Thirty Years of Digital Currency: From DigiCash to the Blockchain

Matthew Green
Johns Hopkins University

My background

- Prof. at Johns Hopkins University
- Mainly work in applied cryptography (TLS, messaging systems, privacy-preserving protocols)
- I write a blog
- Co-founded a cryptocurrency (Zcash) and boy was that weird

Why talk about cryptocurrency?

Cryptocurrencies Aren't 'Crypto'

As the price of Bitcoin and Ethereum skyrocket, and more and more people who are unfamiliar with technology join in the craze, words start to lose their original and correct meaning.

SHARE TWEET


2. Oct
13. Nov
25. Dec
5. Feb
19. Mar
30. Apr
2015
2016
2017
2018

- **Whether you love it or hate it…**
  - Cryptocurrencies are exerting a massive influence on our field
  - The first major exposure to cryptography
  - That's both a good thing and a bad thing
    - The good: we get to deploy some really exciting new cryptography
    - The bad: if you stare into the abyss…

This talk

- A bit of history (payments & cryptocurrency)
- Some of the exciting practical directions being investigated today
- Some of the most exciting research directions (both in currency and outside of currency)
- With an admitted focus on privacy problems
- *Postscript: Some random bad crypto in cryptocurrency*

1980s-2007
(or: *how we got PayPal*)

UNITED STATES
FEDERAL RESERVE SYSTEM

1980s: Retail Payments

- **Goal: Digital payment system that**
  - Allows payments between customers and merchants (c2m)
  - Or between individual customers (c2c)
- **Strong cryptographic security**
- **Privacy**

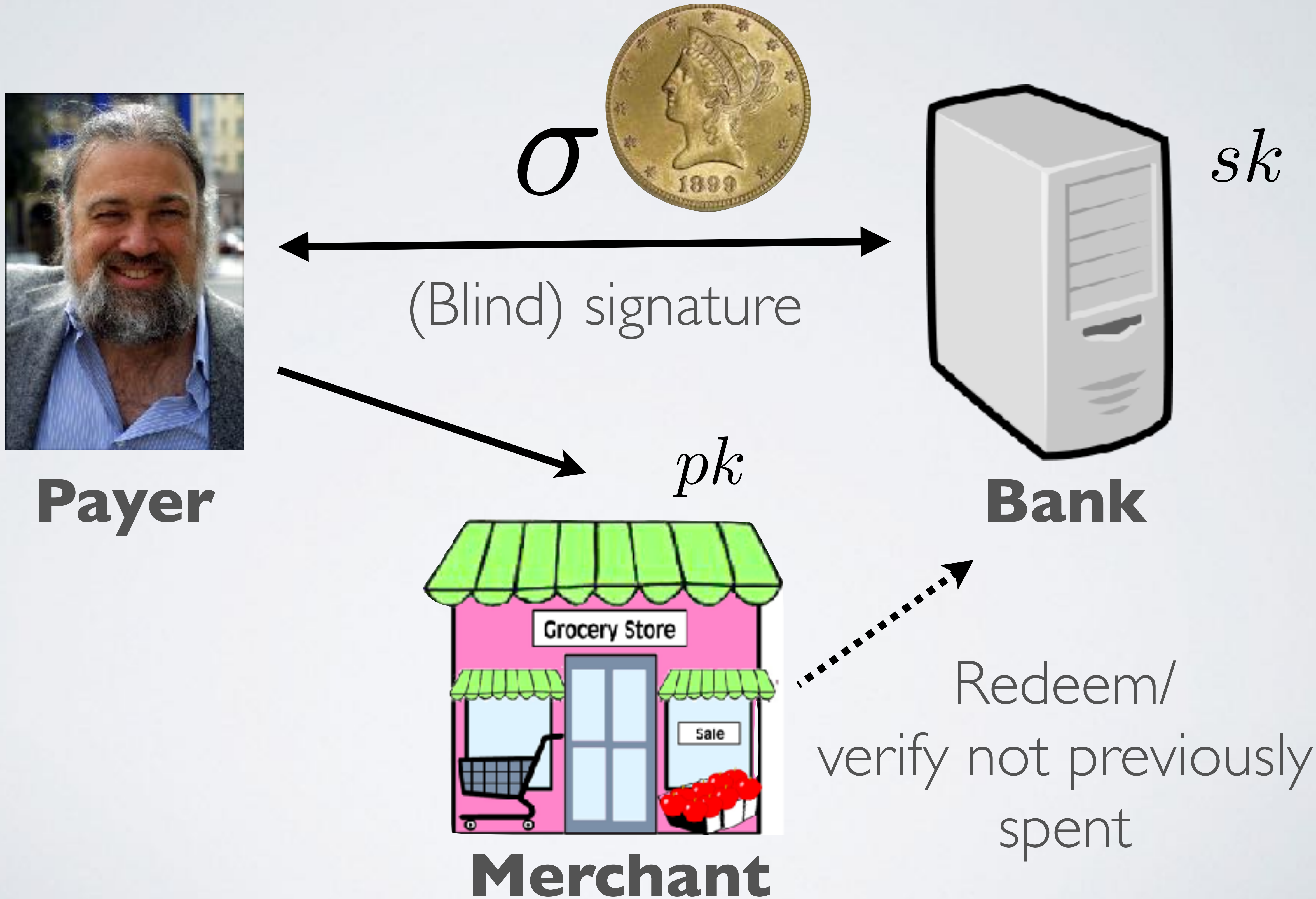
Problems

- **Double spending**
  - To capture double spending you need an online (networked) party that must be trusted
  - They can attack the system or simply fail
- **Privacy**
  - In many naive systems, the bank sees every transaction you make
- **Origin**
  - How is new currency created?

e-Cash

- Devised by Chaum, Chaum/Fiat/Naor, Brands, etc.
  - Move to a “cash” model, with added privacy
  - Individuals would carry redeemable tokens
  - Reduces the problem to detecting double spending and user privacy

Chaum (CRYPTO '83)



CHL (Eurocrypt '05)

$$\sigma^K$$

Payer

$SN = PRF(K, i)$

$NIZK\ \Pi$

For I = 1 to N

(Blind) signature

sk

Bank

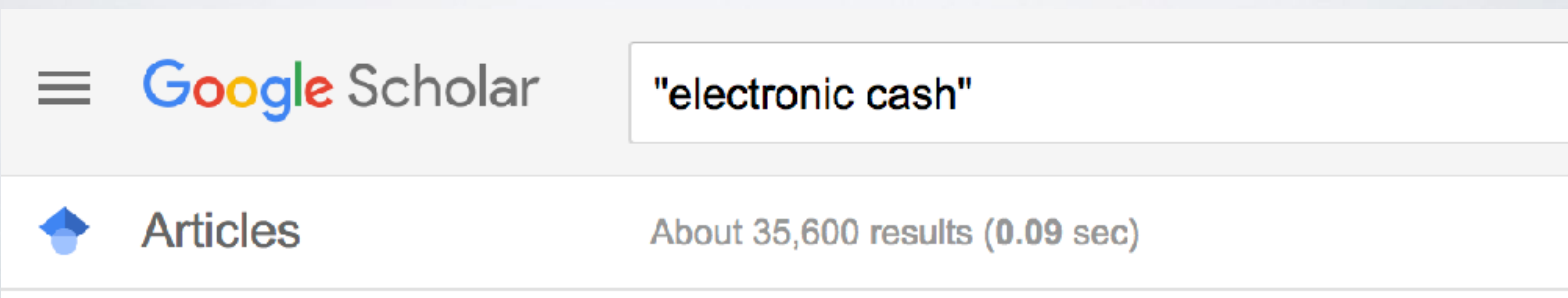
pk

Merchant

Redeem/
verify not previously spent

e-Cash

- Huge number of academic works / practical improvements
  - Online schemes / offline schemes
  - (Offline required using tamper-resistant storage)
  - Main research problem continued to be privacy



Why did centralized e-Cash fail?

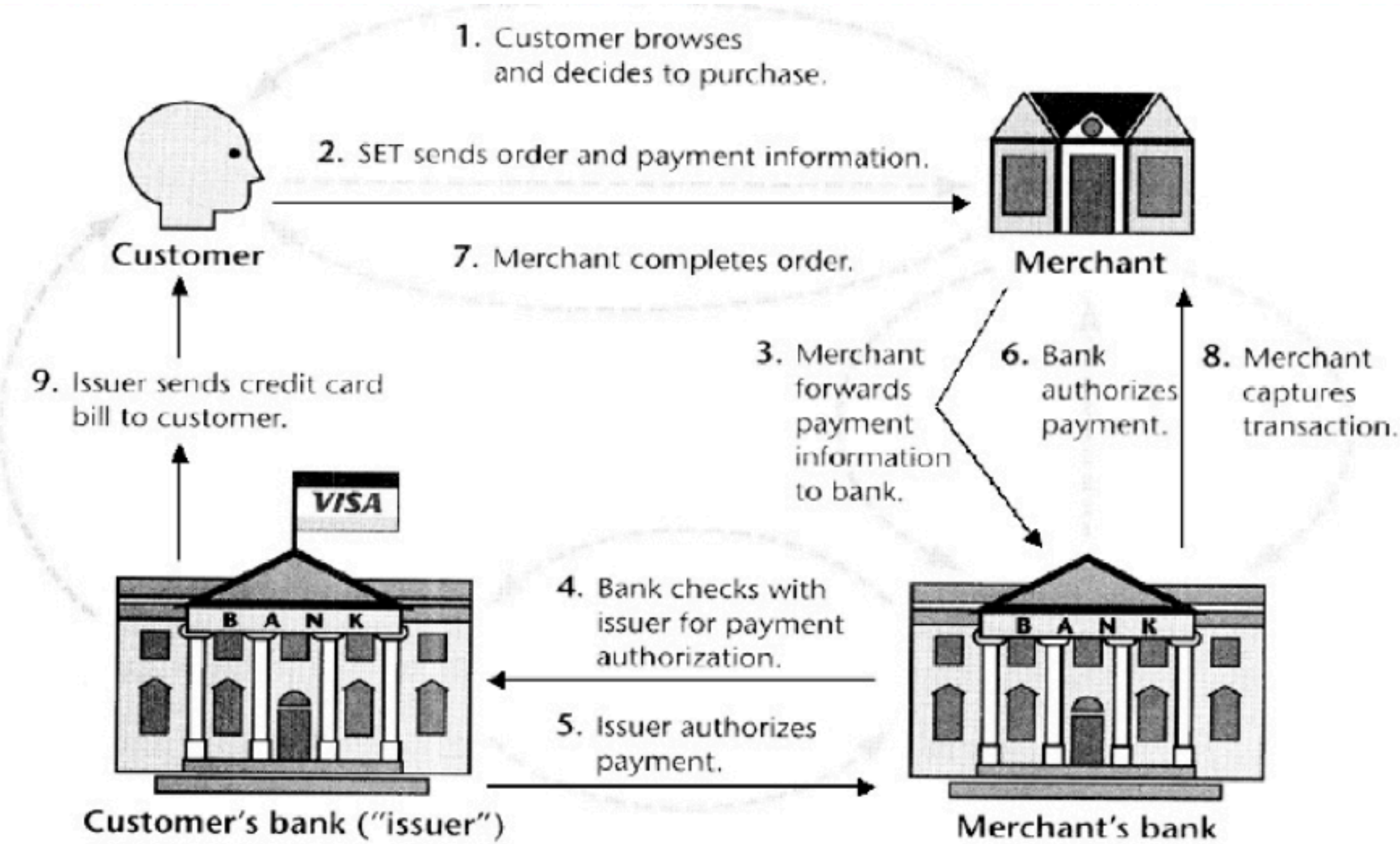
- Deploying e-Cash systems required a centralized bank
  - Required a trusted server with money issuing powers
  - In 1994, EU regulations made this more challenging
  - 9/11 and beyond saw closures of *non-anonymous* currencies (e-Gold and Liberty Reserve)

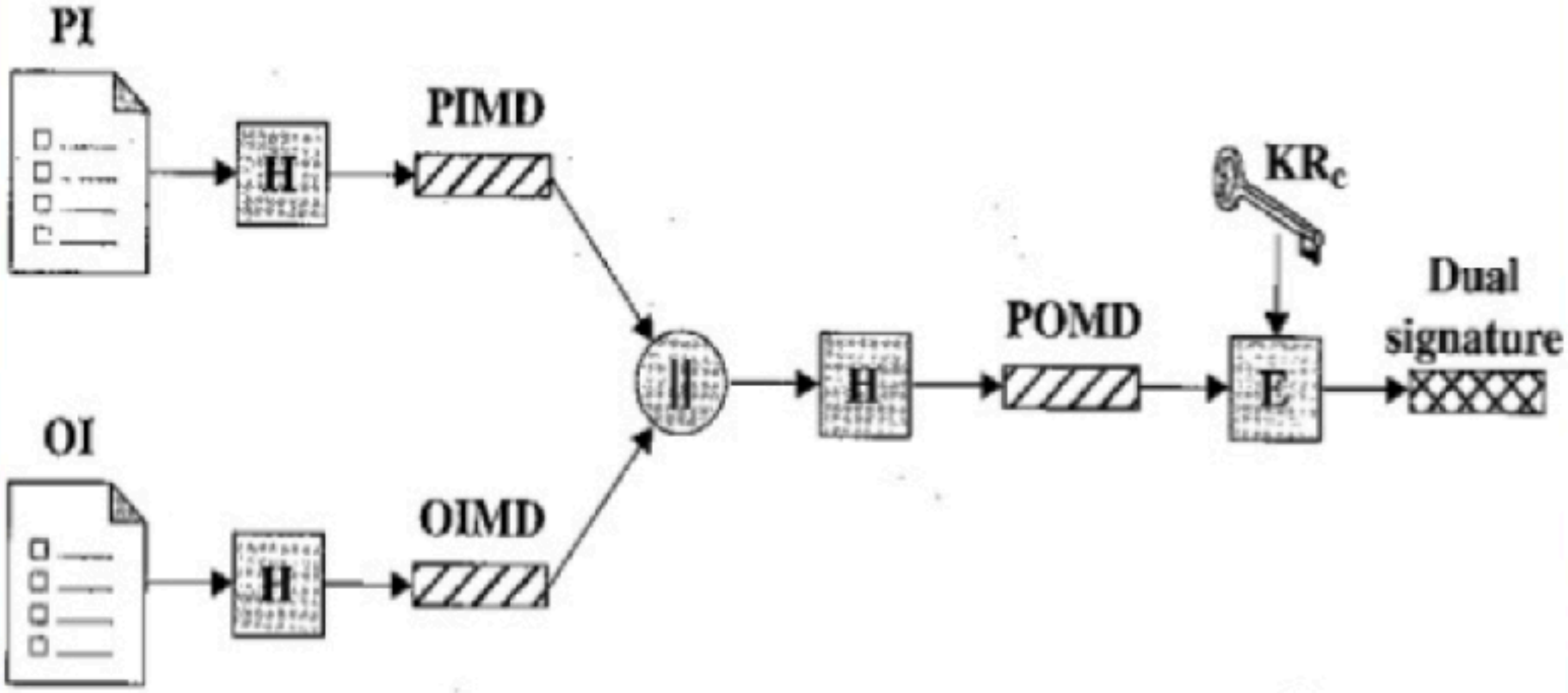
Why did e-Cash fail? (2)

- Were these technical or policy failures? Maybe both.
- The e-Cash model was centralized and relied on a vulnerable interface with the banking system
  - Privacy was (eventually) off the table for regulators
  - Any solution would have to work around those (manufactured) technical problems

1996: SET

- Developed by Visa and MasterCard
  - Cryptographic architecture based on certificates
  - Assurance, authenticity and confidentiality


1. Customer browses and decides to purchase.
2. SET sends order and payment information.
Customer
7. Merchant completes order.
Merchant
3. Merchant forwards payment information to bank.
6. Bank authorizes payment.
8. Merchant captures transaction.
9. Issuer sends credit card bill to customer.
VISA
BANK
4. Bank checks with issuer for payment authorization.
5. Issuer authorizes payment.
BANK
Customer's bank ("issuer")
Merchant's bank

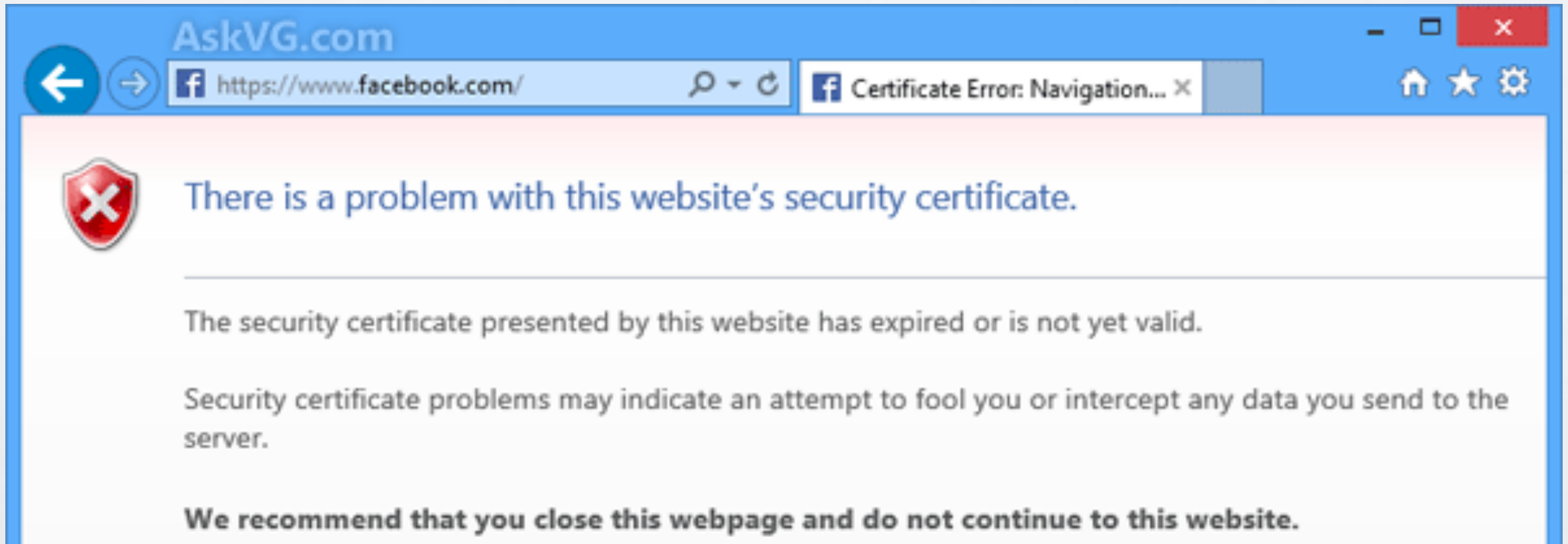


PI = Payment information
OI = Order information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

Why SET failed

- Required end-user certificates
- All the problems of key management PLUS all of the problems of identity verification
- Binding keys to user identities seems to trouble users



Conclusions (1980s-2007)

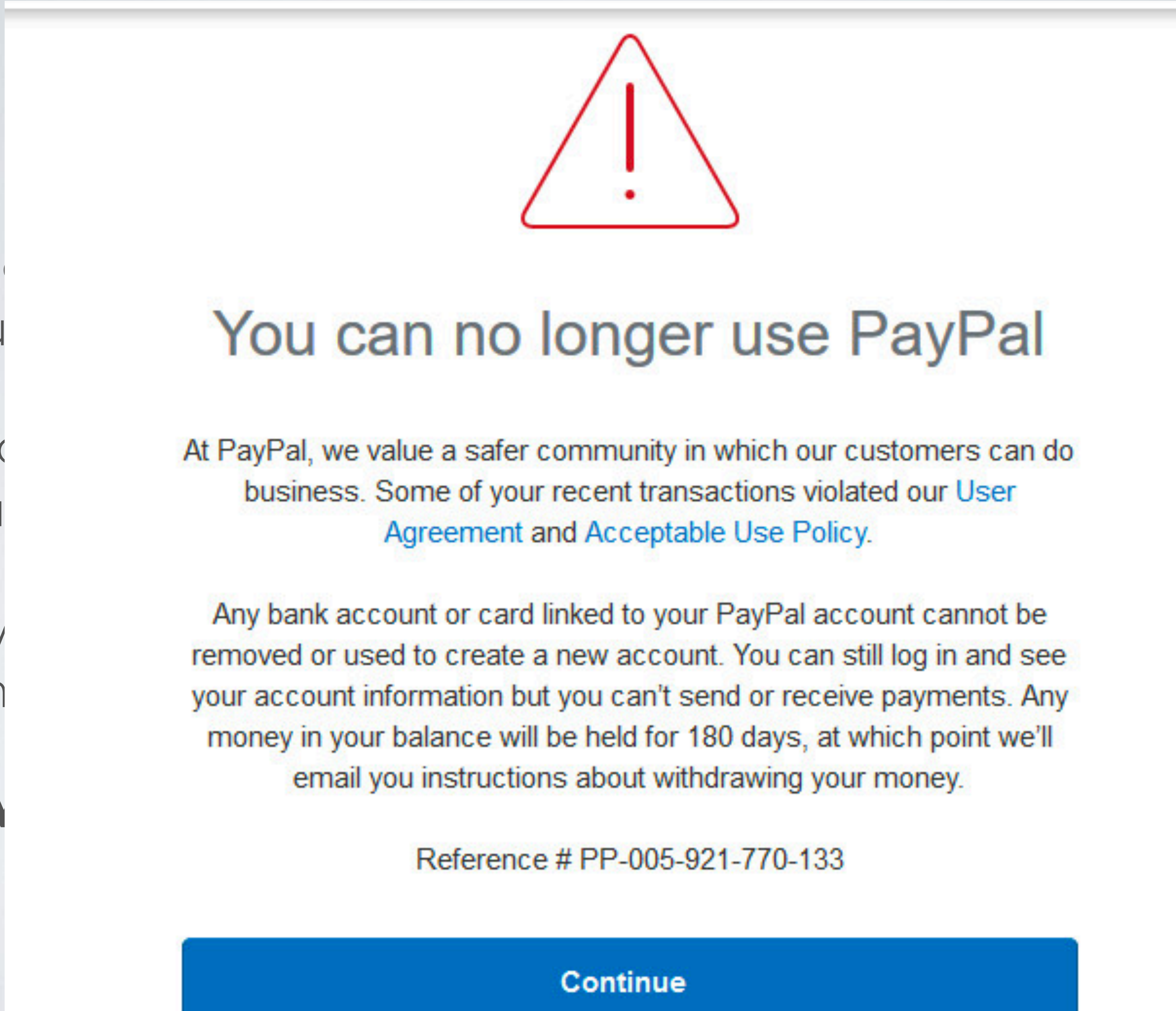
- Most cryptographic solutions too complex, or had "undesirable" features (privacy)

- Commercial solutions (existing credit cards, SET) failed to support the case of person->person transfers

- Web browsers didn't support fancy crypto anyway.

- **We got PayPal**

- M
  “u
- C
  su
- W
  an
- **W**


You can no longer use PayPal
At PayPal, we value a safer community in which our customers can do business. Some of your recent transactions violated our User Agreement and Acceptable Use Policy.
Any bank account or card linked to your PayPal account cannot be removed or used to create a new account. You can still log in and see your account information but you can't send or receive payments. Any money in your balance will be held for 180 days, at which point we'll email you instructions about withdrawing your money.
Reference # PP-005-921-770-133
Continue

Conclusions (1980s-2007)

- Mo ons were too complex, or had unc cy)
- Co sting credit cards, SET) failed to sup n->person transfers
- We port fancy crypto any
- **W**

Conclusions (1980s-2007)

- Mo… on… unc… acy…
- Co… sti… to sup… n-…
- We… po… any…
- **W…**

The decentralized era
2008-2018

Nakamoto, 2008

- Replace the server with a distributed ledger (blockchain)
- Use a new consensus technique to construct the ledger

Nakamoto, 2008

- Replace the server with a distributed ledger (blockchain)
- Use a new consensus technique to construct the ledger
- Use puzzles to handle consensus & generate funds
  [Credit to Dai, (B-Cash) Back (HashCash) etc.]

Nakamoto, 2008

- Replace the server with a distributed ledger (blockchain)
- Use a new consensus technique to construct the ledger
- Use puzzles to handle consensus & generate funds
- Eliminate the need for explicit key/identity bindings

Nakamoto, 2008

- Replace the server with a distributed ledger (blockchain)
- Use a new consensus technique to construct the ledger
- Use puzzles to handle consensus & generate funds
- Eliminate the need for explicit key/identity bindings
- Everything else is straightforward crypto and excellent engineering

Credit for Bitcoin

- With much credit due:
  - Wei Dai, B-cash laid out many ideas
  - Adam Back, HashCash
  - Ledgers used in e-Cash (Sander and Ta-Shma)
  - Years of existing consensus systems (mostly ignored)

Lessons of Bitcoin

- Getting the consensus algorithm right makes all the difference

Lessons of Bitcoin

[B]lockchain-style consensus indeed achieves certain robustness properties in the presence of sporadic participation and node churn that none of the classical style protocols can attain.

- Pass, Shi 2018 (also ‘16, ’17, Daian, Pass, Shi ’16)

Lessons of Bitcoin

- Using the right consensus algorithm really makes a difference
- **Eliminating the need for key/identity management**
  significantly simplifies the currency problem

Lessons of Bitcoin

- Using the right consensus algorithm really makes a difference
- Eliminating the need for key/identity management significantly simplifies the currency problem
- **Human beings are weird**

Lessons of Bitcoin

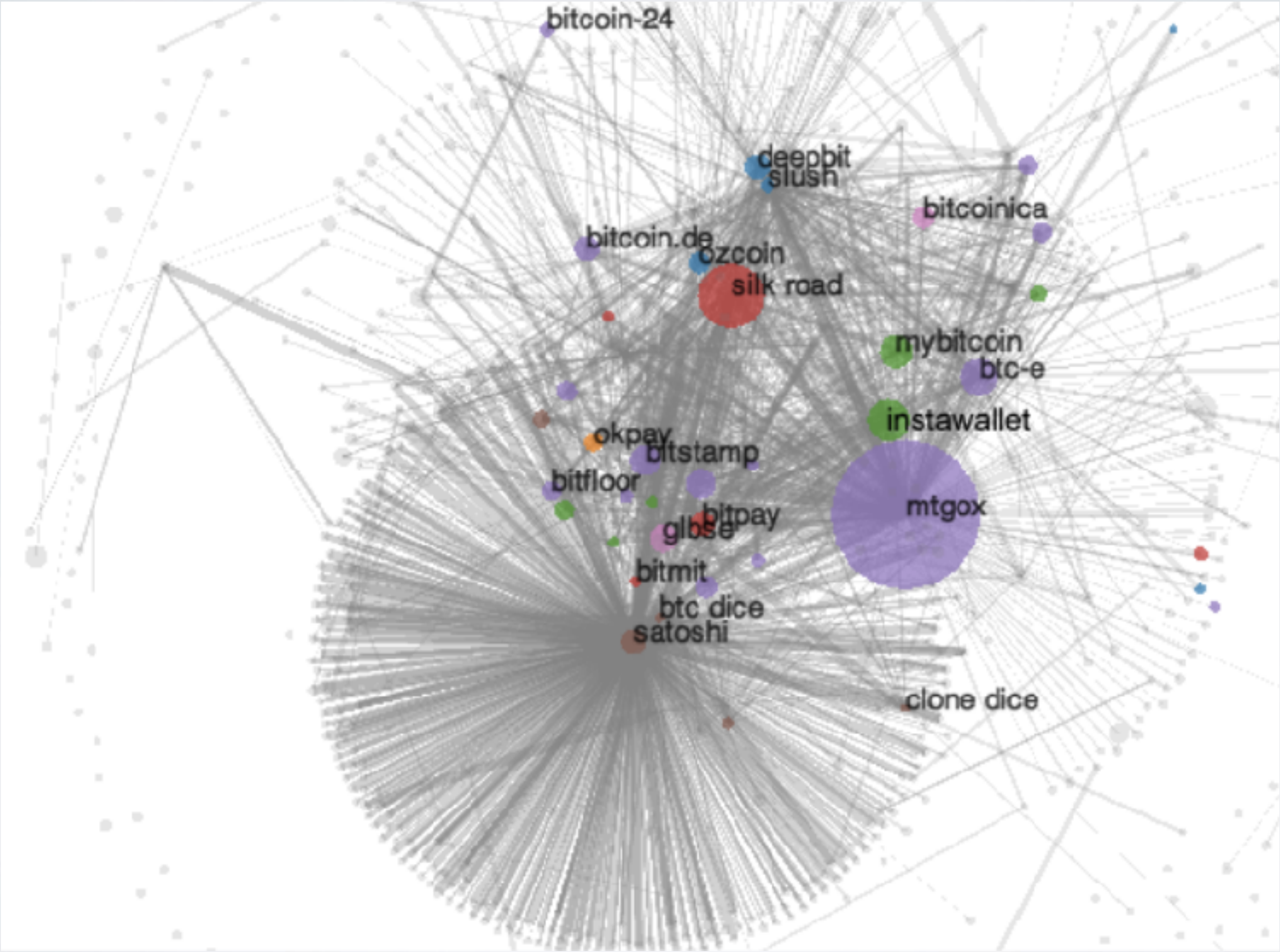
This is simultaneously trivial and the most unexpected lesson of the entire cryptocurrency experiment:

People will assign significant value to **meaningless electronic tokens** — *if* you convince them that the tokens are **secure** and have a **predictable supply.**

Limitations of Bitcoin

- Privacy limitations
- Functionality limitations
- Scalability & Sustainability limitations

Bitcoin & Privacy



Source: MPJLMVS13

🐵🚀

(TS//NF) Met with SSG11 and S2F on the MR access. The following topics were discussed:

- Checking to see if the DTG/Port/IP Address could be assessed to validate if it hits against the BITCOIN Targets
- Checking to see if the partner does any user validation
- The relationship between BITCOIN targets and the MONKEYROCKET data
- Additional data that is not found in XKS-central, but can be made available to the customer
  - The following files were sent to the customer for analysis:
    - Mac_address.csv
    - Password_hash_history.csv
    - Provider user full.csv
    - User_sessions full.csv

As of right now, MONKEYROCKET is offering a sole source for SIGDEV for the BITCOIN Targets. We requested feedback and any mission highlights they have or will have. (SNM)

Zerocoin/Zcash

WARNING

THIS IS DEVELOPMENT SOFTWARE. WE DON'T CERTIFY IT FOR PRODUCTION USE. WE ARE RELEASING THIS DEV VERSION FOR THE COMMUNITY TO EXAMINE, TEST AND (PROBABLY) BREAK. IF YOU SEE SOMETHING, SAY SOMETHING! IN THE COMING WEEKS WE WILL LIKELY MAKE CHANGES TO THE WIRE PROTOCOL THAT COULD BREAK CLIENT COMPATIBILITY. SEE HOW TO CONTRIBUTE FOR A LIST OF WAYS YOU CAN HELP US.

WARNING WARNING

NO, SERIOUSLY. THE ABOVE WARNING IS NOT JUST BOILERPLATE. THIS REALLY IS DEVELOPMENT CODE AND WE'RE STILL ACTIVELY LOOKING FOR THE THINGS WE'VE INEVITABLY DONE WRONG. PLEASE DON'T BE SURPRISED IF YOU FIND OUT WE MISSED SOMETHING FUNDAMENTAL. WE WILL BE TESTING AND IMPROVING IT OVER THE COMING WEEKS.

WARNING WARNING WARNING

WE'RE NOT JOKING. DON'T MAKE US PULL AN ADAM LANGLEY AND TAKE AWAY THE MAKEFILE.

From payments to state

- Of course once you have a ledger…
  - Each Bitcoin transaction can be considered a function f() consuming some previous state and producing a state update
  - Obviously this generalizes nicely to more complex programs and stored data

The future: 2018-

What interests me

- Scaling (channels)
- Replacing PoW
- Conditioning (trustworthy) computation on ledgers

Scaling

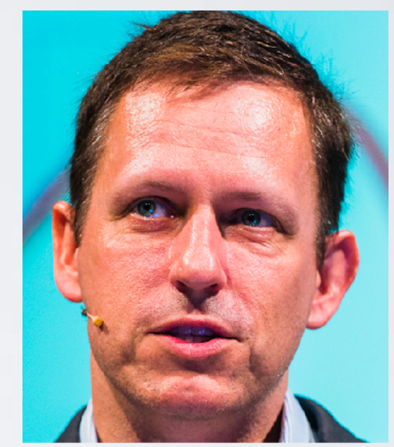
- Current Bitcoin/Ethereum transaction rate is ~7TX/s
- Compare with Visa at 10,000-40,000+ TX.s globally
- This gets worse as transaction complexity increases
- Problems are storage/throughput/validation bandwidth

L2 (Channels)

1

Open

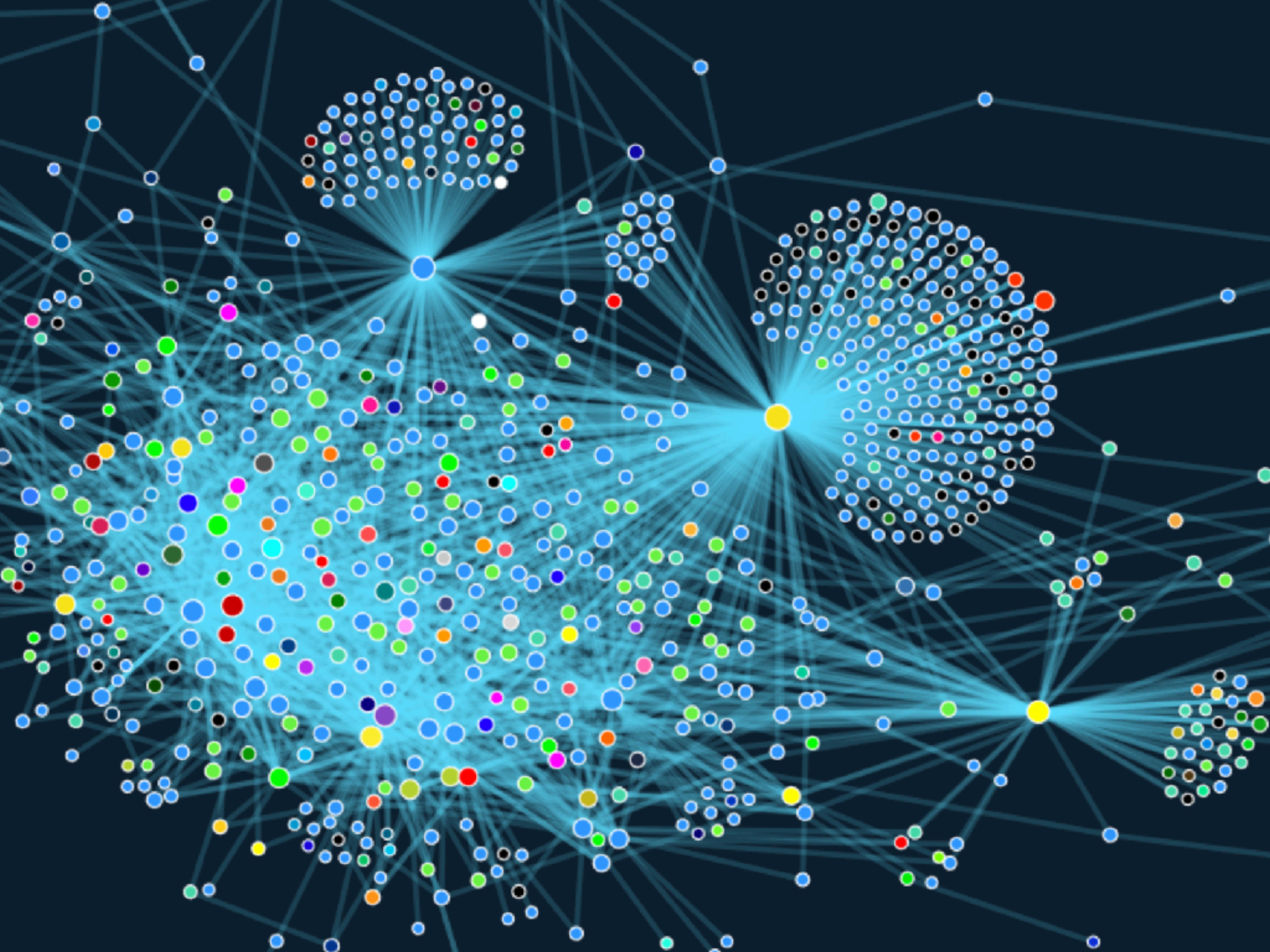
0

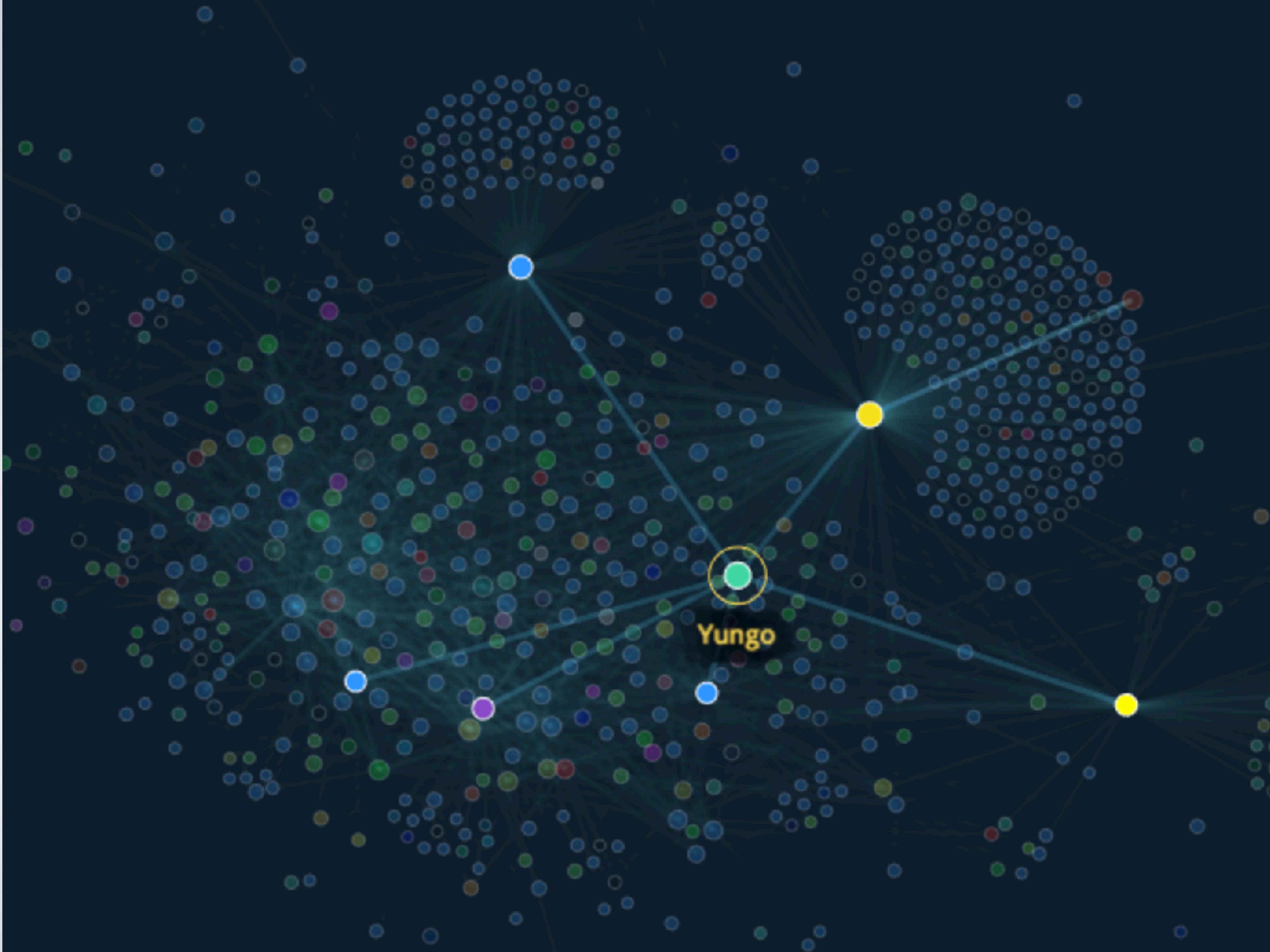


0.9 Update 0.1

0.8 Update 0.2

… Close result on blockchain …

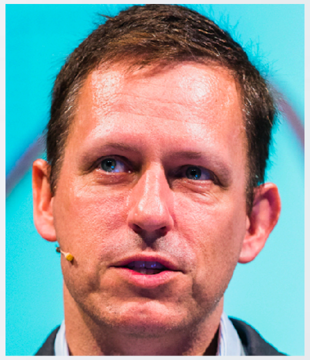



Yungo

Bitcoin / Lightning Network Privacy

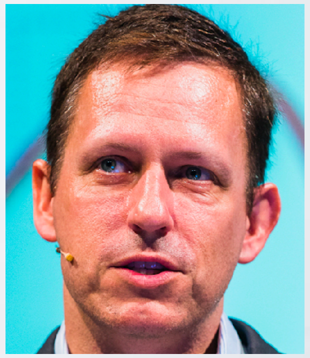
- No real privacy between peers on a single payment channel
- Only way to achieve privacy is to use longer paths
- Requires a complex "Onion Routing" style protocol

$$\rightarrow I_1 \rightarrow I_2 \rightarrow I_3 \rightarrow$$



Channel problems: privacy

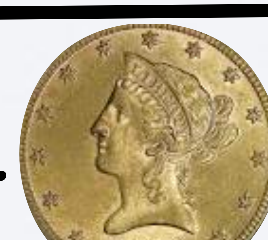
- However, this arrangement doesn't really work well. Aside from cost, it falls to even limited collusion
- Reason: transactions in each channel must share a structure called a "hash lock" that is common between all peers

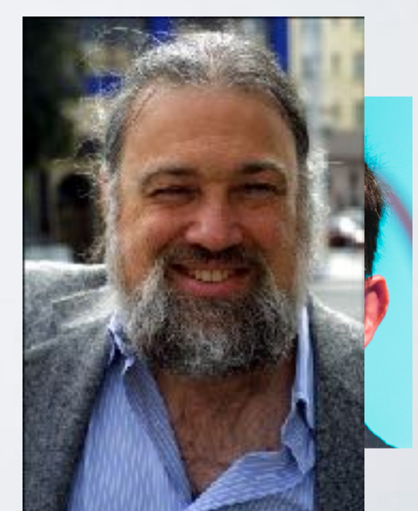
 $\xrightarrow{H} I_1 \xrightarrow{H} I_2 \xrightarrow{H} I_3 \xrightarrow{H}$ 

Channel problems: privacy

- In principle this can be fixed using Chaumian e-cash ideas
- Treat one endpoint of the channel as a Chaumian bank, withdraw coins and spend them back.
- Use channel to ensure fair exchange
- E.g., TumbleBit (Heilman *et al*, 2016), Bolt (Miers, Green, 2016)

σ





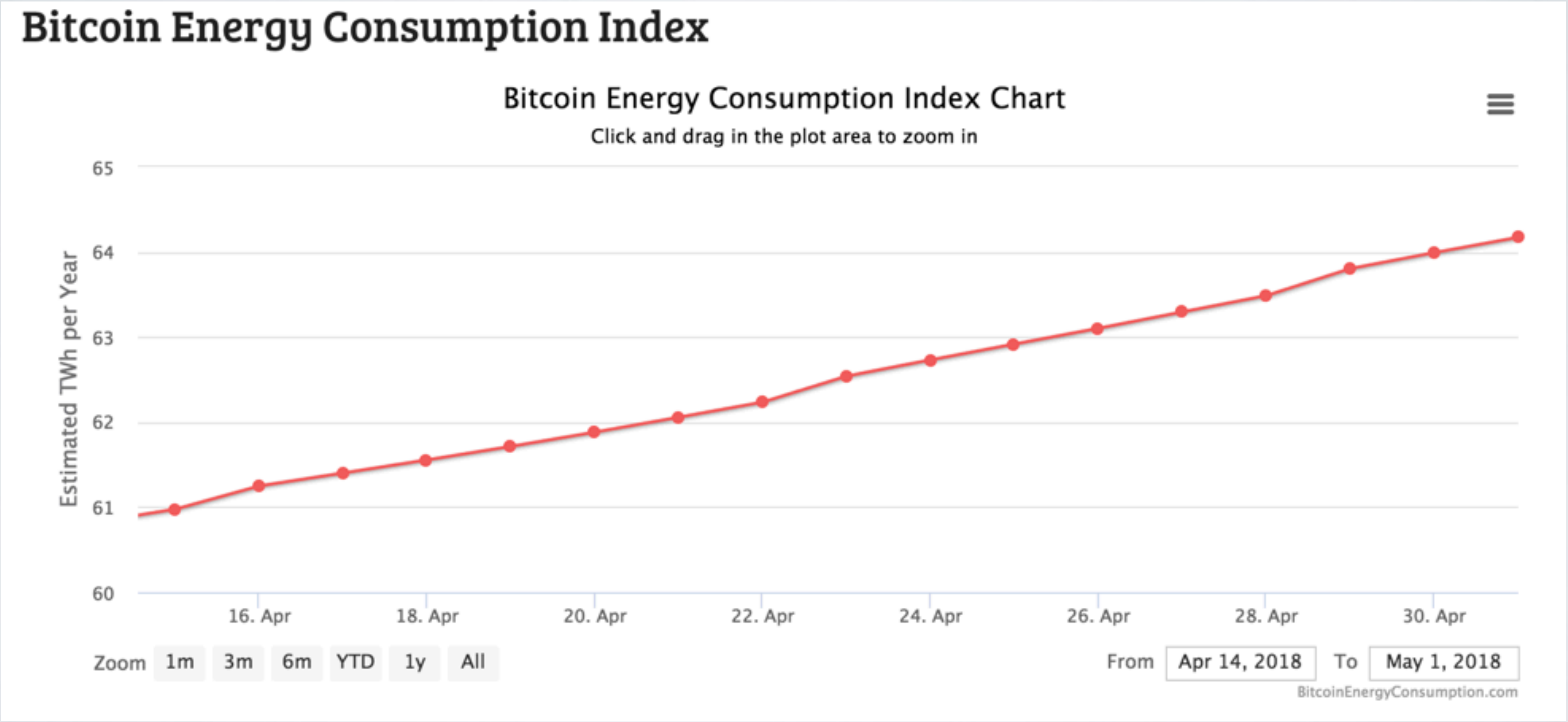
Channel problems: privacy

- This works fairly well for channels of length 1
- Can be made to work for channels of length 2

Channel problems: privacy

- This works fairly well for channels of length 1
- Can be made to work for channels of length 2
- But this model fails to scale to longer paths (2+ hops)
- Fundamentally this is because the disparate channels (with different participants) have to be tied together in some recognizable way
- **Open Problem:** build networks with many-hop paths, without losing (value, payer ID) privacy

Replacing PoW



Proof of Stake

- Current PoW design is obviously unsustainable
- Most common solution (in permissionless) chains is Proof of Stake"
- Rough summary: enumerate all stakeholders of the coin, scaled by their stake — and then sample one to construct the next block

Proof of Stake

- Some excellent work on this happening (here at Eurocrypt!)
- E.g., [DGKR18], [KRDO17]
- Some is currently deployed (Cardano), Ethereum Casper on Testnet
- All current systems require randomness to sample
  [KRDO17] proposed an interactive VSS scheme!
  [DGKR18] uses a grinding-resistant hash function
  (based on CDH)
- This seems to require experimental validation

Ledger-conditioned computation

- Most of the solutions discussed so far use **cryptography** to secure **ledgers** (blockchains)
- Why not use ledgers to secure cryptography?

Ledger-conditioned computation (Setting 1)

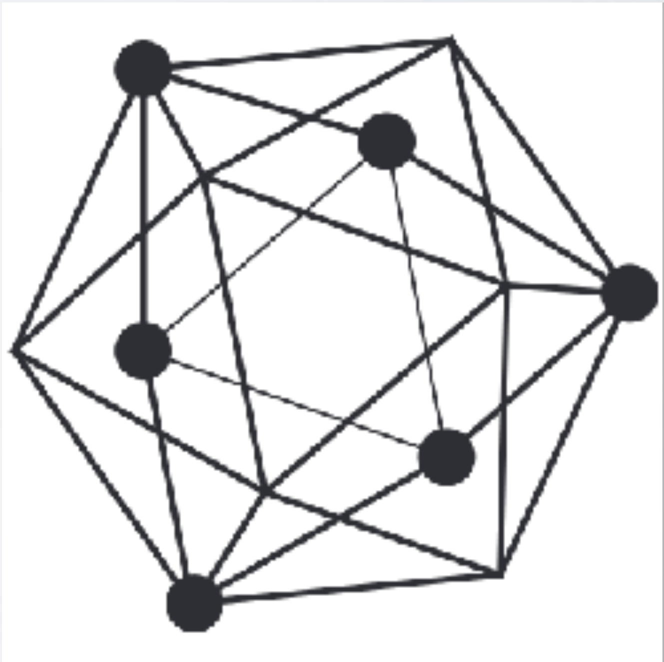
- Assume a trustworthy computing device with internal secrets — but no ability to keep state
- These devices can be constructed inexpensively from hardware, or "virtually" from cryptographic obfuscation and/or MPC
- Assume we want multi-step interactive computation

Ledger-conditioned computation (Setting 2)

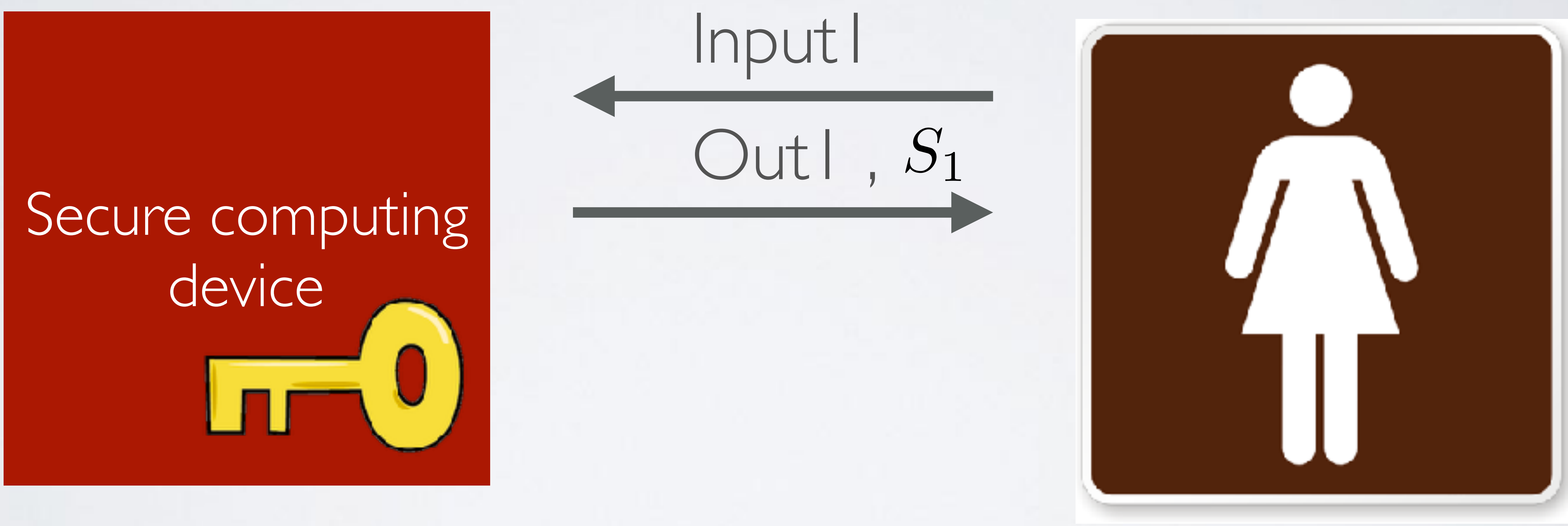
- Alternatively, imagine a network of identical trustworthy computing devices, each provisioned with secrets

- We want to run a single multi-step interactive computation where the node performing the computation can be replaced
  between steps

- "Private smart contracts"
  "AWS Lambda"

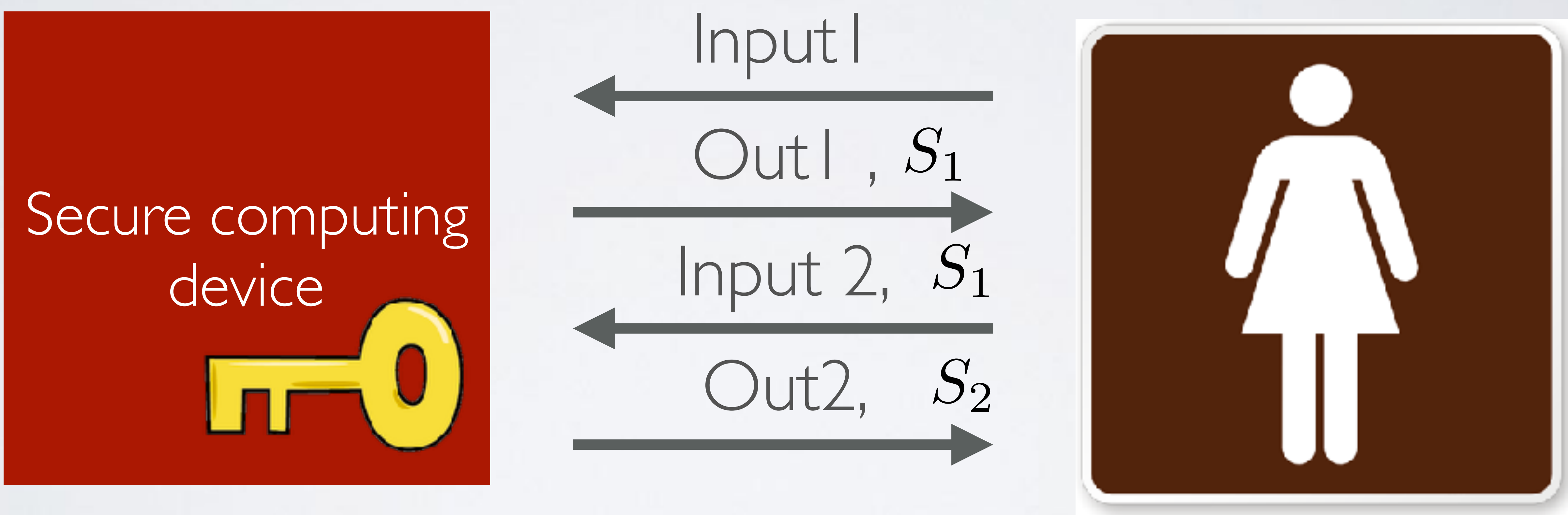


State without ledgers



$$S_1 \leftarrow \mathsf{Encrypt}(K, \mathsf{state}_1)$$

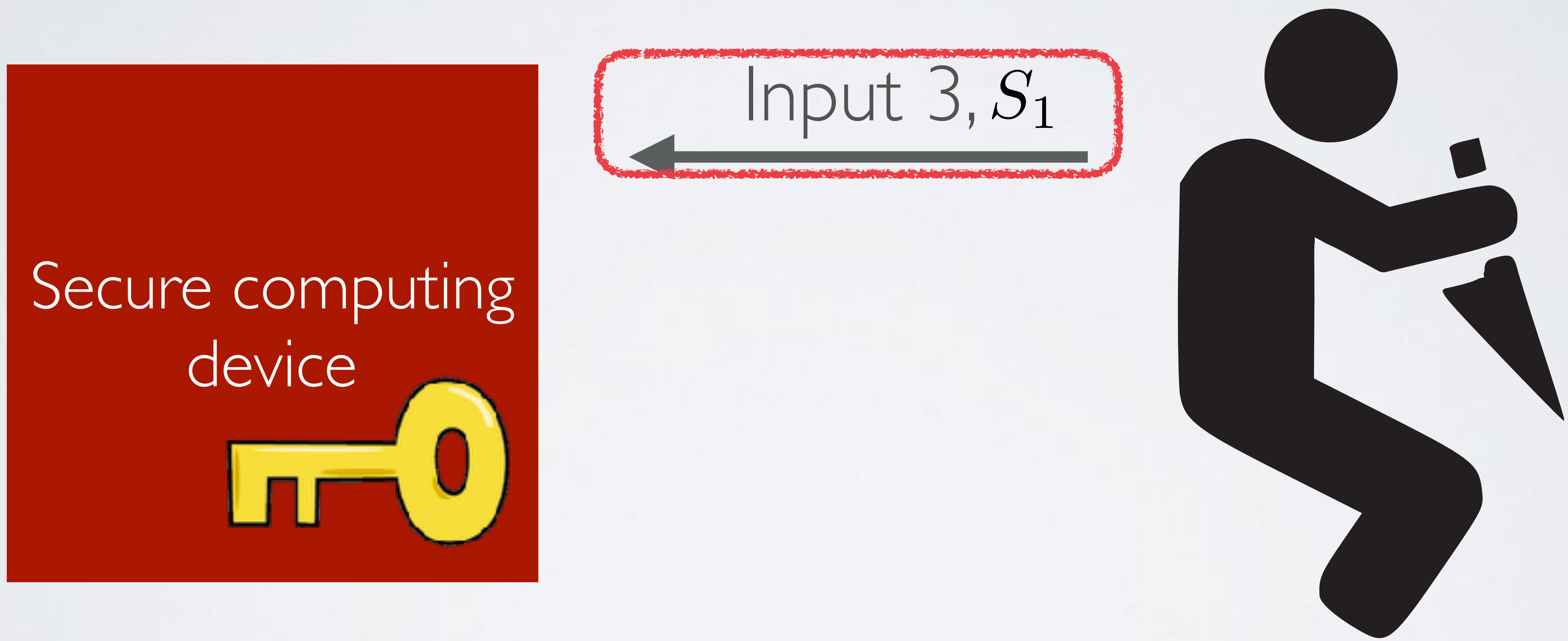
State without ledgers



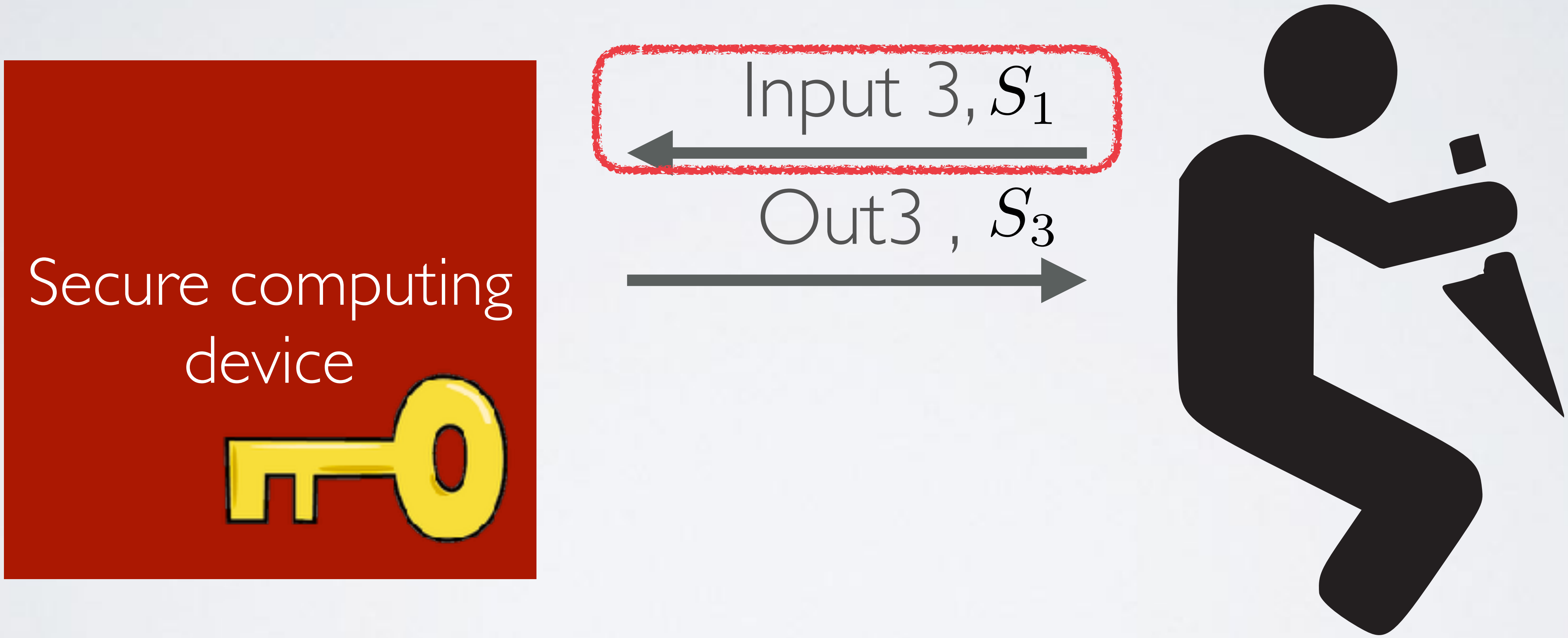
$$\text{state}_2 \leftarrow \text{Decrypt}(K, S_1)$$

$$S_2 \leftarrow \text{Encrypt}(K, \text{state}_2)$$

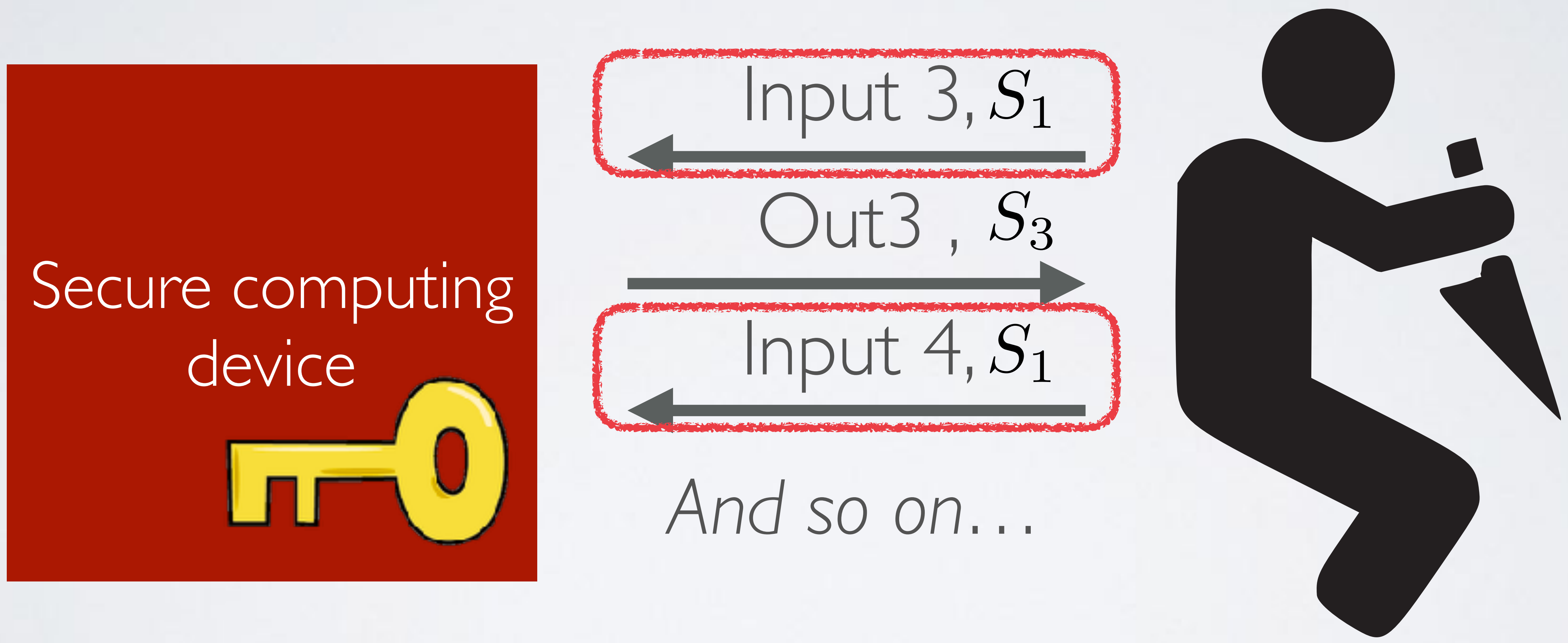
Reset attacks



Reset attacks



Reset attacks



Imagine we have a “publicly verifiable” blockchain:

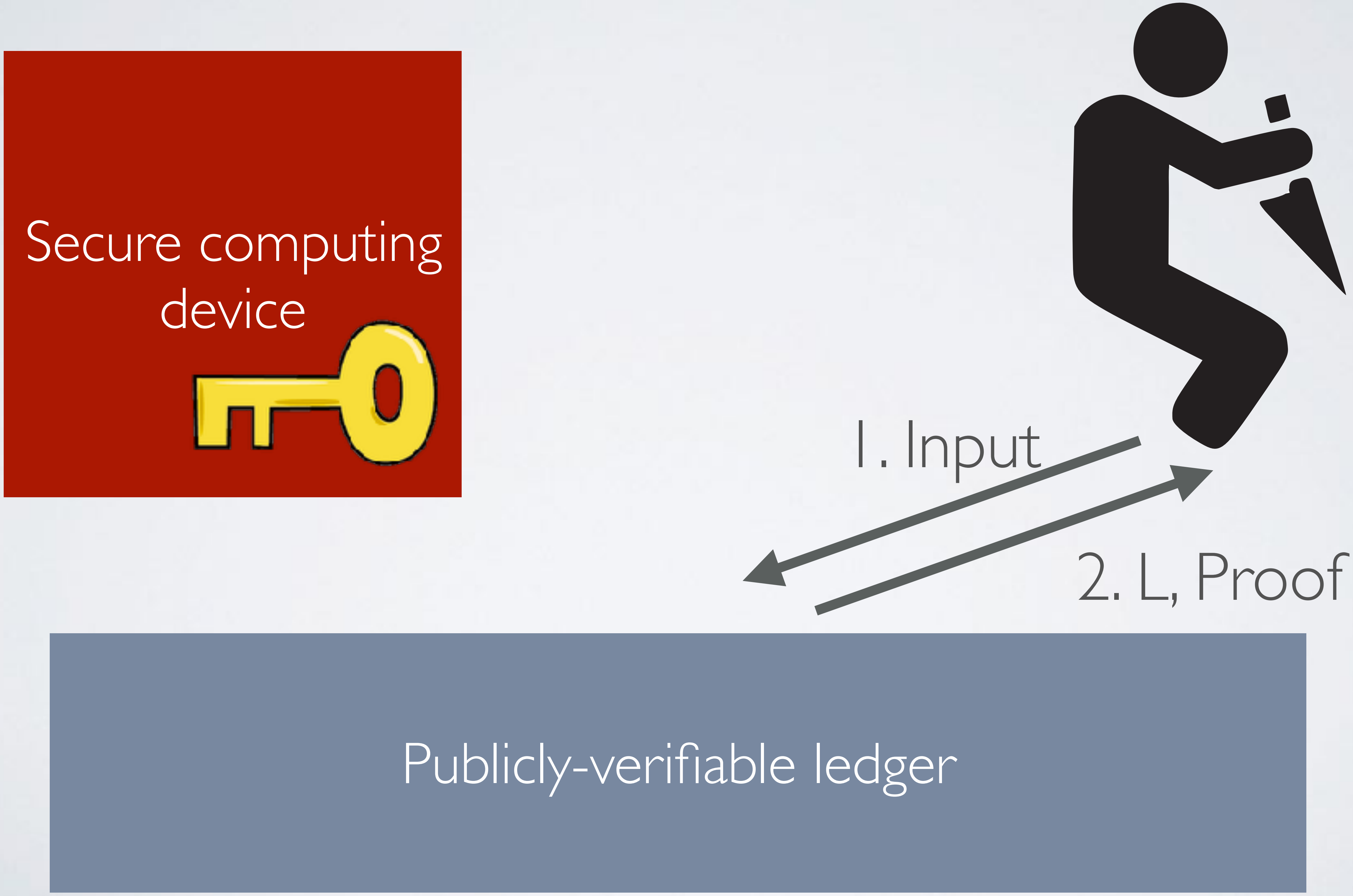
1. We can post a string S
2. Obtain a copy of the <u>full</u> Ledger, plus a proof that the ledger is valid

(This covers most private blockchains, many public blockchains if we make an economic assumption)

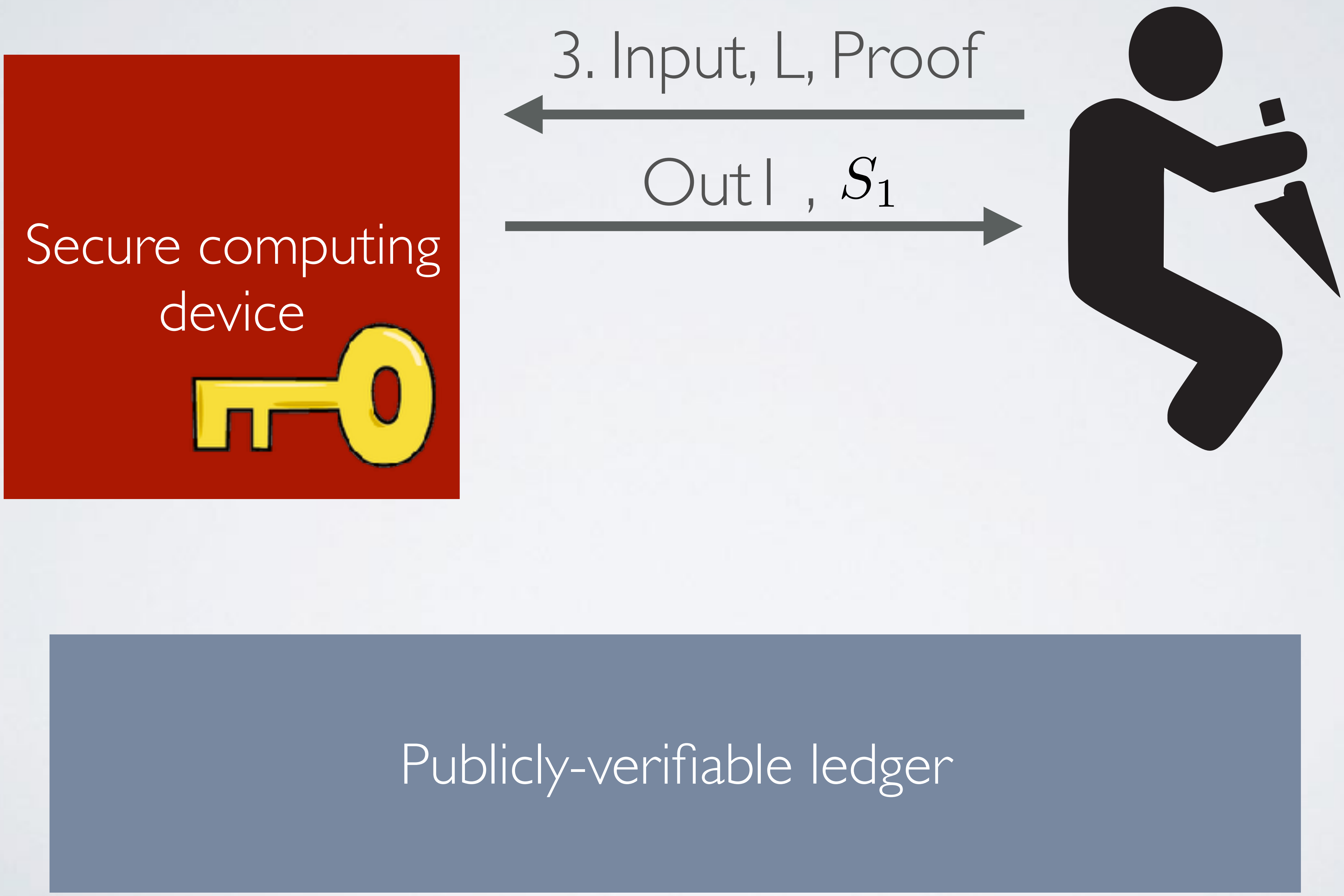
Secu

Publicly-verifiable ledger

Securing state with ledgers



Securing state with ledgers



The ugly

Tony Arcieri
@bascule

Follow

BitGrail lost $170 million worth of Nano XRB tokens because... the checks for whether you had a sufficient balance to withdraw were only implemented as client-side JavaScript reddit.com/r/CryptoCurren …

Anonymous (ID: [illegible]) 02/10/18(Sat)22:27:30 No.7535244

There was a bug, on the withdraw page.

But this check was only on java-script client side, you find the js which is sending the request, then you inspect element - console, and run the java-script manually, to send a request for withdrawal of a higher amount than in your balance.

Bitgrail delivered this withdrawal

How many people did this? Who knows. This bug was later closed.

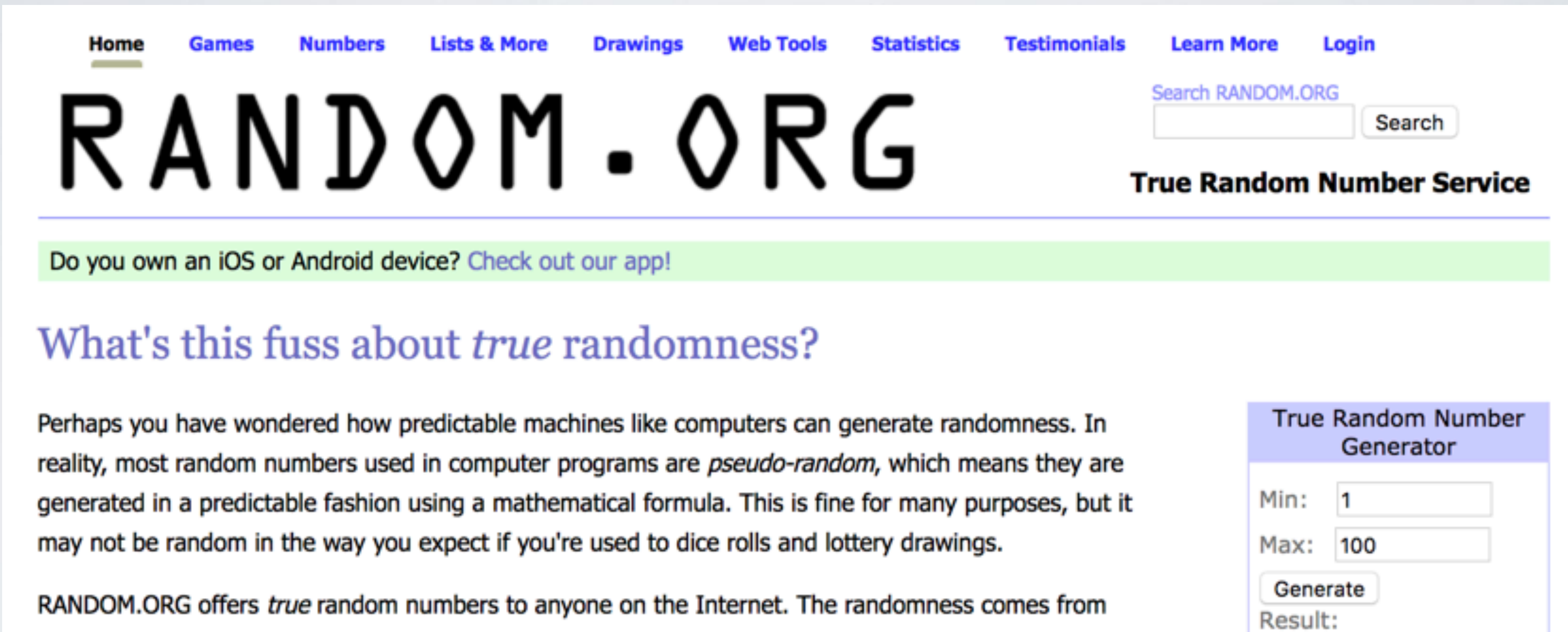
There was another bug, you could request a withdrawal to your address - from another userid, from another user-account. That would cause the other users balance to have "missing funds" or "negative balance".
Bitgrail bomber solved this bug by manually entering the "correct" numbers in his database

This is what you get for using a PHP website coded by same skill-level as CfB of IDIOTA.

9:31 AM - 11 Feb 2018

3,000 Retweets **5,025** Likes

Routine entropy failures



Thanks @ben_h

Routine entropy failures

And the final mistake: They were using HTTP instead of HTTPS to make the webservice call to random.org. On Jan 4, random.org started enforcing HTTPS and returning a 301 Permanently Moved error for HTTP - see https://www.random.org/news/. So since that date, the entropy has actually been the error message (turned into bytes) instead of the expected 256-bit number. Using that seed, SecureRandom will generate the private key for address 1Bn9ReEocMG1WEW1qYjuDrdFzEFFDCq43F 100% of the time. Ouch. This is around the time that address first appears, so the timeline matches.

Thanks @ben_h

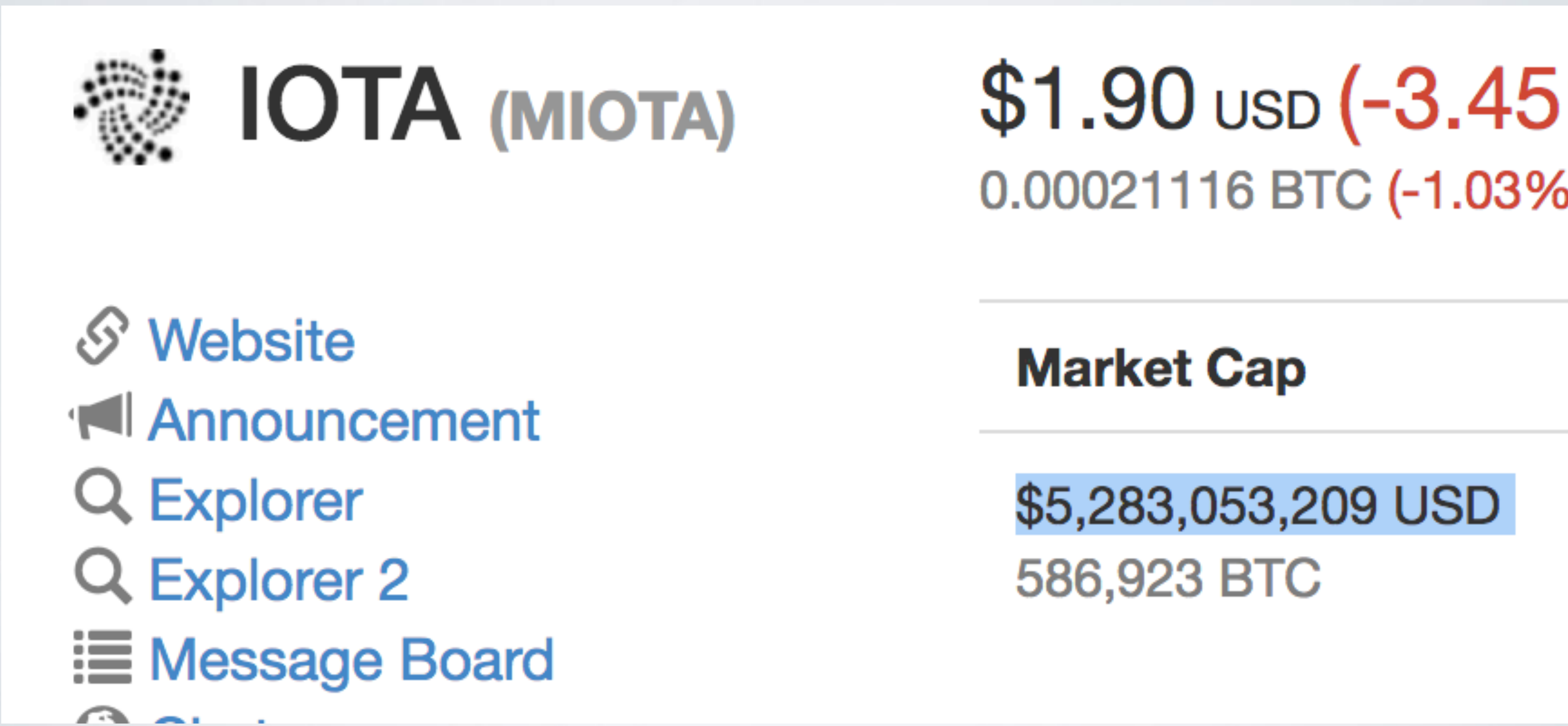
Routine entropy failures

Ethereum Bug Bounty Submission: Predictable ECDSA Nonce

Breaks an ecdsa implementation that uses `privKey xor message` as nonce. Recovering the full private key requires 256 signatures. In other words, every signature leaks 1 bit. A detailed explanation of the attack can be found in the explanation.pdf.

`main.go` is the implementation of an attack specifically against a vulnerable version of github.com/obscuren/secp256k1-go and thus also against go-ethereum . It takes roughly 11 minutes for my 3.0Ghz processor to solve the system. The obvious fix is to use the operating system's PRNG to generate the nonce just like the original project by haltingstate.

Thanks @ben_h


IOTA (MIOTA)
$1.90 USD
0.00021116 BTC (-1.03%)
Website
Announcement
Explorer
Explorer 2
Message Board
Market Cap
$5,283,053,209 USD
586,923 BTC

Neha Narula Follow

Director, Digital Currency Initiative at the MIT Media Lab. I work on scaling applications and platforms for the internet.

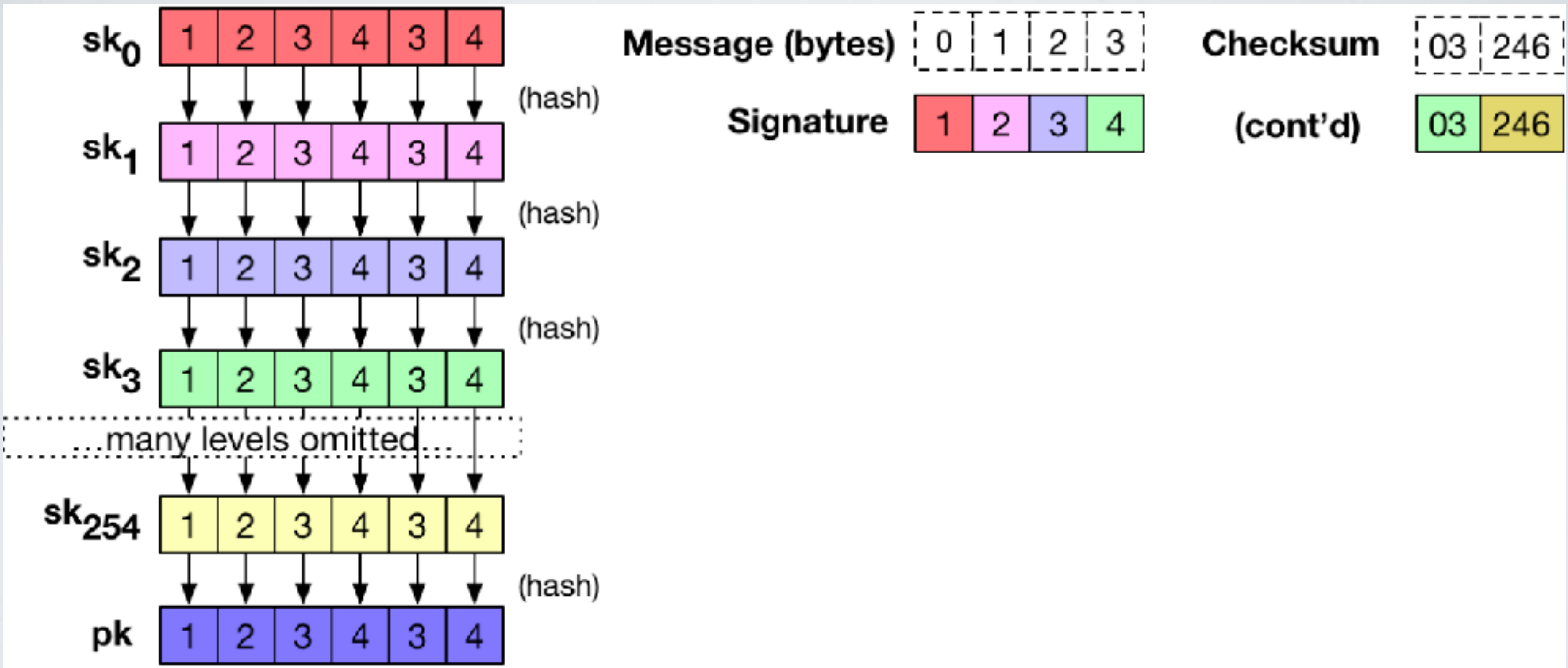
Sep 7, 2017 · 7 min read

Cryptographic vulnerabilities in IOTA

Last month, Ethan Heilman, Tadge Dryja, Madars Virza, and I took a look at IOTA, currently the 8th largest cryptocurrency with a $1.9B market cap. In its repositories on GitHub, we found a serious vulnerability—the IOTA developers had written their own hash function, Curl, and it produced collisions (when different inputs hash to the same output). Once we

Curl-P was created by following the idea of simplicity. While de-jure I can say that it was me who created Curl-P, de-facto it was created by a primitive AI created by me. That wasn't AI of general purpose; an improved version of the AI is working on the final version of Curl now while I'm writing this post. This situation is quite funny because it look unusual, interesting if in the future we'll see cases similar to

IOTA was created to be immune to quantum computer attacks, today I have revealed that it was also created to be immune to attacks from an AI. IOTA was the very first


sk0
1 2 3 4 3 4
(hash)
sk1
1 2 3 4 3 4
(hash)
sk2
1 2 3 4 3 4
(hash)
sk3
1 2 3 4 3 4
…many levels omitted…
sk254
1 2 3 4 3 4
(hash)
pk
1 2 3 4 3 4
Message (bytes)
0 1 2 3
Signature
1 2 3 4
Checksum
03 246
(cont'd)
03 246

Zerocoin

WARNING

THIS IS DEVELOPMENT SOFTWARE. WE DON'T CERTIFY IT FOR PRODUCTION USE. WE ARE RELEASING THIS DEV VERSION FOR THE COMMUNITY TO EXAMINE, TEST AND (PROBABLY) BREAK. IF YOU SEE SOMETHING, SAY SOMETHING! IN THE COMING WEEKS WE WILL LIKELY MAKE CHANGES TO THE WIRE PROTOCOL THAT COULD BREAK CLIENT COMPATIBILITY. SEE HOW TO CONTRIBUTE FOR A LIST OF WAYS YOU CAN HELP US.

WARNING WARNING

NO, SERIOUSLY. THE ABOVE WARNING IS NOT JUST BOILERPLATE. THIS REALLY IS DEVELOPMENT CODE AND WE'RE STILL ACTIVELY LOOKING FOR THE THINGS WE'VE INEVITABLY DONE WRONG. PLEASE DON'T BE SURPRISED IF YOU FIND OUT WE MISSED SOMETHING FUNDAMENTAL. WE WILL BE TESTING AND IMPROVING IT OVER THE COMING WEEKS.

WARNING WARNING WARNING

WE'RE NOT JOKING. DON'T MAKE US PULL AN ADAM LANGLEY AND TAKE AWAY THE MAKEFILE.

Zerocoin (not Zcash)

Emin Gün Sirer
@el33th4xor

Zerocoin gets hacked, hacker creates 370,000 coins out of thin air: zcoin.io/language/en/im…

8:34 PM - Feb 17, 2017

95 80 people are talking about this