number of tags	Number of common tags (size)	Examples of common tags
services	33	5 (> 100K)	*Bitpay.com, Xapo.com*
gambling	34	6 (> 50K)	*999Dice.com, primedice.com*
mixer	3	1 (> 100K)	*BitcoinFog*
dnm	14	5 (> 100K)	*SilkRoad Marketplace*
exchange	64	12 (> 100K)	*BTC-e.com, Bittrex.com*
pool	15	2 (> 50K)	*BTCChina, Hashnest.com*

Table: Tags of the biggest cluster in case of clustering without constraints (26, 694, 671 addresses).

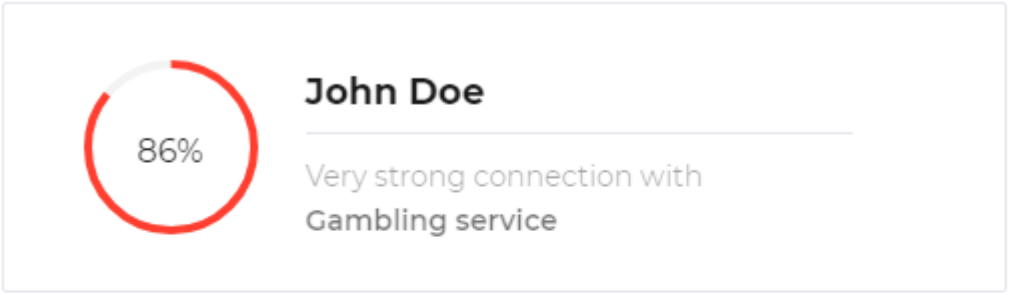


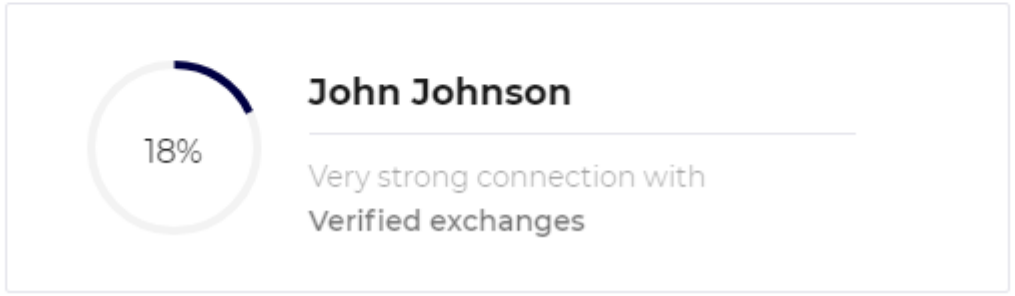


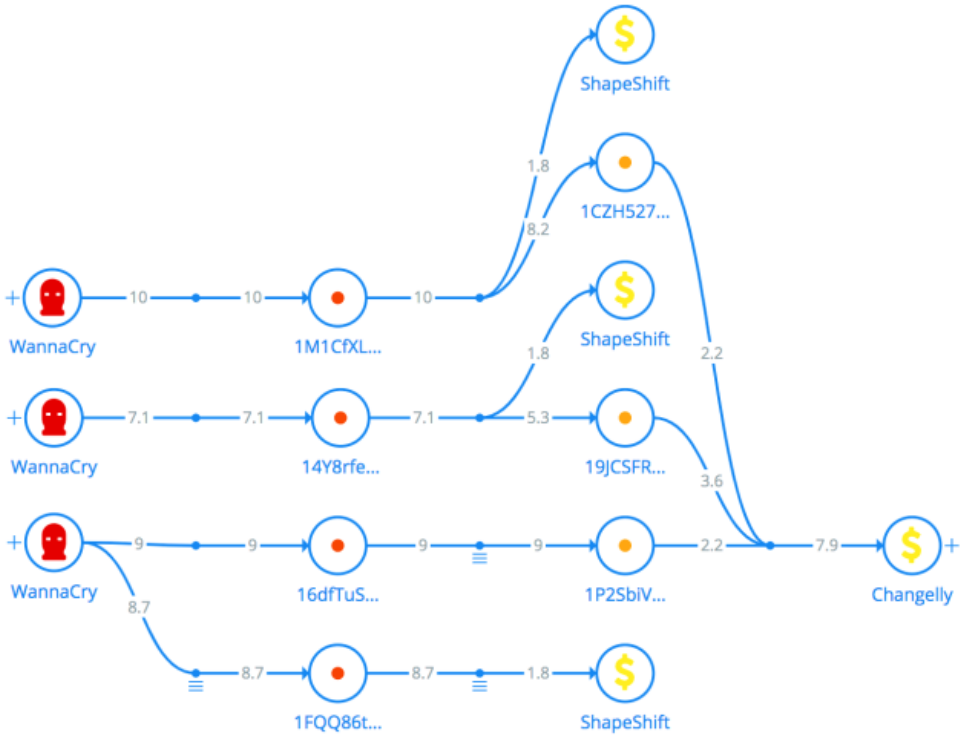
Other problems

- Classification
- Risk scoring

WannaCry attack
May 12, 2017



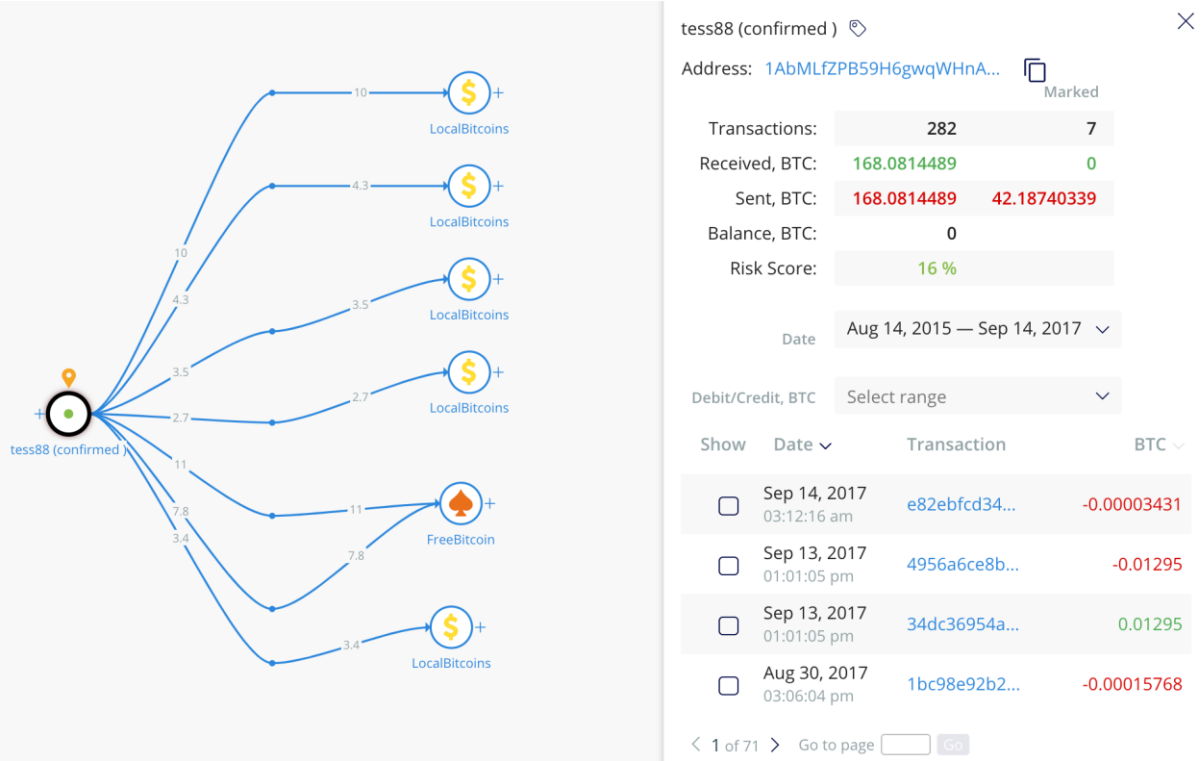




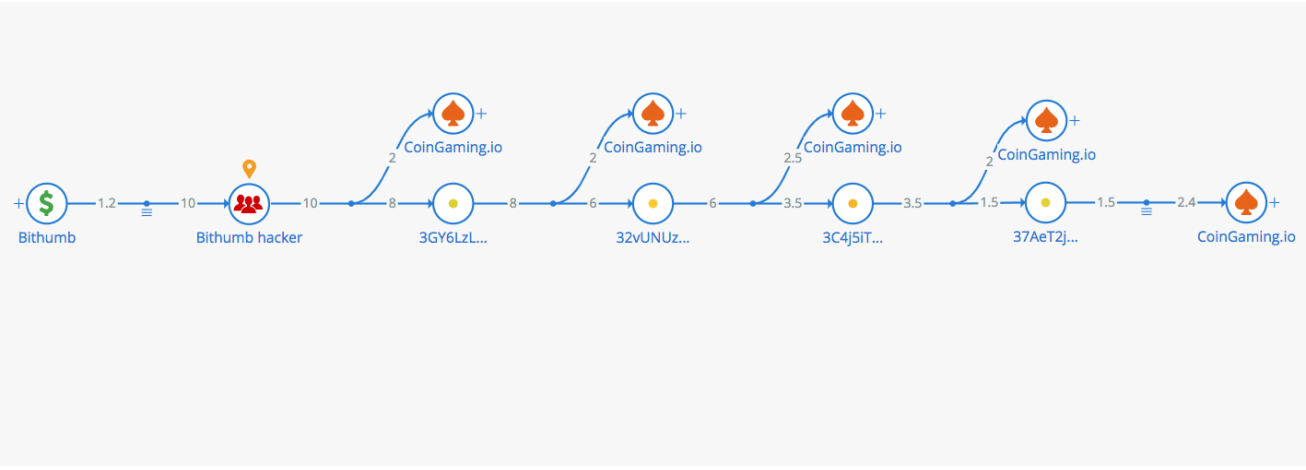
Other problems

- Risk scoring
- Classification

True Identity of Notorious Hacker tessa88 Revealed



Bithumb exchange hack in June 2018



3. Shield

Homomorphic encryption

Function E(x) of x satisfies the following properties:

- For most x, with a known value of E(x), finding x is a difficult task.
- Different input values result in different function values
  - for x≠y: E(x)≠E(y).
- If someone knows E (x) and E (y), then he can generate the HE from arithmetic operations for x and y. For example, he can calculate E(x + y), knowing E (x) and E (y).

Types

- additively homomorphic (RSA)
- multiplicative homomorphic (Paillier)
- fully homomorphic:
  - Automatic PhD problem by Dan Boneh
  - Craig Gentry solved in 2009.

Unpadded RSA

If the RSA public key is modulus m and exponent e, then the encryption of a message x is given by $\mathcal{E}(x) = x^e \mod m$. The homomorphic property is then

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = x_1^e x_2^e \mod m = (x_1 x_2)^e \mod m = \mathcal{E}(x_1 \cdot x_2)$$

ElGamal

In the ElGamal cryptosystem, in a cyclic group G of order q with generator g, if the public key is (G, q, g, h), where $h = g^x$, and x is the secret key, then the encryption of a message m is $\mathcal{E}(m) = (g^r, m \cdot h^r)$, for some random $r \in \{0, \ldots, q-1\}$. The homomorphic property is then

$$\mathcal{E}(m_1) \cdot \mathcal{E}(m_2) = (g^{r_1}, m_1 \cdot h^{r_1})(g^{r_2}, m_2 \cdot h^{r_2})$$

$$= (g^{r_1+r_2}, (m_1 \cdot m_2) h^{r_1+r_2}) = \mathcal{E}(m_1 \cdot m_2).$$

Paillier

In the Paillier cryptosystem, if the public key is the modulus m and the base g, then the encryption of a message x is $\mathcal{E}(x) = g^x r^m \mod m^2$, for some random $r \in \{0, \ldots, m-1\}$. The homomorphic property is then

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = (g^{x_1} r_1^m)(g^{x_2} r_2^m) \mod m^2 = g^{x_1+x_2}(r_1 r_2)^m \mod m^2 = \mathcal{E}(x_1 + x_2)$$

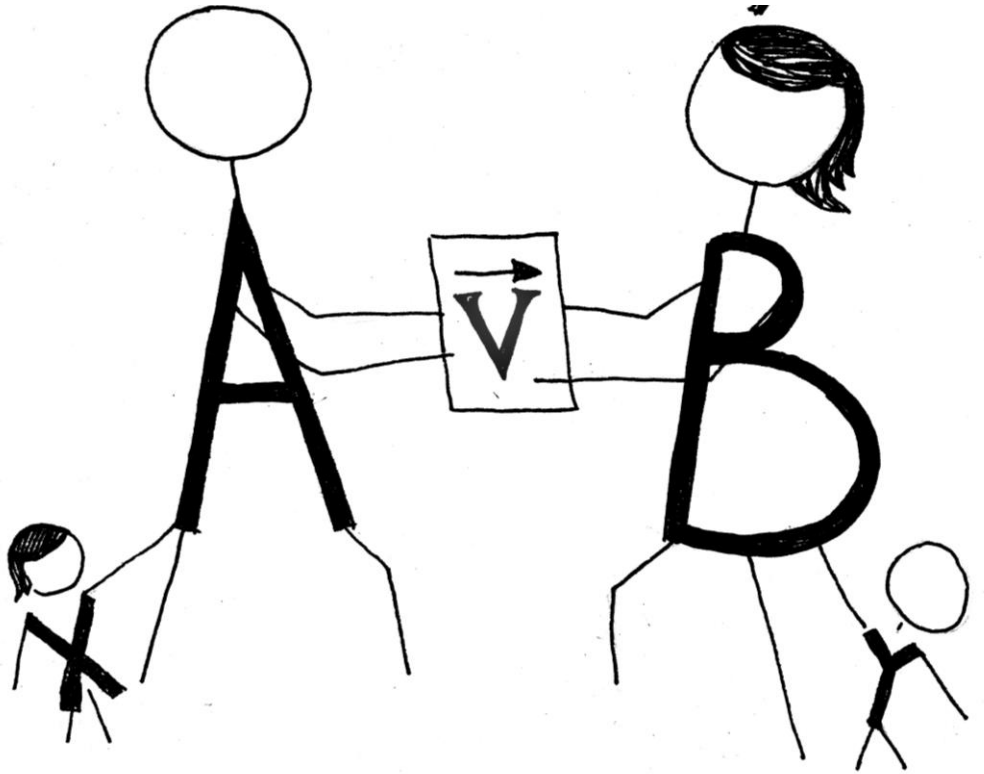
Fully Homomorphic encryption

Almost eigenvectors

- v is a secret
- values x, y are private
- Matrixes A, B are public
  - Av ~= x v
  - Bv ~= y v

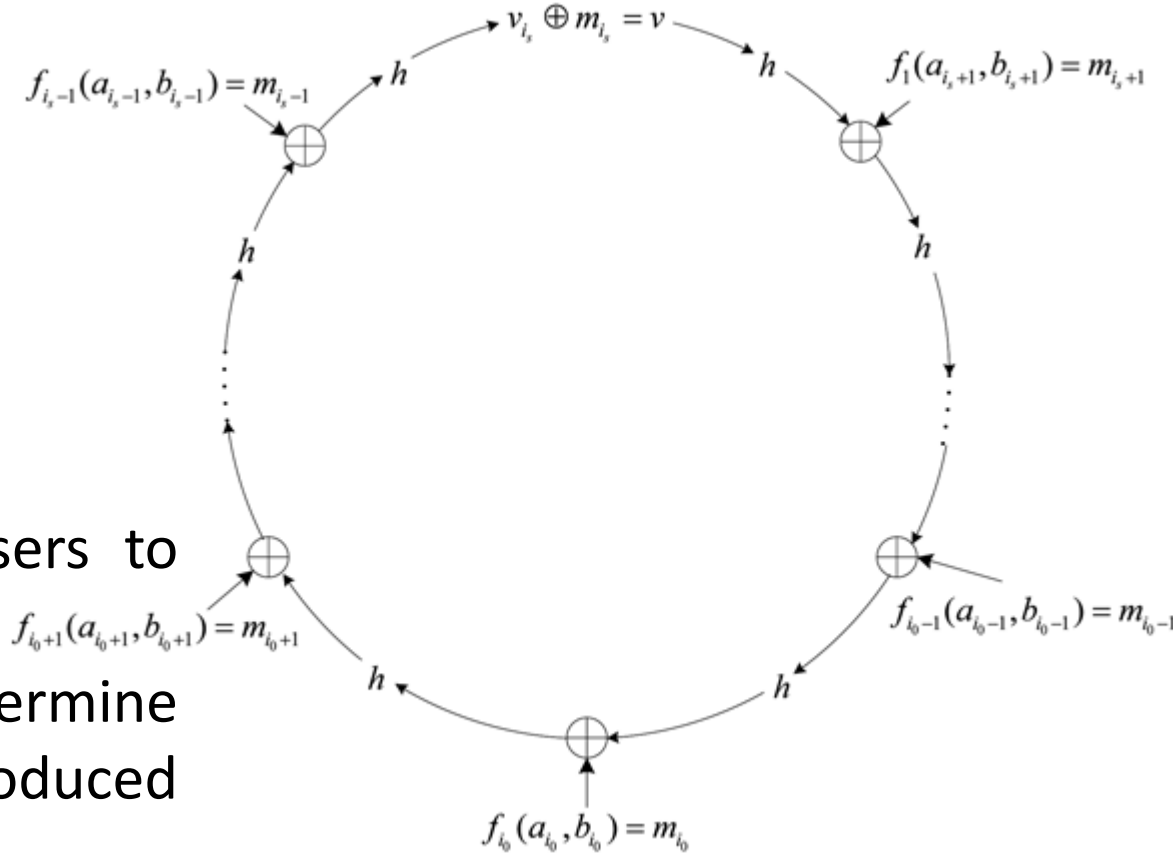
(AB)v = A (Bv) ~= A xv = x (Av) ~= (xy)v

(A + B)v = Av + Bv ~= xv + yv ~= (x + y)v



Ring Signatures

The message is signed by one of the members of the list of potential signatories without revealing who



- **Threshold ring signatures:** requires t users to cooperate in the signing protocol
- **Linkable ring signatures:** allows to determine whether any two signatures have been produced by the same member
- **Traceable ring signature:** the public key of the signer is revealed, if they issue more than one signatures under the same private key.

Ring Signatures (2)

[1] Back, A. “Ring signature efficiency.” Bitcointalk (2015)

[2] Noether, Shen, and Adam Mackenzie. "Ring confidential transactions." *Ledger* 1 (2016): 1-18.

GEN: Let G be the basepoint of cyclic group where the discrete logarithm assumption is assumed to hold (Monero currently uses Ed25519).[6] Find a number of public keys $P_i, i = 0, 1, \ldots, n$ and a secret index $j \in \{0, 1, \ldots, n\}$ such that $xG = P_j$ where G is the base-point and x is the signer's spend key. Let $I = xH_p(P_j)$ be the key image corresponding to P_j where H_p is a cryptographically secure hash function returning a point whose logarithm with respect to the base-point G is unknown.

SIGN: Let $\mathfrak{m}$ be a given message to sign (in practice, $\mathfrak{m}$ is a sha512 has of an arbitrary string. Let $\alpha, s_i,\ i \neq j,\ i \in \{1, \ldots, n\}$ be random values in $\mathbb{Z}_q$ (the Ed25519 base field).

Compute

$$L_j = \alpha G$$

$$R_j = \alpha H_p(P_j)$$

$$c_{j+1} = H_s(\mathfrak{m}, L_j, R_j)$$

where H_s is a cryptographic hash function returning a value in $\mathbb{Z}_q$. Now, working successively in j modulo n, define

Ring Signatures (3)

$$L_{j+1} = s_{j+1}G + c_{j+1}P_{j+1}$$

$$R_{j+1} = s_{j+1}H_p\left(P_{j+1}\right) + c_{j+1} \cdot I$$

$$c_{j+2} = H_s\left(\mathfrak{m},\, L_{j+1},\, R_{j+1}\right)$$

$$\ldots$$

$$L_{j-1} = s_{j-1}G + c_{j-1}P_{j-1}$$

$$R_{j-1} = s_{j-1}H_p\left(P_{j-1}\right) + c_{j-1} \cdot I$$

$$c_j = H_s\left(\mathfrak{m}, L_{j-1},\, R_{j-1}\right)$$

so that $c_1, \ldots, c_n$ are defined.

Let $s_j = \alpha - c_j \cdot x_j \; mod \; l$, ($l$ being the Ed25519 curve order) hence $\alpha = s_j + c_j x_j \; mod \; l$ so that

$$L_j = \alpha G = s_j G + c_j x_j G = s_j G + c_j P_j$$

Ring Signatures (4)

$$R_j = \alpha H_p(P_j) = s_j H_p(P_j) + c_j I$$

and

$$c_{j+1} = H_s(\mathfrak{m}, L_j, R_j)$$

and thus, given a single c_i value, the message $\mathfrak{m}$, the P_j values, the key image I, and all the s_j values, then all the other c_k, $k \neq i$ can be recovered by an observer. The signature therefore becomes:

$$\sigma = (I, c_1, s_1, ..., s_n)$$

which represents a space savings over CryptoNote,[22] where the ring signature would instead look like:

$$\sigma = (I, c_1, ..., c_n, s_1, ..., s_n)$$

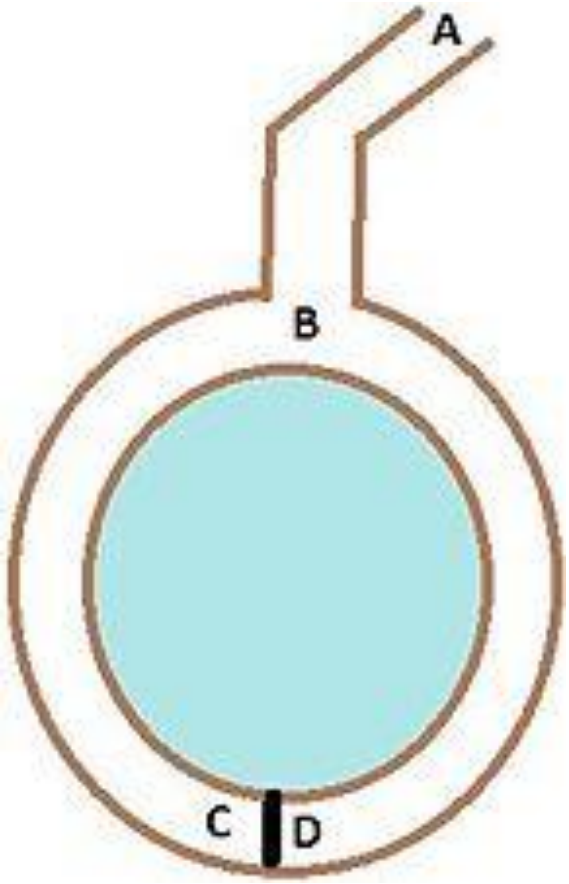
VER: Verification proceeds as follows. An observer computes L_i, R_i, and c_i for all i and checks that $c_{n+1} = c_1$. Then the verifier checks that

$$c_{i+1} = H_s(\mathfrak{m}, L_i, R_i)$$

for all $i \mod n$

Zero-knowledge proofs

The Ali Baba cave

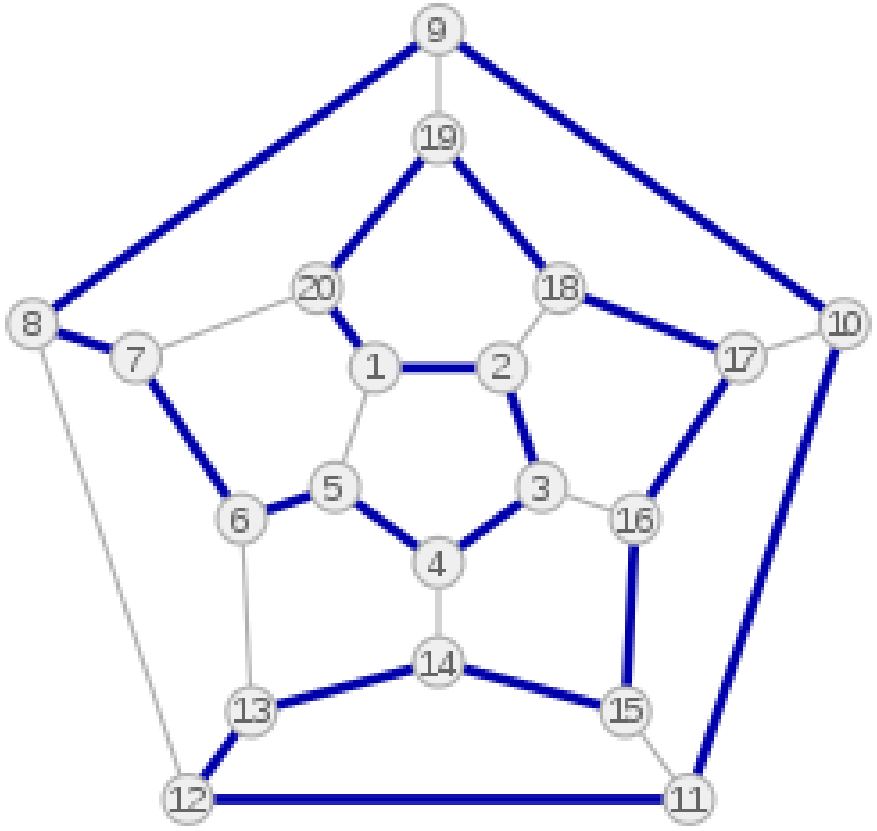


Probability cheat in
N iterations in a row:
$\mathbf{2^{-N}}$

Properties:

- **Completeness:** convince if the secret is known
- **Correctness:** no cheating prover can convince, except with some small probability
- **Zero-knowledge.**

Hamiltonian cycle and isomorphism for large graphs



ZK-Snarks

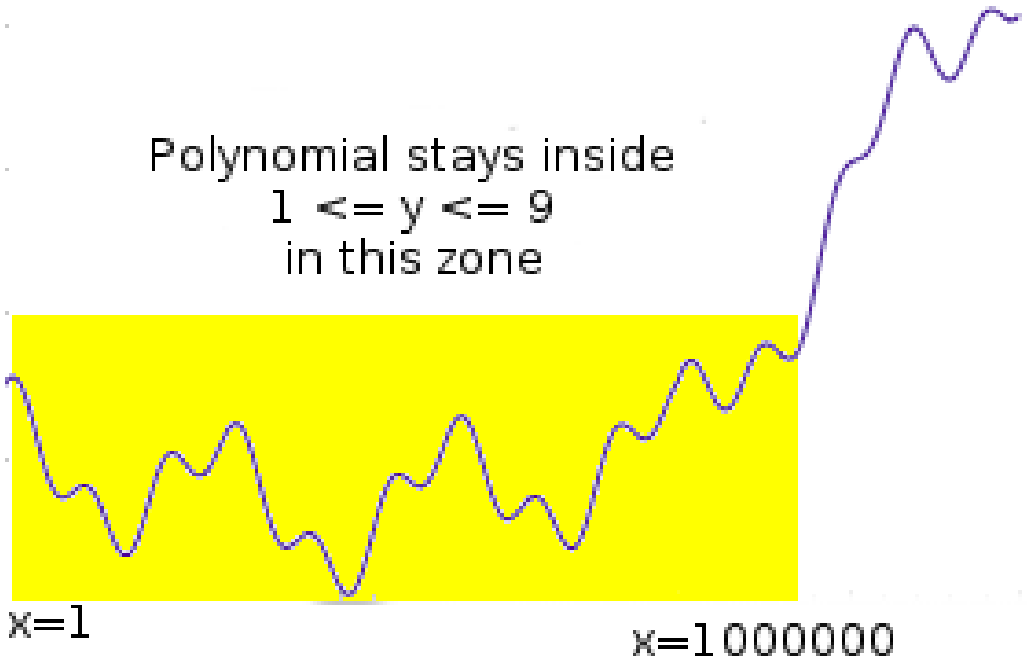
- https://z.cash/technology/zksnarks/
- https://medium.com/@VitalikButerin/zk-snarks-under-the-hood-b33151a013f6

1. Homomorphic Hiding
2. Blind Evaluation of Polynomials
3. The Knowledge of Coefficient Test and Assumption
4. How to make Blind Evaluation of Polynomials Verifiable
5. From Computations to Polynomials
6. The Pinocchio Protocol
   - Convert proofs into Quadratic Arithmetic Program
7. Pairings of Elliptic Curves
   - Tate reduced pairing

$$\forall a, b \in F_q^*, \ \forall P \in G_1, Q \in G_2 : \ e(aP, bQ) = e(P, Q)^{ab}$$

ZK-Starks

https://vitalik.ca/general/2017/11/09/starks_part_1.html

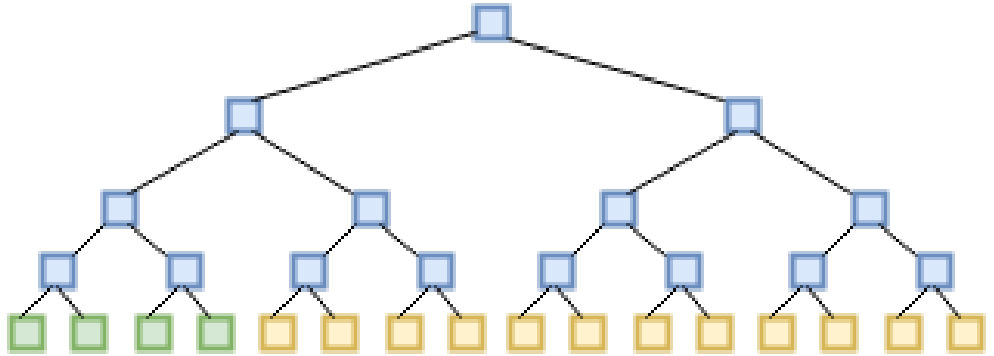


Constraint checking polynomial

- C(x) = x * (x-1) * (x-2) * ... * (x-9)

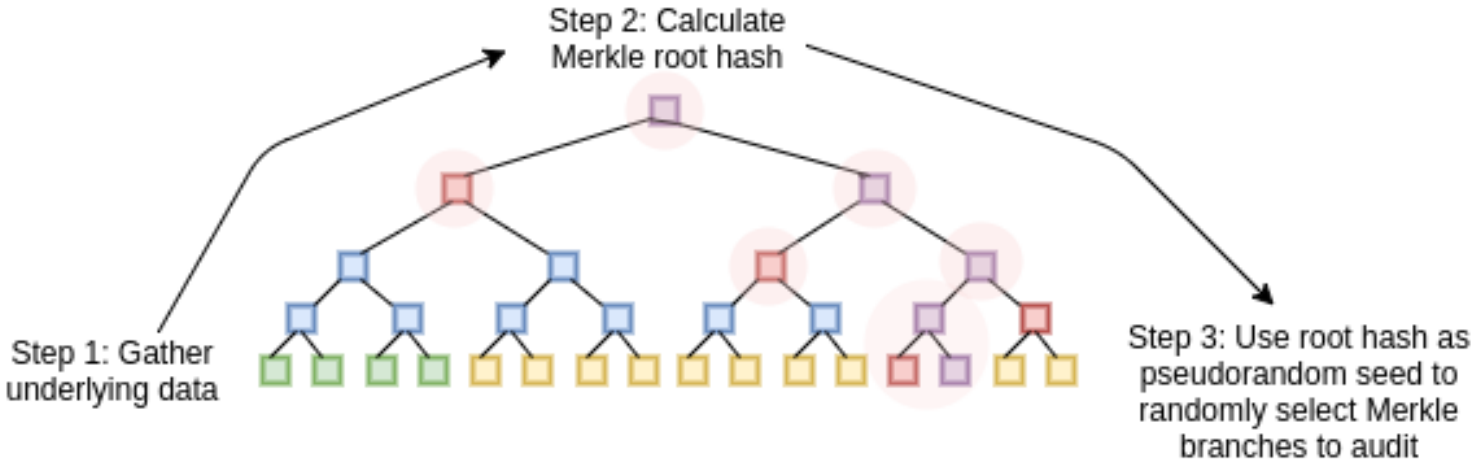
Prove that you know P such that C(P(x)) = 0 for all x from 1 to 1,000,000.

ZK-Starks (2)



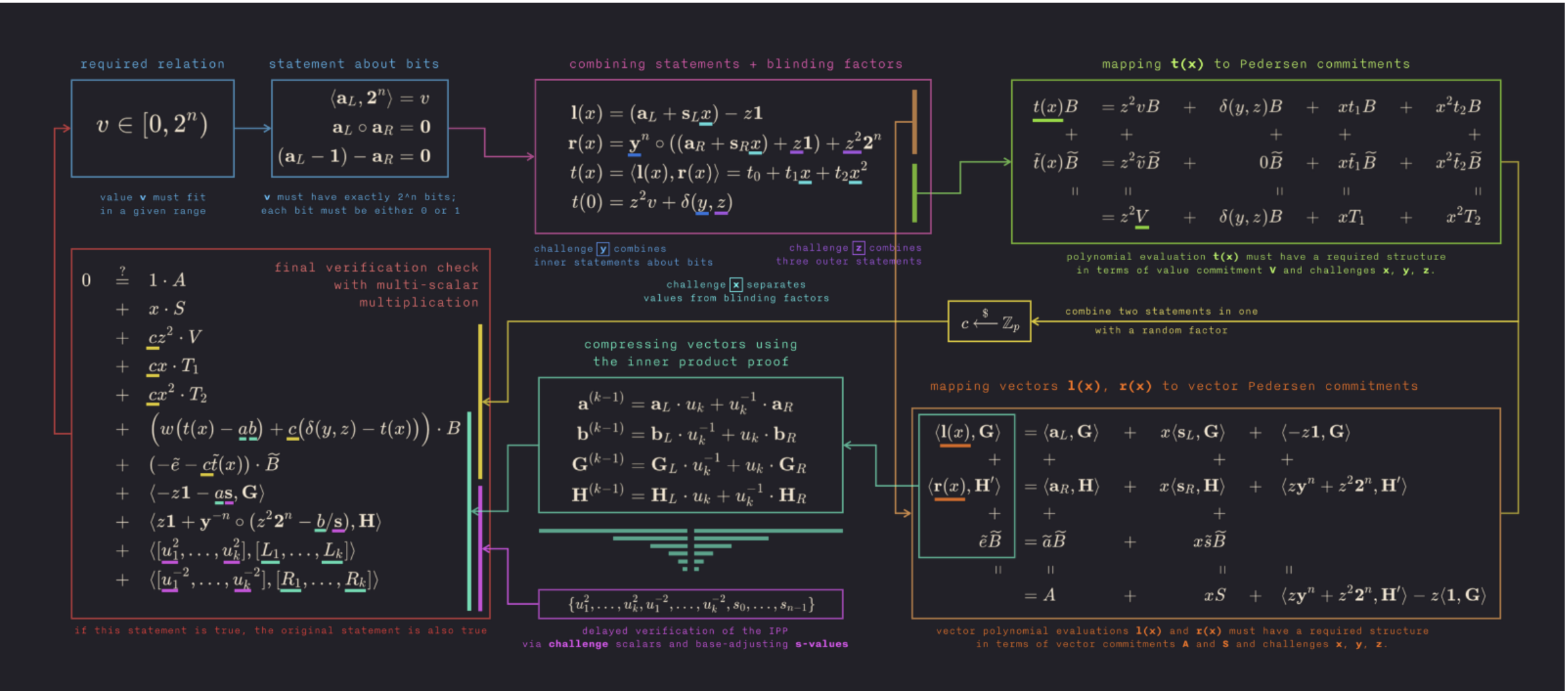
Values of P(x) here satisfy
1 <= P(x) <= 9

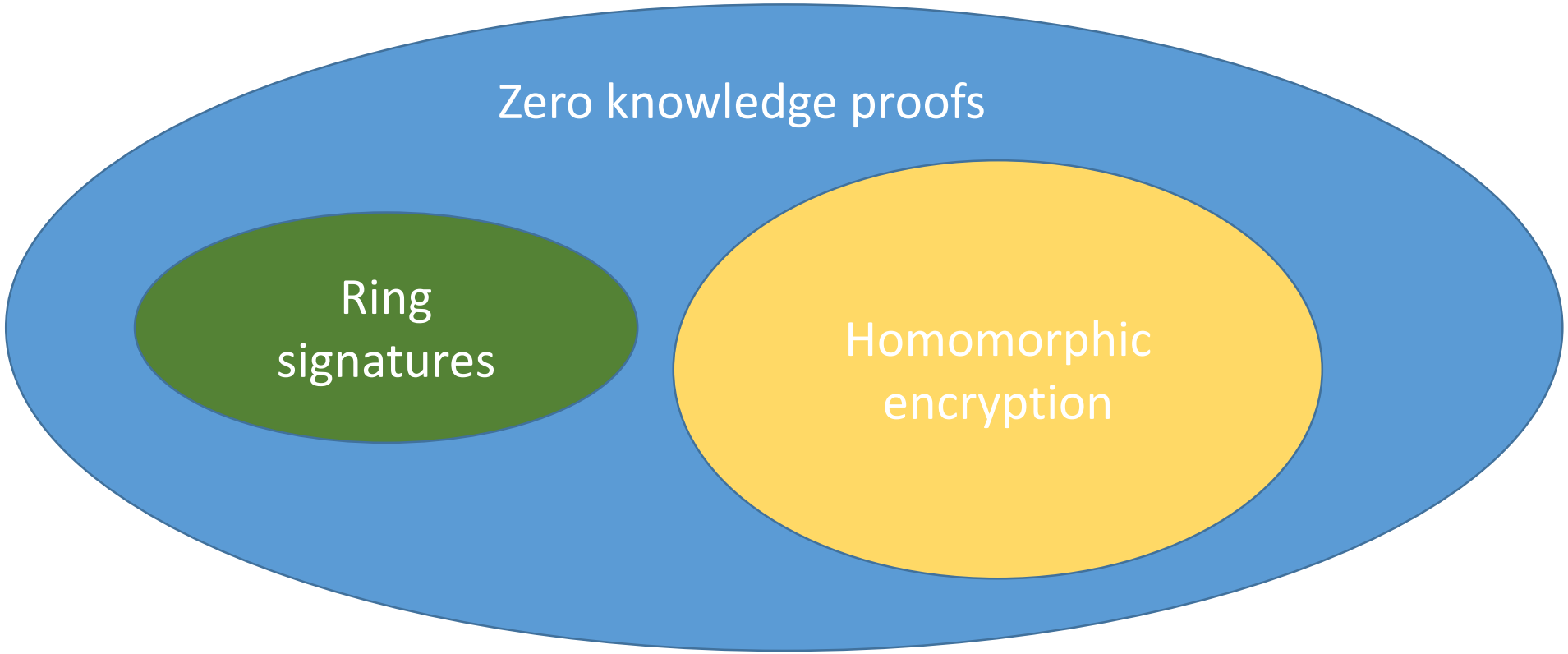
These values probably don't, but they're still a very important part of the proof



Bulletproofs

https://medium.com/interstellar/bulletproofs-pre-release-fcb1feb36d4b




Zero knowledge proofs
Ring signatures
Homomorphic encryption

Usage examples

- Cryptocurrencies
  - ZCash
  - Monero
- E-Voting
  - One token per person as predesign. Ring signatures to send it.
- Peer-to-peer random number generation
  - Private systems/KYC: secret sharing without dealer and homomorphic encryption
  - Public systems: verifiable delay functions/time-lock puzzles.

Code examples

- Homomorphic encryption
  - https://bitsofpy.blogspot.com/2014/03/homomorphic-encryption-using-rsa.html
- Ring signatures
  - https://github.com/boneyard93501/ring-sig
- ZKP bulletproofs
  - https://github.com/AdamISZ/bulletproofs-poc

Thank you for your attention!!!