Lecture 17:
Payment with Unlinkability

CPSC 4440/5440: Real World Cryptography

Instructor: Prof. Fan Zhang

Yale University

Spring 2026

Yale

Computer Science

Plan for the rest of semester

- This week: wrapping up identity & credentials
- Upcoming presentations
  - March 25th: presentations by Ben R and Amin
  - April 15th: Ian & Hadi
  - April 20th: Ekow
  - You are given two weeks to decide (by 3rd)
- Upcoming deadlines
  - project midterm report: come to OH to give me a brief update.
  - one more homework mid April
- April 6th is canceled due to NSF duties

Recap: identity & credentials

- Authentication is the first step of using any secure system
- Technical topics we've seen:
  - passwords, password-authenticated key exchange (PAKE)
  - OPAQUE: state of the art strong aPAKE
  - OPRF: a very useful tool
  - Federated authentication: SSO, OAuth
  - PASTA (CCS'18): improving SSO with threshold issuance [using threshold OPRF]
  - One-show credentials: e.g., privacy pass [OPRF or blind sigs]
- Today: a specific application: **electronic money with unlinkability**

Physical money v.s. digital money

- Reasonably hard to counterfeit
- Strong privacy
- Downside: can’t use online.

Physical money v.s. digital money

- Reasonably hard to counterfeit
- Strong privacy
- Downside: can't use over Internet.

PayPal

venmo

WeChat Pay

Alipay™

- Easy to use within the same platform
- Impossible to counterfeit
- Privacy??

Can we get cash-like privacy?

- Bank notes (cash):
  - Verifying authenticity doesn’t require a third party
  - Double spending is impossible
  - If I spent $10 dollar at Tea Shop, no body needs to know that
- With digital money:
  - Authority keeps track of balances (e.g., bank, PayPal)
  - When spending, Tea Shop contacts authority to check my balance; I authenticate to move money.
  - Authority learns (me, Tea Shop, $100)
- Does this problem ring a bell?

Setup of e-Cash

- **Bank** trusted for everything but privacy (i.e., honest but interested in tracking users)
- Bank's "API" include…
- **Mint**: Alice deposits $ and gets "digital cash"
  - Users may open accounts non-anonymously (e.g., they are required by law to show an ID)
- **Spend**: Alice sends "digital cash" to Bob online
- **Redeem**: Bob submits "digital cash" to Bank and gets $

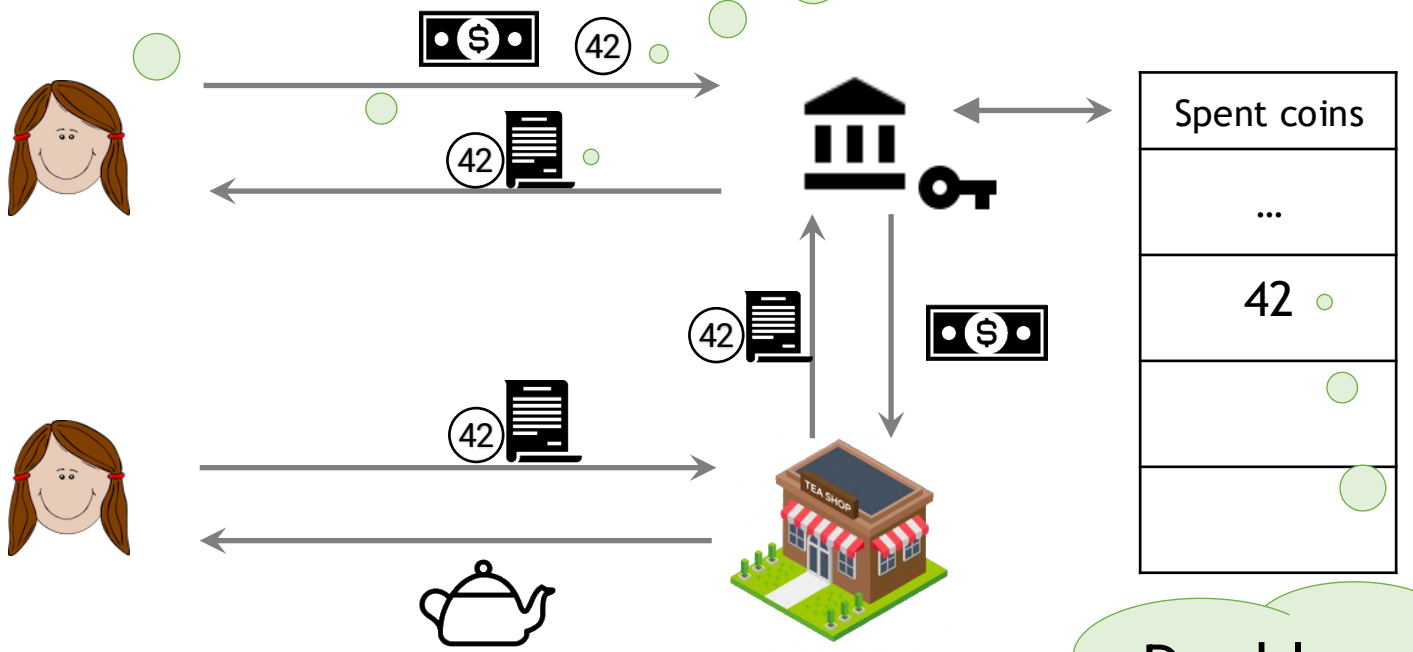
Goals

- Prevent counterfeiting
- Prevent double spending
- Cash-like privacy: prevent bank from tracking

Warmup: Centralized e-cash via signatures (no privacy)

Counterfeiting is prevented by digital sigs

User chosen serial number



Double spending is prevented

Untraceable Electronic Cash †

(Extended Abstract)

David Chaum[1] *Amos Fiat*[2] *Moni Naor*[3]

One of two key ideas: use blind signatures (introduced last time)

E.g., RSA signature (‘77)

- *RSA assumption*: ***Given input y, hard to take cube root***, i.e., hard to find x s.t.

$$x^3 = y \bmod N$$

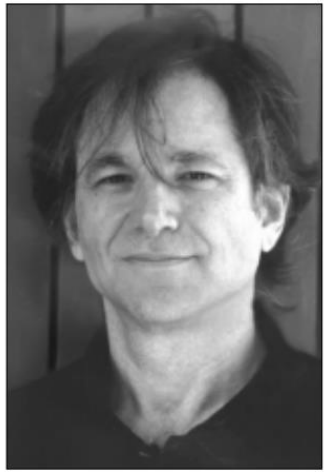
Unless one can find factors of N, or have a quantum computer.

Yale’69 (BA)

Rivest, Shamir, and Adleman Receive 2002 Turing Award

Ronald L. Rivest

Adi Shamir



Leonard M. Adleman

E.g., RSA signature

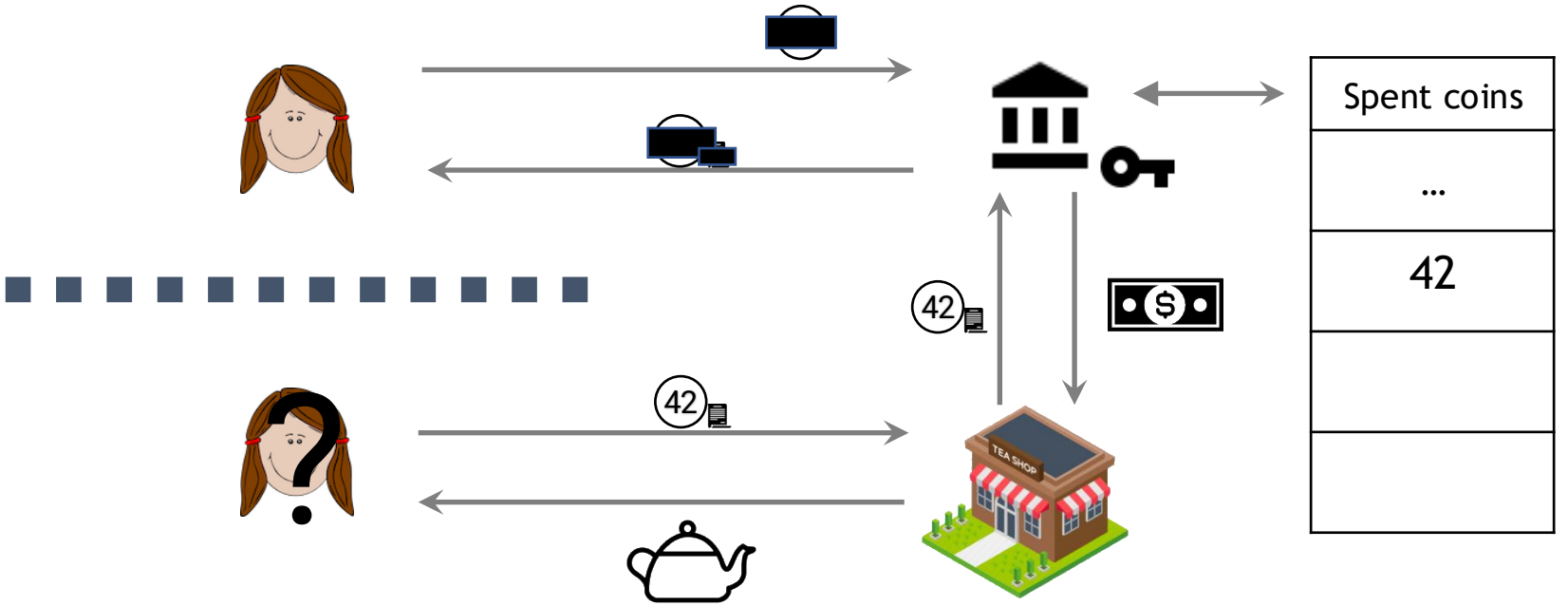
- **KeyGen()**: sample p, q two random primes
  - PK: $N = pq$
  - SK: (p, q)
- **Sign**(*SK, m*) = $H(m)^{\frac{1}{3}} \bmod N$
- **Verify**(*m, σ, PK*):
  checks $\sigma^3 = H(m) \bmod N$

Why hash?
O.w. Easy to forge valid (message, signature) pair $(m = \sigma^3, \sigma)$

Blind Signature from RSA

- Alice chooses a random SN and a *random blinding factor r*
- Alice sends $H(sn) \times r^3$ to the bank
- Alice gets back $H(sn)^{\frac{1}{3}} \times r$, from which she can extract $H(sn)^{\frac{1}{3}}$

e-cash via blind signatures



Double spending

- We saw the **online protocol** for double spending prevention: Merchant calls the bank to make sure sn hasn't been spent for every txn.
- This is undesirable in practice: The merchant may go offline (e.g., at a farmer’s market with poor cellular coverage)
- Can this be done **offline**?
  - i.e., without bank in the loop

The 2nd nice ideas in e-cash

- Encode identification information (e.g., SSN, name, or anything the bank can hold a user accountable) in coins
- encoded info is hidden *unless* a user spend a coin more than once

- Money = $\prod_{i=1}^{k/2} H(x_i, y_i)^{1/3}$

- k: security parameters affects the chance of a cheater succeeding

Double spend prevention in e-cash

- Money = $\prod_{i=1}^{k/2} H(x_i, y_i)^{1/3}$

- $x_i = H(a_i)$ where a_i is a random serial number chosen by Alice
- $y_i = H(a_i \oplus ID)$ where ID is Alice's identification info (same as her registration)

- $\oplus$ = XOR
- Goal: when Alice double-spends, both a_i and $a_i \oplus ID$ will be revealed.

- $x_i = H(a_i)$ where a_i is a random serial number chosen by Alice
- $y_i = H(a_i \oplus ID)$ where ID is Alice's identification info

Bob (merchant)'s protocol

1. Bob receives Money = $\prod_{i=1}^{k/2} H(x_i, y_i)^{1/3}$
2. Bob flips k/2 coins and asks Alice to reveal pre-image of x_i or y_i

0	1	2	…	k/2
				
1	1	0		1
(a_0, y_0)	(a_1, y_1)	$(x_2, a_2 \oplus ID)$		$(a_{k/2}, y_{k/2})$

3. Verify Money = $\prod_{i=1}^{k/2} H(x_i, y_i)^{1/3}$ is properly formed.
4. Bob later deposits the money and passes along all the info.

Double spend prevention in e-cash

0	1	2	…	k/2
				
Bob 1	1	0		1
	(a_0, y_0)	(a_1, y_1)	$(x_2, a_2 \oplus ID)$	$(a_{K/2}, y_{K/2})$
Charlie 1	0	0		1
		$(x_1, a_1 \oplus ID)$		

When Bob and Charlie deposit coins, Bank catches Alice for double spending.

Not done yet

- $x_i = H(a_i)$ where a_i is a random serial number chosen by Alice
- $y_i = H(a_i \oplus ID)$ where ID is Alice's identification info

- How does bank make sure Alice has embedded the correct ID?
- Alice chooses k random SNs $a_1, \ldots, a_k$
- … sends k numbers $H(x_i, y_i)$ to bank [omitting blinding factors]
- Bank asks Alice to open a random half. Verify if any numbers are ill-formed. Sign the rest.
- Bank catches Alice with overwhelming probability. Yet Alice will can use the rest w/ privacy.

Cut and choose

The commercialization of e-cash

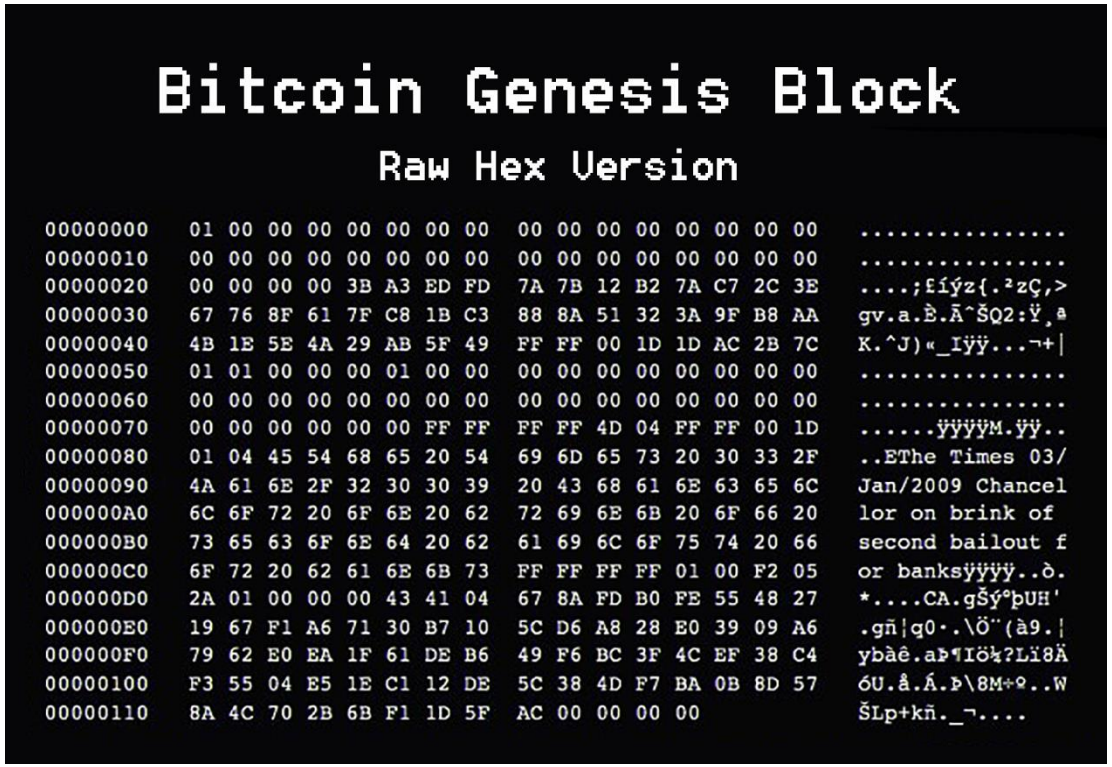
- DigiCash Inc. was founded in 1989 on the idea of e-cash
- In USA: only one bank (the Mark Twain bank in Saint Louis, MO) implemented e-cash, and attracted 5,000 users in a three-year trial.
- Went bankrupt in 1998, despite flourishing electronic commerce, but with credit cards as the “currency of choice”
- David Chaum opined then “*As the Web grew, the average level of sophistication of users dropped. It was hard to explain the importance of privacy to them*”

Problem for innovation

- We trust banks (maybe because we have to)
- But unlikely to trust a *new* bank
- Even less likely to trust a new bank with ”private” ”digital” ”money”

- **Finding trusted/centralized parties is a major obstacle**

In 2009, Satoshi figured out how to build payments without trusted parties



Bitcoin Genesis Block

Raw Hex Version

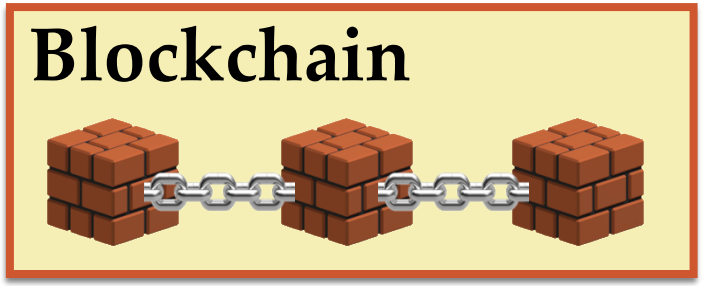
```
00000000   01 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000010   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000020   00 00 00 00 3B A3 ED FD   7A 7B 12 B2 7A C7 2C 3E   ....;£íýz{.²zÇ,>
00000030   67 76 8F 61 7F C8 1B C3   88 8A 51 32 3A 9F B8 AA   gv.a.È.Ã^ŠQ2:Ÿ¸ª
00000040   4B 1E 5E 4A 29 AB 5F 49   FF FF 00 1D 1D AC 2B 7C   K.^J)«_Iÿÿ...¬+|
00000050   01 01 00 00 00 01 00 00   00 00 00 00 00 00 00 00   ................
00000060   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000070   00 00 00 00 00 00 FF FF   FF FF 4D 04 FF FF 00 1D   ......ÿÿÿÿM.ÿÿ..
00000080   01 04 45 54 68 65 20 54   69 6D 65 73 20 30 33 2F   ..EThe Times 03/
00000090   4A 61 6E 2F 32 30 30 39   20 43 68 61 6E 63 65 6C   Jan/2009 Chancel
000000A0   6C 6F 72 20 6F 6E 20 62   72 69 6E 6B 20 6F 66 20   lor on brink of
000000B0   73 65 63 6F 6E 64 20 62   61 69 6C 6F 75 74 20 66   second bailout f
000000C0   6F 72 20 62 61 6E 6B 73   FF FF FF FF 01 00 F2 05   or banksÿÿÿÿ..ò.
000000D0   2A 01 00 00 00 43 41 04   67 8A FD B0 FE 55 48 27   *....CA.gŠý°þUH'
000000E0   19 67 F1 A6 71 30 B7 10   5C D6 A8 28 E0 39 09 A6   .gñ¦q0·.\Ö¨(à9.¦
000000F0   79 62 E0 EA 1F 61 DE B6   49 F6 BC 3F 4C EF 38 C4   ybàê.aÞ¶Iö¼?Lï8Ä
00000100   F3 55 04 E5 1E C1 12 DE   5C 38 4D F7 BA 0B 8D 57   óU.å.Á.Þ\8M÷º..W
00000110   8A 4C 70 2B 6B F1 1D 5F   AC 00 00 00 00            ŠLp+kñ._¬....
```

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

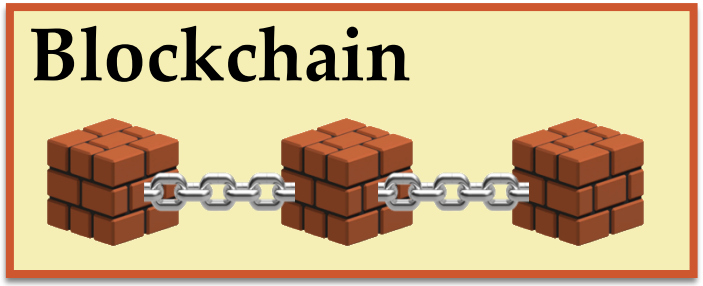
Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Crash course on Bitcoin



- Each coin has a specific “value”, and an “owner” (think of a public key)
  - Bitcoin supports rather elaborate ownership beyond public keys
- Blockchain records a list of transactions
  - transaction = coins burnt and created, plus signatures by the owners of the input coins

How private is Bitcoin?



- Pseudonymity (e.g., like Reddit)
- In theory stronger than Reddit
  - Like creating a new account for every post on Reddit: user can have an arbitrary number of PKs.
- In reality: assume no privacy
  - it's possible to link keys to same user due to user behaviors
  - with enough data from blockchain, exchanges, merchants, and even forum, de-anonymizing is practical

Same for many cryptocurrencies

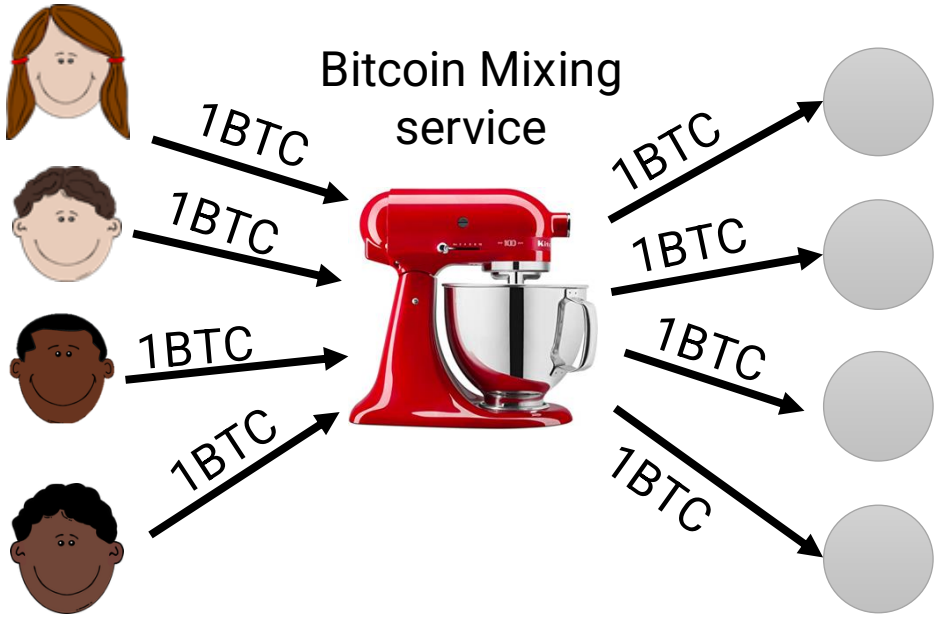
- Smart contracts platforms (e.g., Ethereum) offer strong security properties
  - **Transparency**: state & code fully open
  - **Integrity**: execution is tamper-resistant
  - **Availability**: running 24/7, accessible by everyone

“World computer” in the sky

- Still the same privacy guarantees (i.e., pseudonymity)

- E-cash requires trusted bank, which is perhaps hard to find.
- Bitcoin (and cryptocurrencies) doesn’t require trusted bank, but also isn’t private.
- Is there a way to make Bitcoin more private?

Fundamental idea: Mixing

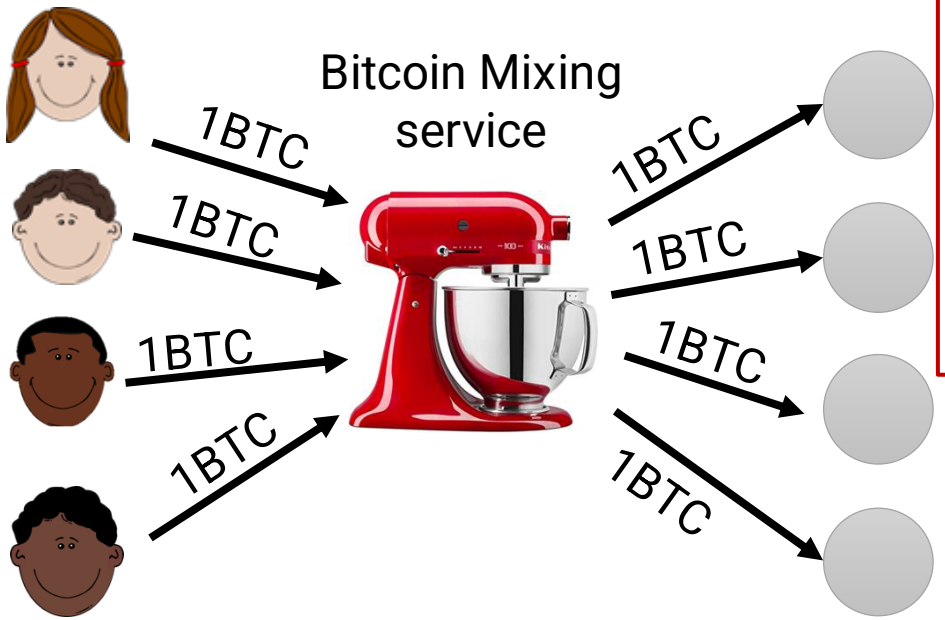


Each output links to each input with **equal probability ¼**

1. Each user sends (input coin, *fresh* address) to the service
2. The service creates a transaction with N input coins and N fresh outputs in a *random* order
3. Each user sends the signature for her input

https://en.bitcoin.it/wiki/CoinJoin

Fundamental idea: Mixing



Drawbacks

- Service trusted for privacy
- Anonymity-latency tension: stronger anonymity requires longer wait

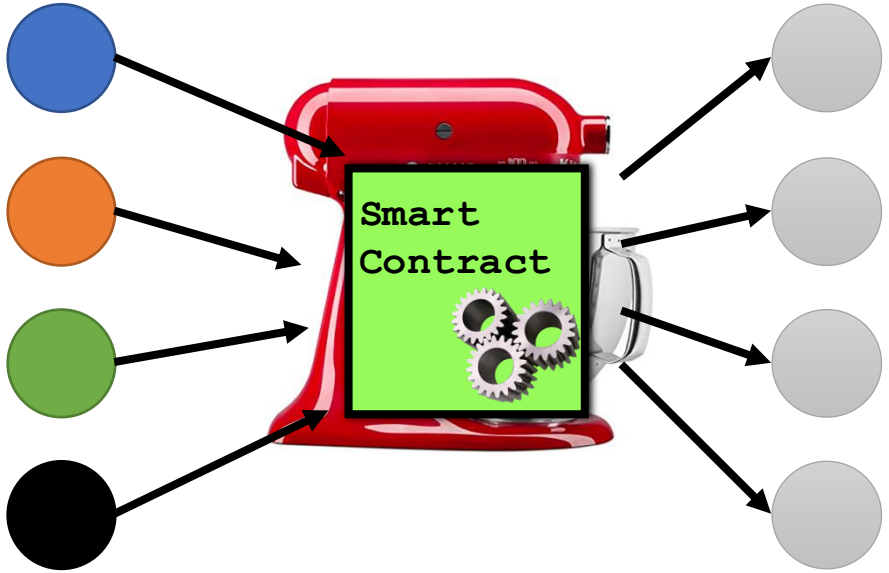
- Drawbacks?

https://en.bitcoin.it/wiki/CoinJoin

Removing trust:
Mixing with a *smart contract*

Setup

- Smart contract (the mixer) maintains
  - A list of deposits
  - A list of spent coins



Smart Contract State
(in practice, the Merkle roots)

Deposit
CM_1
CM_2
...

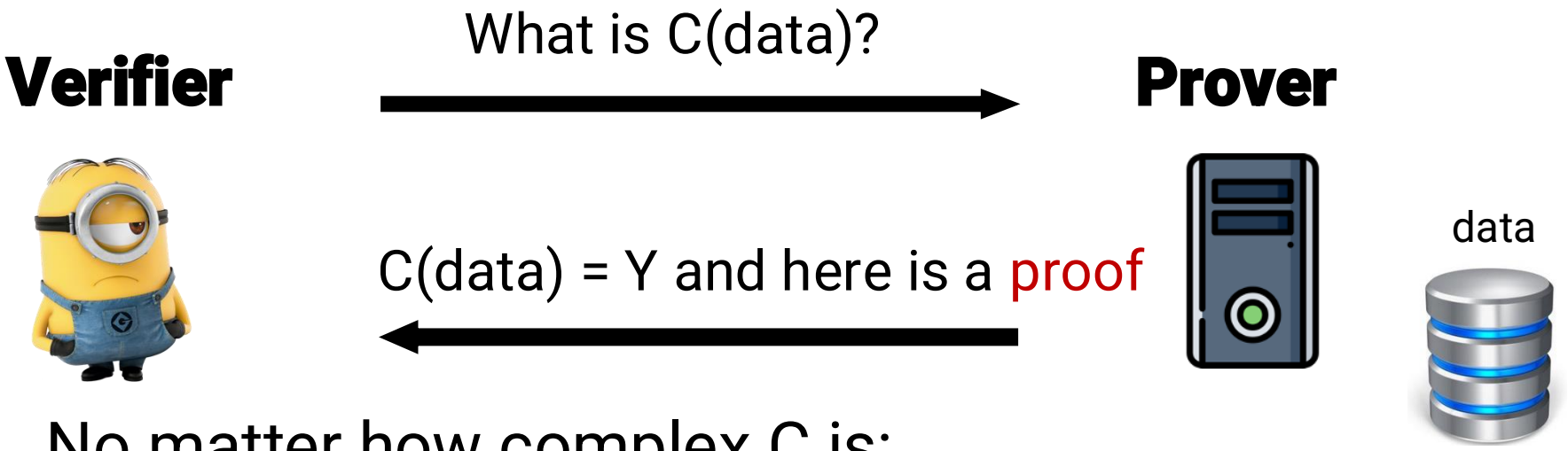
Spent
SN_1
sN_2
...

Commitments

- Like a secret note sealed in an envelope
  - Commit(s, r)->c
  - Open(c, s, r) –> {0,1}
- Two properties: hiding and binding
  - **Hiding**: c does not reveal anything about s (Note: (c, r) does)
  - **Binding**: a given c cannot be opened to two different s
- Examples of Commit(v, r)
  - H(v || r)
  - $g^r h^v$ (Pedersen commitment)

One minute intro to SNARK

- SNARK: Succinct Non-interactive ARgument of Knowledge



No matter how complex C is:

- Small proof (eg a few KB)
- Verifying proofs is blazingly fast

Catch: generating proof can be slow (e.g., hours)

In a bit of detail

- Public input X, output Y
- Witness w: known only to prover
- Function/circuit C(X,w)
- Prover want to prove that she knows w such that C(X;w)=Y
  - E.g., “I know the pre-image of X” can be encoded with C(X;w)=SHA256(w)-X

- Verifier can verify π against C,X,Y (won’t learn w)
- Soundness: prover cannot convince verifier unless she *knows* w
  - defining what it mean for someone to know something is a challenge
- zero-knowledge: proof does not leak any information about w

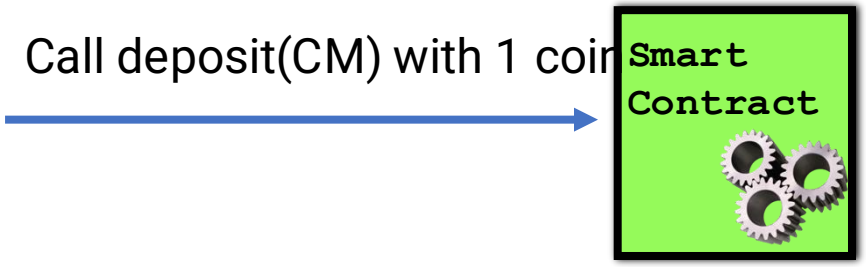

tornado
CASH

Deposit coins by creating a commitment

Sample random SN, r

CM = hash(SN || r)

Call deposit(CM) with 1 coin



Deposit	Spent
CM_1	
CM_2	
...	
CM_N	

Withdraw Mixed Coins by opening a commitment

Sample random SN, r

CM = hash(SN || r)

Call deposit(CM) with 1 coin

Smart Contract

Deposit	Spent
CM_1	SN
CM_2	
CM_N	

To prevent double spending

???

π_c, SN

1 coin

$\pi_c =$ I know ***r*** and SN s.t. H(*SN || r*) is in the Deposit list

Privacy:
To an observer, the spender could be *any* depositor (they form the *anonymity set*).

- Breaks the linkability between withdrawal & deposits

- Does not support payments
  - payee must get r & SN out of band
  - payer can spend before the payee can.
- Fixed value

Next: Full-blown private payments.

2014 IEEE Symposium on Security and Privacy

Zerocash: Decentralized Anonymous Payments from Bitcoin

Eli Ben-Sasson*, Alessandro Chiesa†, Christina Garman‡, Matthew Green‡, Ian Miers‡, Eran Tromer§, Madars Virza†
*Technion, eli@cs.technion.ac.il
†MIT, {alexch, madars}@mit.edu
‡Johns Hopkins University, {cgarman, imiers, mgreen}@cs.jhu.edu
§Tel Aviv University, tromer@cs.tau.ac.il

Zcash

(Note that Zerocash predates Tornado Cash by 6 years)

Challenges in enabling anonymous payments

- Basic idea: spending a commitment creates *new commitments*

So far: CM = hash(SN || r)

- How to create new commitments?
  - In particular, how to create SNs?
- How to enable payment of arbitrary value?
- How to validate solvency (output sum < input sum) when both are hidden?

New SNs

- Suppose Alice wants to pay Bob. Alice needs to create a new coin with a new SN.
- B should not be actively involved (payee may not be online)
- A can’t determine the SN by herself (why?)
  - A can trace B
  - A can create a spent SN so that B can’t spend
  - A can spend before B can
- Solution: A contributes randomness ρ
  - $sn = H(\rho|sk_B)$; Only B knows sk_B (and thus SN)

Coins

Each coin is represented by a commitment to

- owner's PK (only the owner knows the SK)
- value v
- ρ: a randomness to determine its serial number

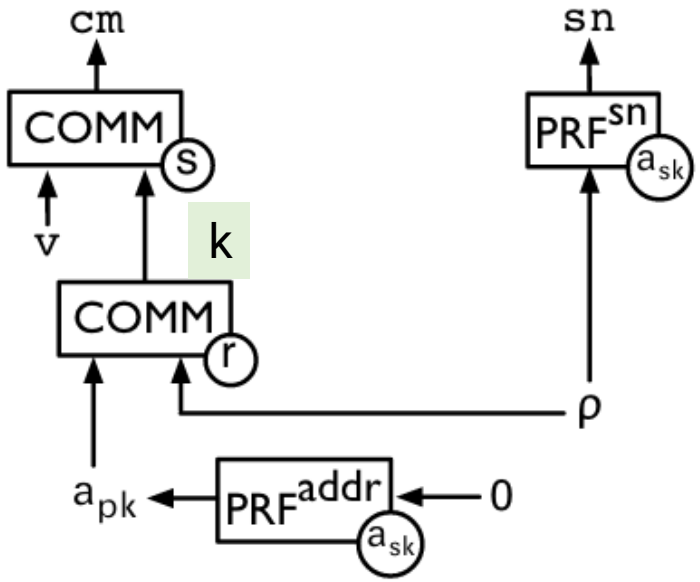


Fig. 1 in Zerocash paper.

Known only to the owner

Technically, a coin is (pk, v, ρ, r, s, cm)

- r, s are secret blinding factors
- $k = Com_r(pk|\rho)$
- $cm = Com_s(v|k)$

- K commits to only PK and ρ
- Useful when minting (next slide)

Mint zerocash (from Bitcoin)

- To mint an anonymous coin of v BTC, user sends v BTC to an escrow with (v, k, s, cm) in transaction metadata (e.g., OP_RETURN).
  - v: the value
  - k commits to PK, ρ
  - cm commits to v and k
- **cm now represents v "anonymous BTC"**
- This "escrow" can be implemented by modifying Bitcoin protocol or building an overlay on top of Bitcoin

- Zcash took the first approach.

Zerocash: Alice Pays Bob $3

blockchain

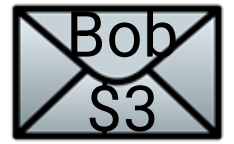
Serial #'s	31	29	34	11	7	8	6	17				
Coins												

$c_{in} = (pk^{\text{old}}, \$10, \rho^{\text{old}}, r^{\text{old}}, s^{\text{old}}, cm^{\text{old}})$

Suppose $sn = H(\rho, sk^{old}) = 42$

C_{in} 

1. Alice samples new ρ_1, r_1, s_1
2. creates $k_1 = Com(pk_B|\rho_1)$ and $cm_1 = Com(\$3|k_1)$
3. New coin is $c_{out_1} = (pk_B, \$3, \rho_1, r_1, s_1, cm_1)$



$Cout_1$

1. Alice samples new ρ_2, r_2, s_2
2. creates $k_1 = Com(pk_A|\rho_2)$ and $cm = Com(\$7|k_1)$
3. New coin is $c_{out_2} = (pk_A, \$7, \rho_2, r_2, s_2, cm_1)$

$Cout_2$

Zerocash: Alice Pays Bob $3

blockchain

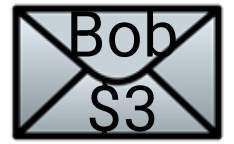
Serial #'s	31	29	34	11	7	8	6	17				
Coins	✉	✉	✉	✉	✉	✉	✉	✉				

$c_{out_1} = (pk_B, \$3, \rho_1, r_1, s_1, cm_1)$

$c_{in} = (pk^{\text{old}}, \$10, \rho^{\text{old}}, r^{\text{old}}, s^{\text{old}}, cm^{\text{old}})$

Suppose $sn = H(\rho, sk^{old}) = 42$

C_{in} 



$Cout_1$

π =

witness:

$(sk^{old}, c_{in}, c_{out_1}, c_{out_2})$

Public input:

$(cm_1, cm_2, 42)$

- Three coins are well formed
- sk^{old} matches pk^{old}
- $42 = H(\rho^{old}, sk^{old})$
- cm^{old} is in the list of coins
- $v_1 + v_2 \leq v^{old}$ (in c_{in})

$c_{out_2} = (pk_A, \$7, \rho_2, r_2, s_2, cm_1)$

$Cout_2$

Zerocash: Alice Pays Bob $3

blockchain

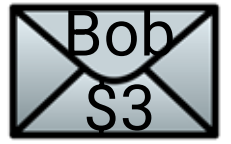
Serial #'s	31	29	34	11	7	8	6	17				
Coins												

$c_{out_1} = (pk_B, \$3, \rho_1, r_1, s_1, cm_1)$

$c_{in} = (pk^{\text{old}}, \$10, \rho^{\text{old}}, r^{\text{old}}, s^{\text{old}}, cm^{\text{old}})$

Suppose $sn = H(\rho, sk^{old}) = 42$

C_{in}



$Cout_1$

- Finally, Alice sends $\text{tx}_{pour} = (\pi, sn^{old}, cm_1, cm_2)$ to blockchain
- Send c_{out_1} to Bob in private because Bob needs ρ_1, r_1, s_1 to spend cm_1

$c_{out_2} = (pk_A, \$7, \rho_2, r_2, s_2, cm_1)$

$Cout_2$

That’s pretty much how Zerocash works…

- Some technical details omitted (e.g., non-malleab
- Zcash is the commercial implementation of zeroc

Summary:

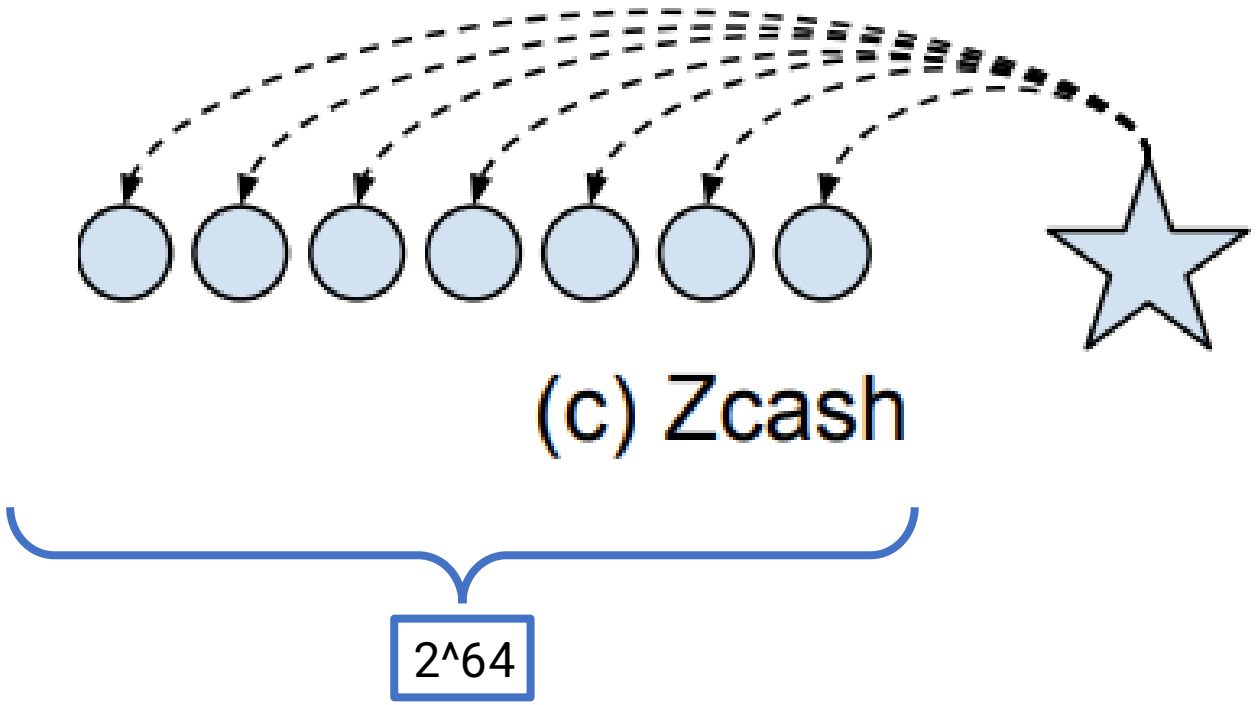
- coins are commitment to (PK, v, p)
  - PK: owner
  - v: value
  - p: randomness to derive SN
- spending = reveal SN & proving SK
- transfer involves spending and creating new commitments

Technically

- a coin is (pk, v, ρ, r, s, cm)
  - where r, s are random blinding factors
  - $k = Com(pk|\rho)$ and $cm = Com(v|k)$
- To mint an anonymous coin of v BTC, user sends v BTC to an escrow with (v, k, s, cm) in transaction metadata.
- To spend, create new coins and send $\text{tx}_{pour} = (\pi, sn^{old}, cm_1, cm_2)$

What is the anonymity set of Zerocash?

Anonymity set is the entire user universe



Any questions?