Applied Cryptography

Zero-Knowledge Proofs in Cryptocurrencies

Mary Maller

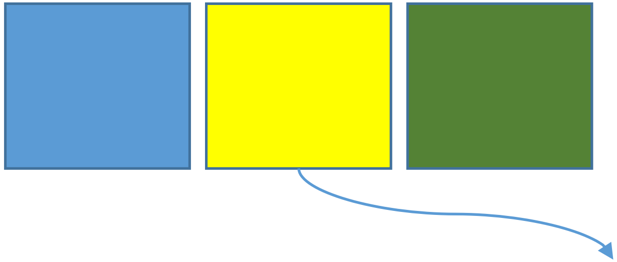
mary.maller.15 at ucl.ac.uk

March, 2017

What is Bitcoin?

What is Bitcoin?

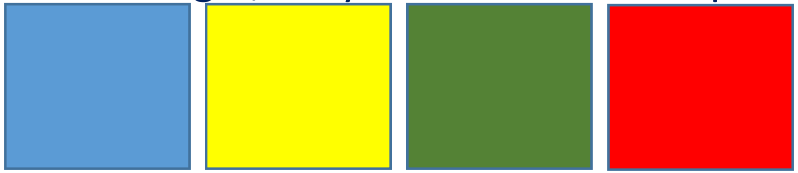
- Bitcoin is a decentralised cryptocurrency that makes use of a globally public, append only ledger that contains a list of every transaction that has ever occurred.



Public ledger maintained by decentralised network of miners.

What is Bitcoin?

- Miners are incentivised to maintain the ledger by a reward of bitcoins, which they receive whenever their block appears on the ledger. To get their block of transactions on the ledger, they must solve a computationally difficult problem.



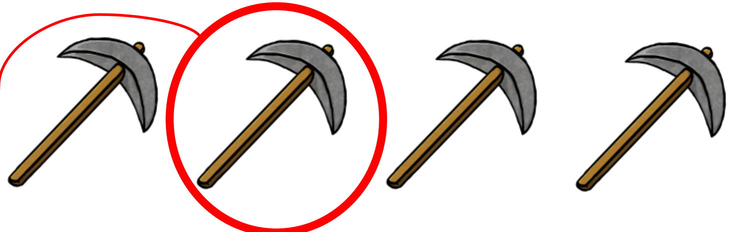
When Miner M solves problem, miner M’s block of valid transactions is added to the ledger, and the new block puts coin C in M’s account.

What is Bitcoin?

- If Miner M wants to spend coin C, they broadcast a transaction saying to transfer coin C to (a hash of) another public key, say pk_A, and they sign the transaction using their secret key.

New block contains transaction saying move coin C from M's account to A's account.

Miner M



What is Bitcoin?

If A now wishes to spend coin C, they broadcast a transaction to move coin C from A's account to another users account, and sign it with their secret key.

What is Bitcoin?

If A's signature does not verify, or if A has previously spent coin C, then the miners will not include A's transaction in the ledger.

Does Pseudonymity Provide Anonymity?

- When receiving funds users can use a new public key to prevent them from being traced.

- When spending funds the users have to use the public key that contains their funds.

Does Pseudonymity Provide Anonymity?

- Each coin can be traced on the ledger from its creation to its current state.
- It is often possible to link public addresses to real world identities of the people who own them.
- See Sarah Meiklejohn's thesis for more details.

A Fistful of Bitcoins: Characterizing Payments Among Men with No Names

Sarah Meiklejohn   Marjori Pomarole   Grant Jordan
Kirill Levchenko   Damon McCoy[†]   Geoffrey M. Voelker   Stefan Savage

University of California, San Diego   George Mason University[†]

ABSTRACT

Bitcoin is a purely online virtual currency, unbacked by either physical commodities or sovereign obligation; instead, it relies on a combination of cryptographic protection and a peer-to-peer protocol for witnessing settlements. Consequently, Bitcoin has the unintuitive property that while the ownership of money is implicitly anonymous, its flow is globally visible. In this paper we explore this unique characteristic further, using heuristic clustering to group Bitcoin wallets based on evidence of shared authority, and then using re-identification attacks (i.e., empirical purchasing of goods and services) to classify the operators of those clusters. From this analysis, we characterize longitudinal changes in the Bitcoin market, the stresses these changes are placing on the system, and the challenges for those seeking to use Bitcoin for criminal or fraudulent purposes at scale.

Categories and Subject Descriptors

K.4.4 [**Electronic Commerce**]: Payment schemes

Keywords

Bitcoin; Measurement; Anonymity

1. INTRODUCTION

Demand for low friction e-commerce of various kinds has driven a proliferation in online payment systems over the last decade. Thus, in addition to established payment card networks (e.g., Visa and Mastercard) a broad range of so-called "alternative payments" has emerged including eWallets (e.g., Paypal, Google Checkout, and WebMoney), direct debit systems (typically via ACH, such as eBillMe), money transfer systems (e.g., Moneygram) and so on. However, virtually all of these systems have the property that they are denominated in existing fiat currencies (e.g., dollars), explicitly identify the payer in transactions, and are centrally or quasi-centrally administered.[1]

[1]In particular, there is a central controlling authority who has the technical and legal capacity to tie a transaction back to a pair of individuals.

IMC'13, October 23–25, 2013, Barcelona, Spain.

http://dx.doi.org/10.1145/2504730.2504747.

By far the most intriguing exception to this rule is Bitcoin. First deployed in 2009, Bitcoin is an independent online monetary system that combines some of the features of cash and existing online payment methods. Like cash, Bitcoin transactions do not explicitly identify the payer or the payee: a transaction is a cryptographically-signed transfer of funds from one public key to another. Moreover, like cash, Bitcoin transactions are irreversible (in particular, there is no *chargeback* risk as with credit cards). However, unlike cash, Bitcoin requires third party mediation: a global peer-to-peer network of participants validates and certifies all transactions; such decentralized accounting requires each network participant to maintain the entire transaction history of the system, currently amounting to over 3GB of compressed data. Bitcoin identities are thus *pseudo-anonymous*: while not explicitly tied to real-world individuals or organizations, all transactions are completely transparent.[2]

This unusual combination of features has given rise to considerable confusion about the nature and consequences of the anonymity that Bitcoin provides. In particular, there is concern that the combination of scalable, irrevocable, anonymous payments would prove highly attractive for criminals engaged in fraud or money laundering. In a widely leaked 2012 Intelligence Assessment, FBI analysts make just this case and conclude that a key "advantage" of Bitcoin for criminals is that "law enforcement faces difficulties detecting suspicious activity, identifying users and obtaining transaction records" [7]. Similarly, in a late 2012 report on Virtual Currency Schemes, the European Central Bank opines that the lack of regulation and due diligence might enable "criminals, terrorists, fraudsters and money laundering" and that "the extent to which any money flows can be traced back to a particular user is unknown" [6]. Indeed, there is at least some anecdotal evidence that this statement is true, with the widely publicized "Silk Road" service using Bitcoin to trade in a range of illegal goods (e.g., restricted drugs and firearms). Finally, adding to this urgency is Bitcoin's considerable growth, both quantitatively — a merchant servicer, Bitpay, announced that it had signed up over 1,000 merchants in 2012 to accept the currency, and in April 2013 the exchange rate soared to 235 USD per bitcoin before settling to a more modest 100 USD per bitcoin — and qualitatively via integration with existing payment mechanisms (e.g., Bitinstant offering to tie users' Bitcoin wallets to Mastercard accounts [5] and Bitcoin Central's recent partnership with the French bank Crédit Mutuel Arkéa to gateway Bitcoin into the banking system [16]) and the increasing attention of world financial institutions (e.g., Canada's recent decision to tax Bitcoin transactions [3] and FinCEN's recent regulations

[2]Note that this statement is not strictly true since private exchanges of Bitcoin between customers of a single third party exchange, such as Mt. Gox, need not (and do not) engage the global Bitcoin protocol and are therefore not transparent.

Anonymous Cryptocurrencies

- In recent years various alternative cryptocurrencies have been springing up that claim to provide spender anonymity.

- Currently the two most prominent are Monero and Zcash.

Anonymous Cryptocurrencies

- In recent years various alternative cryptocurrencies have been springing up that claim to provide spender anonymity.

- Currently the two most prominent are Monero and Zcash.

- Zcash is a fork of bitcoin that provides provable anonymity (although schemes with security proofs do get broken) and is the focus of this lecture.

Anonymous Cryptocurrencies

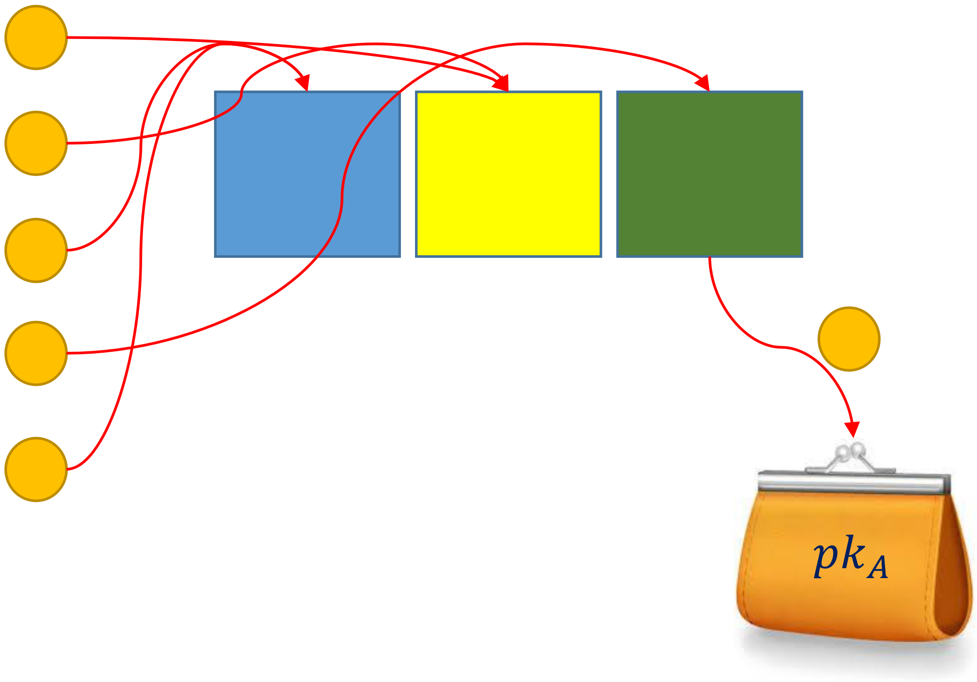
- In recent years various alternative cryptocurrencies have been springing up that claim to provide spender anonymity.

- Currently the two most prominent are Monero and Zcash.

- Monero aims to provide anonymity without using zero-knowledge. See http://www.nicolascourtois.com/bitcoin/paycoin_privacy_monero_6.pdf for more on Monero.

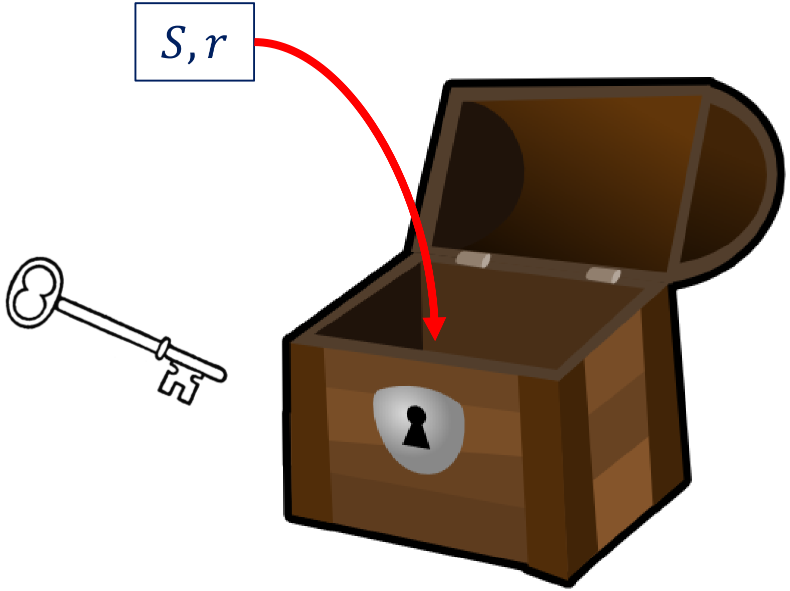
Zcash is a Sophisticated Mixing Service

Many coins from many public keys are placed on the ledger (via an algorithm called $mint$).



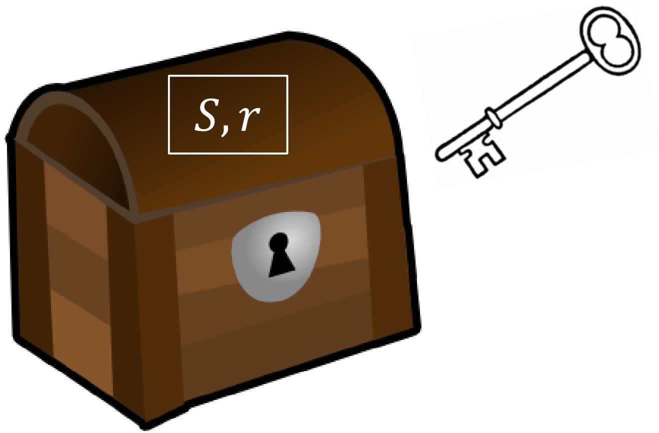
One of the coins that is on the ledger is sent to pk_A, but it is impossible to tell which one

Commitments are used to Shield Coins



Commitments are used to Shield Coins

Nobody can see what is inside commitment.



The owner of the commitment can only open the commitment to one value.

Commitments are used to Shield Coins

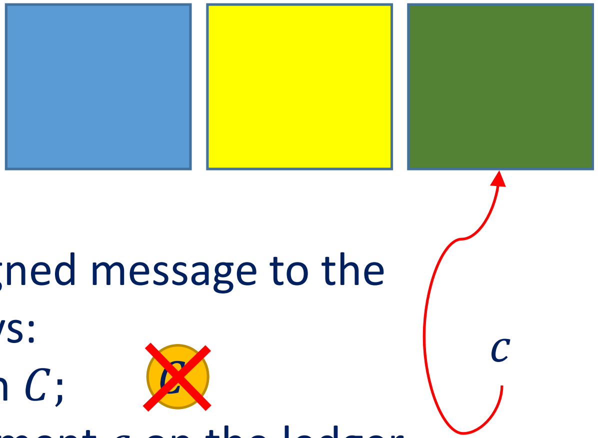
- The commitment scheme takes a serial number S, some randomness r and calculates $c = com(S, r)$.

- Commitment schemes are binding: given S, r, and $c = com(S, r)$, it is hard to find another S', r' such that $c = com(S', r')$.

- Commitment schemes are hiding: given $c = com(S, r)$, it is impossible to know which S, r were used.

Commitments are used to Shield Coins

The owner of the public key pk_A that contains coin C wants to put a coin C on the ledger. C



They choose serial number S, randomness r, and calculate a commitment $c = com_{ck}(S, r)$.

They send a signed message to the miners that says:

- Destroy coin C; $\cancel{C}$
- Put commitment c on the ledger.

Zero-Knowledge Arguments are used to Spend Shielded Coins

Somebody sends a message m to the miners that says to send a coin to pk_A, and it also sends a serial number S, and a proof that one of the commitments on the ledger contains S.

S, π

m, S, π

Coin C to pk_A

If the proof verifies, and the serial number S is not on the ledger, then a coin is created and sent to pk_A and S, π, is put on the ledger.

Zcash Uses Zero-Knowledge Arguments

PROs:

- Proofs reveal no information about which commitment contains the users serial number rendering network analysis hard.

- The proofs themselves will not even reveal identities even against infinitely powerful adversaries i.e. they are perfect zero-knowledge arguments[1].

1. Arguments are computationally sound and proofs are perfectly sound. It is not possible to get perfect zero-knowledge proofs

Zcash Uses Zero-Knowledge Arguments

CONs:

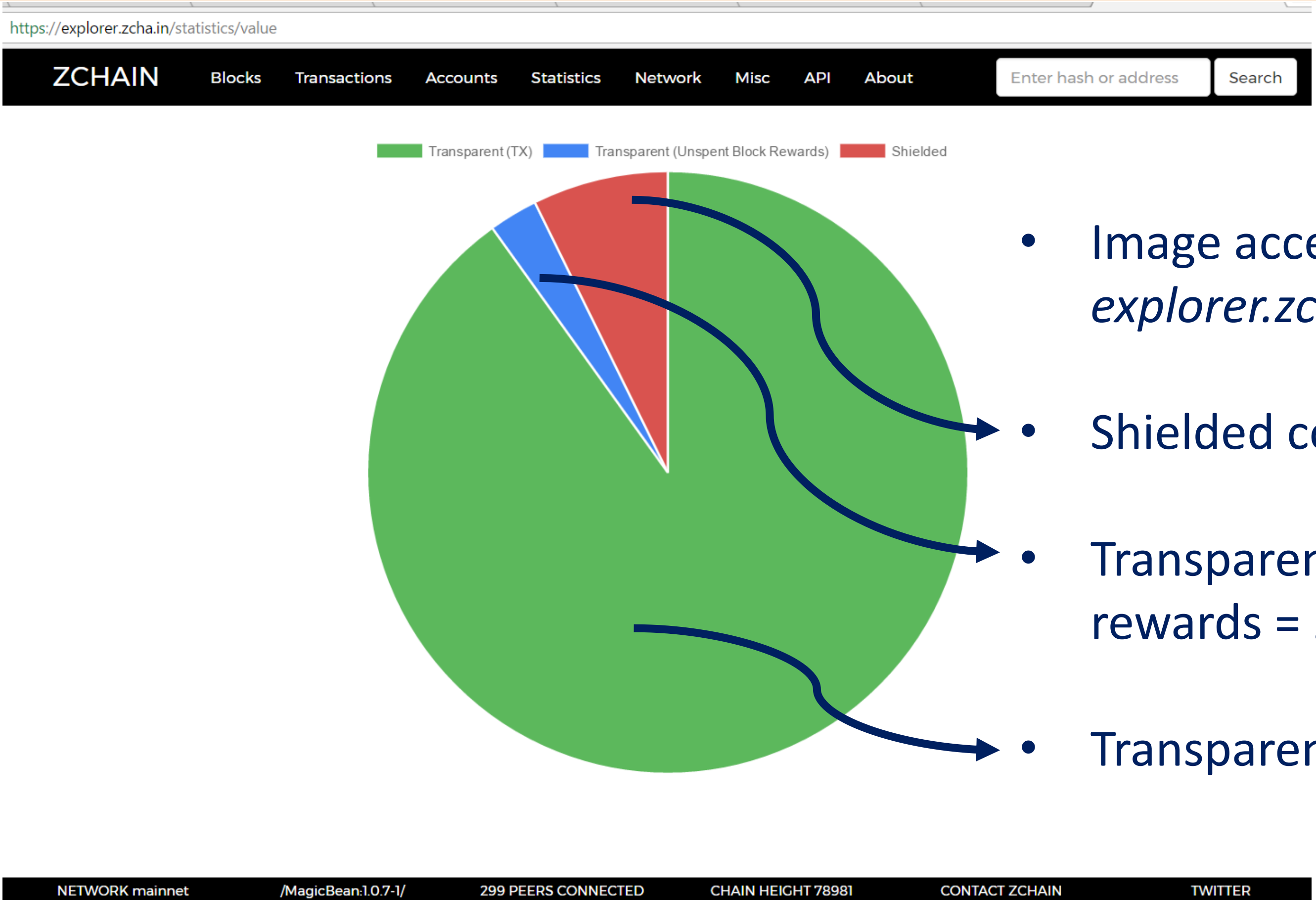
- Zero-Knowledge is expensive. Bitcoin already suffers scalability issues, and Zcash is worse.

- Zero-Knowledge argument systems have a trapdoor by definition. Lots more on this to come.

Dealing with Scalability Issues

- Zcash runs off the bitcoin protocol and users do not have to use shielded coins.

- Many Zcash users are not currently using the Zcash extension at all.



- Image accessed 14/03/17 from *explorer.zcha.in/statistics/value*
- Shielded coins = 7.3%
- Transparent unspent block rewards = 2.6%
- Transparent coins = 90.1%

Dealing with Scalability Issues

- Zcash uses zero-knowledge Succinct Non-interactive Arguments of Knowledge, or zk-SNARKs.

- These zk-SNARKs have very small communication and verification costs compared to other zero-knowledge arguments.

Short Pairing-based Non-interactive
Zero-Knowledge Arguments

Jens Groth

University College London

1 Introduction

Succinctness Requires Non-Standard Assumptions

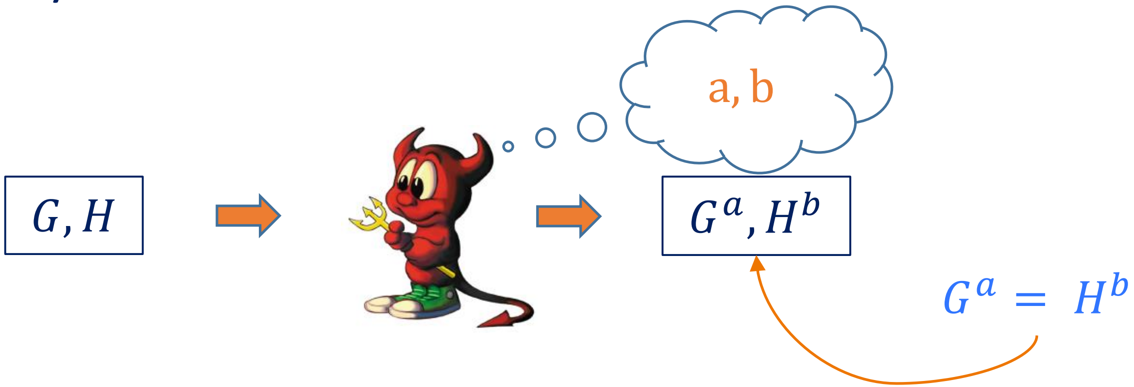
- Succinct: the proof size and verification costs are small (independent of the size of the witness).

- Provably impossible using standard “falsifiable” assumptions such as discrete log or RSA [ACM: GW11].

- Instead zk-SNARKs base their security on q-Power Knowledge of Exponent Assumptions.

Succinctness Requires Non-Standard Assumptions

- q-PKE assumptions are new, and so far only analysed in the generic group model.
- They assume that an adversary can only find G^a, H^b such that $G^a = H^b$ if the adversary also knows a and b.



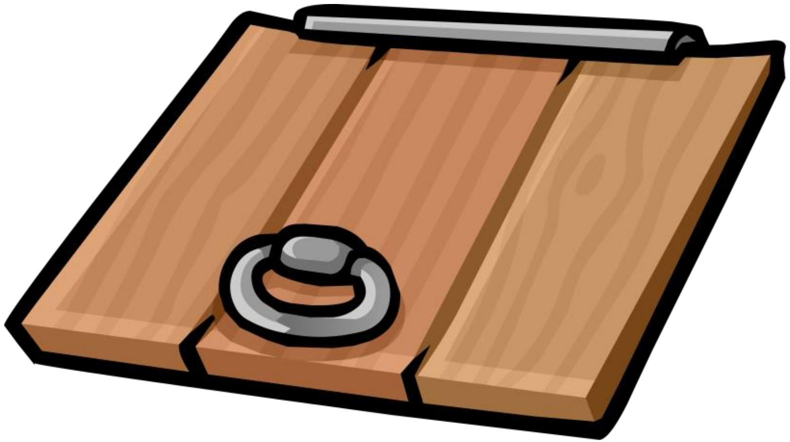
Non-Interactivity Requires a Common Reference String

- Non-Interactivity means that a single message suffices to prove a transaction valid.

- Non-interactivity is provably impossible without a common reference string [Journal of Cryptology: GO94].

Non-Interactivity Requires a Common Reference String

- Whoever generates the common reference string might keep hold of some trapdoor information.



Knowledge of Witness

Let $R = \{(x_i, w_i)\}$ be a NP complete relation such that there exists a probabilistic polynomial time algorithm

$$(\mathrm{x}, \mathrm{w}) \leftarrow YES(R)$$

such that YES outputs $(\mathrm{x}, \mathrm{w}) \in R$.

$$\boxed{R} \rightarrow \boxed{YES} \rightarrow \boxed{(\mathrm{x}, \mathrm{w}) \in R}$$

Knowledge of Witness

Let $R = \{(x_i, w_i)\}$ be a NP complete relation. Then there does not exists a probabilistic polynomial time algorithm

$$\mathrm{w} \leftarrow WitnessGen(x)$$

such that $WitnessGen$ outputs w with $(\mathrm{x}, \mathrm{w}) \in R$.

zk-SNARKs are Arguments of Knowledge?

Prover can prove knowledge because they generated the NP statement, so they have the corresponding witness.

Prover generates statement and witness.

↓

Prover publishes statement and calculates a value that they could only find if they also knew the corresponding witness.

Relations in Zcash

The relations used in ZCash are of the form:

Statements Witnesses

$$R = \{((c_0, \ldots, c_{N-1}, S);\ \ (l, r)) \mid$$

$$(\forall i \ \in [0, N-1], c_i \ \in COM) \wedge (l \in [0, N-1]) \wedge (c_l = com(S, r))\}$$

The c_i's are commitments under commitment key ck.

The l^{th} commitment is a commitment to S, r.

Relations in Zcash

Given a statement $(c_0, \ldots, c_{N-1}, S)$, it is difficult to find some witness (l, r) such that $\big((c_0, \ldots, c_{N-1}, S); (l, r)\big) \in R$.

To spend a shielded coin, a user sends a message m, a serial number S and a zk-SNARK proof π where the proof π convinces the verifier that the user knows some (l, r) such that $\big((c_0, \ldots, c_{N-1}, S); (l, r)\big) \in R$.

The zk-SNARK proof π does not reveal the (l, r) because zk-SNARKs are zero-knowledge schemes.

zk-SNARK Schemes

- 4 polynomial time algorithms: $Setup, Prove, Verify, Simulate$.
- $Setup$ is only run once in Zcash.
- $Prove$ is run by users that spend shielded coins.
- $Verify$ is run by the miners.
- Hopefully nobody runs $Simulate$ because they do not know the trapdoor τ.

$$(CRS, \tau) \leftarrow Setup(R)$$

$$\pi \leftarrow Prove(CRS, x, w)$$

$$0/1 \leftarrow Verify(CRS, x, \pi)$$

$$\pi \leftarrow Simulate(CRS, \tau, x)$$

zk-SNARK Schemes are Sound

A user can only find x, π such that $Verify(CRS, x, \pi) = 1$ if they either:

- Know some w such that $(x, w) \in R$;
- Know the trapdoor τ.

$(CRS, \tau) \leftarrow Setup(R)$

$\pi \leftarrow Prove(CRS, x, w)$

$0/1 \leftarrow Verify(CRS, x, \pi)$

$\pi \leftarrow Simulate(CRS, \tau, x)$

zk-SNARK Schemes are Zero-Knowledge

zk-SNARK Schemes are Zero-Knowledge

$$(CRS, \tau) \leftarrow Setup(R)$$

zk-SNARK Schemes are Zero-Knowledge

CRS is the common reference string known to all parties.

τ is the trapdoor.

$(CRS, \tau) \leftarrow Setup(R)$

CRS

CRS, τ

$\pi \leftarrow Prove(CRS, x, w)$

$\pi \leftarrow Simulate(CRS, \tau, x)$

zk-SNARK Schemes are Zero-Knowledge

$(CRS, \tau) \leftarrow Setup(R)$

CRS

CRS, τ

$\pi \leftarrow Prove(CRS, x, w)$

$\pi \leftarrow Simulate(CRS, \tau, x)$

π

zk-SNARK Schemes are Zero-Knowledge

$(CRS, \tau) \leftarrow Setup(R)$

CRS

CRS, τ

$\pi \leftarrow Prove(CRS, x, w)$

$\pi \leftarrow Simulate(CRS, \tau, x)$

π

zk-SNARK Schemes are Zero-Knowledge

$(CRS, \tau) \leftarrow Setup(R)$

CRS

$\pi \leftarrow Prove(CRS, x, w)$

CRS, τ

$\pi \leftarrow Simulate(CRS, \tau, x)$

Was π generated using the trapdoor or the witness?

What if Someone Knows the Trapdoor?

<u>What they can do:</u>

- Invent their own choice of serial number S.

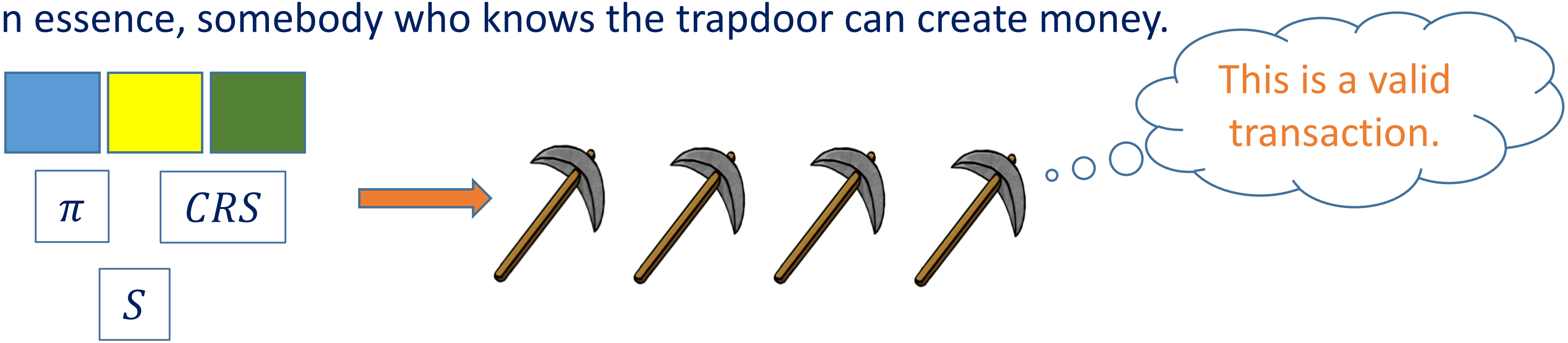
- Use the trapdoor to find a proof that a commitment on the ledger contains S.

What if Someone Knows the Trapdoor?

<u>What they can do:</u>

- Even though no commitment on the ledger contains S, miners will accept the proof as they cannot distinguish it from a genuine proof.

- In essence, somebody who knows the trapdoor can create money.



What if Someone Knows the Trapdoor?

<u>What they cannot do:</u>

- Break anonymity – i.e., they cannot use the trapdoor to uncover which commitment contain which serial number.

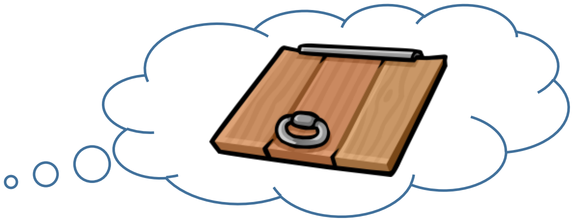
- Steal other peoples money.

So Nobody Knows this Trapdoor, Right?

- If somebody knew the internal state of the $Setup$ algorithm, including all inputs, outputs and random coins, then in the current scheme they would know the trapdoor.

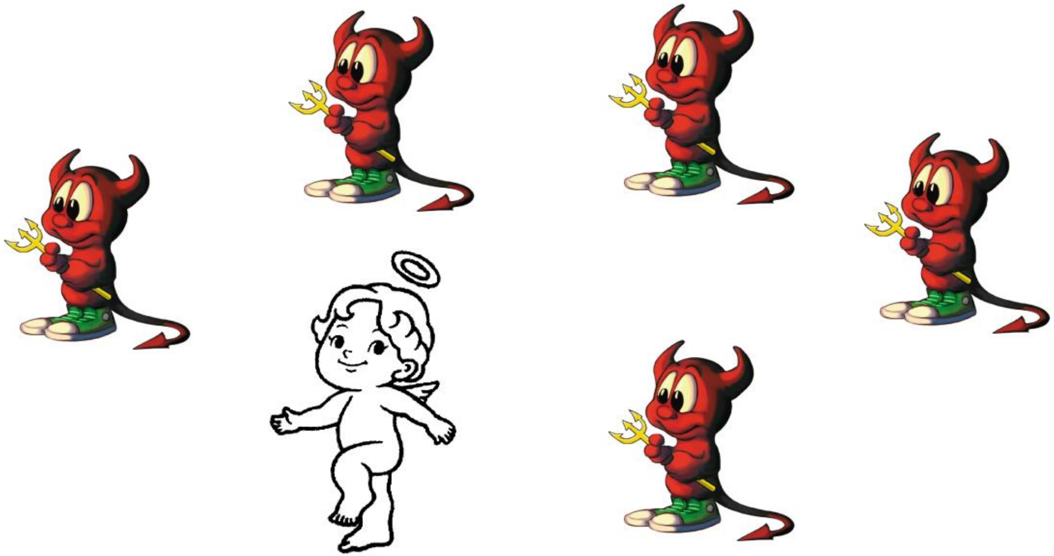
- To deal with this problem, the Zcash company ran a trusted setup ceremony, in which 6 people participated in a multiparty computation protocol to generate the common reference string.

$Setup$



So Nobody Knows this Trapdoor, Right?

- If a single one of these 6 participants is honest, then none of them know the trapdoor.



The Participants in the Trusted Setup Ceremony

Andrew Miller:
Assistant Prof. at the University of Illinois

Peter Van Valkenberg:
Director of Research at Coin Center

John Dobbertin:
Pseudonym

Zooko Wilcox:
Zcash Founder and Chief Ex.

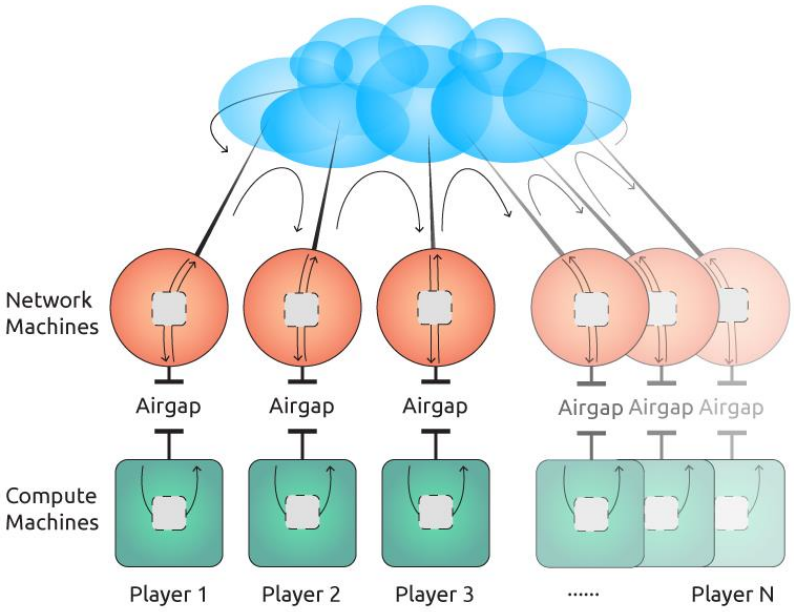
Derek Hinch:
NCC Group

Peter Todd:
Bitcoin Core Developer

The Trusted Setup Ceremony

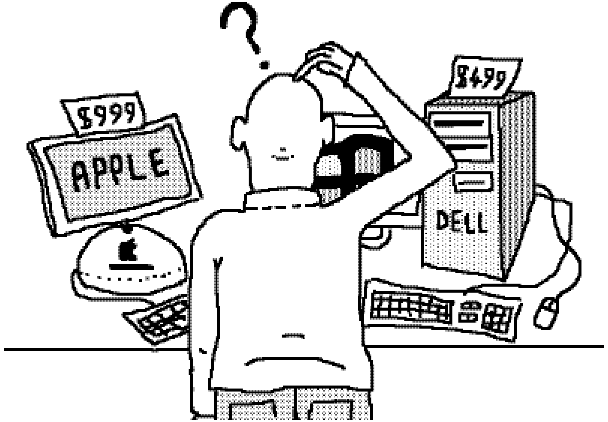
To prevent potential eavesdroppers, the participants each ran their part of the computation on ``air-gapped'' machines; i.e., computers that were disconnected from all networks.

They then used separate machines, or ``network nodes,'' in order to send the necessary information among the other participants.



The Trusted Setup Ceremony

All the computers were purchased specifically for the ceremony, and some of them were destroyed afterwards.



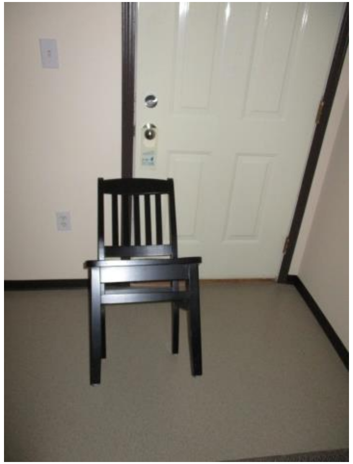
The Trusted Setup Ceremony

The participants burned to disk all messages sent on the network nodes, in order to provide a transcript that could serve as evidence.

Some of the participants filmed the process.

The Trusted Setup Ceremony

One of the participants, Peter Todd, wrote a blog post on the process: petertodd.org/2016/cypherpunk-desert-bus-zcash-trusted-setup-ceremony.



The Trusted Setup Ceremony

It is impossible for the participants to provide concrete evidence of their non-collusion.

There was still a large effort made to bring transparency into the process and convince people that the currency was safe to use.

So Nobody Knows this Trapdoor, Right?

Even if the 6 participants running the Setup Ceremony did not collude and do not know the trapdoor, the common reference string is only computationally secure.

If somebody breaks the discrete log problem once, then they can extract the trapdoor.

When computers get faster, the Setup Ceremony may need to be run again.

Efficiency Concerns

The $Setup$ algorithm is very slow, however it is only run once (in the Trusted Setup Ceremony).

The size of the public parameters needed to spend minted coins is 888, 842 KB.

The size of the public parameters needed to verify transactions is 2 KB.

$$(CRS, \tau) \leftarrow Setup(R)$$

$$\pi \leftarrow Prove(CRS, x, w)$$

$$0/1 \leftarrow Verify(CRS, x, \pi)$$

$$\pi \leftarrow Simulate(CRS, \tau, x)$$

z.cash/technology/paramgen.html

Efficiency Concerns

The proof size small - currently about 288 bytes*.

The spender computation is high. Founder Zooko Wilcox said that spending a minted coin
“takes a whole minute or two on like a high-powered, supercomputer 64-bit laptop CPU.”**

$(CRS, \tau) \leftarrow Setup(R)$

$\pi \leftarrow Prove(CRS, x, w)$

$0/1 \leftarrow Verify(CRS, x, \pi)$

$\pi \leftarrow Simulate(CRS, \tau, x)$

*According to the Zcash Improvement Proposals
**bitcoinmagazine.com/articles/zcash-ceo-zooko-discusses-privacy-and-efficiency-tradeoffs-vs-the-bitcoin-blockchain-1458829054/

Efficiency Concerns

The $Verify$ algorithm is very fast (a few milliseconds per proof).

$(CRS, \tau) \leftarrow Setup(R)$

$\pi \leftarrow Prove(CRS, x, w)$

$0/1 \leftarrow Verify(CRS, x, \pi)$

$\pi \leftarrow Simulate(CRS, \tau, x)$

Zcash Operations

The $Setup$ algorithm is run just once.

$Setup$
$CreateAddress$
$Mint$
$Pour$
$VerifyTransaction$
$Receive$

Zcash Operations

The $CreateAddress$ algorithm can be run by any user that wishes to receive coins.

It takes public parameters as input, and outputs a public key and a corresponding secret key.

- $Setup$
- $CreateAddress$
- $Mint$
- $Pour$
- $VerifyTransaction$
- $Receive$

Zcash Operations

The $Mint$ algorithm can be run by a user to create a shielded coin.

It takes the public parameters, the coin value, and the destination address as input.

It outputs a coin (to be kept secret) and a transaction containing the commitment and the value.

- $Setup$
- $CreateAddress$
- $Mint$
- $Pour$
- $VerifyTransaction$
- $Receive$

Zcash Operations

The $Pour$ algorithm can be run by a user to transfer value from input coins into new output coins.

Pouring allows users to subdivide coins into smaller denominations, merge coins, and transfer ownership of anonymous coins, or make public payments.

$Setup$

$CreateAddress$

$Mint$

$Pour$

$VerifyTransaction$

$Receive$

Zcash Operations

The $Pour$ algorithm takes as input:

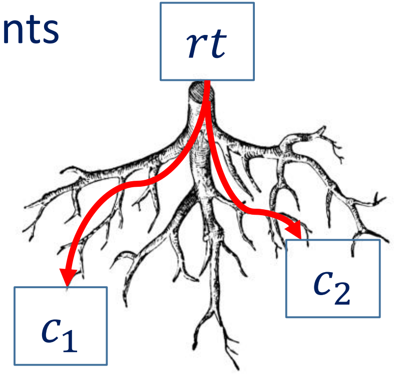
- Two distinct input coins, along with corresponding address secret keys.

- $Setup$
- $CreateAddress$
- $Mint$
- $Pour$
- $VerifyTransaction$
- $Receive$

Zcash Operations

The $Pour$ algorithm takes as input:

- A Merkle root (equals the root of Merkle tree over all coin commitments so far)
- Two authentication paths for the two coin commitments



Zcash Operations

The *Pour* algorithm takes as input:

- The values of two new anonymous coins to be generated, and two input address public keys.

Zcash Operations

The $Pour$ algorithm takes as input:

- A third value specifying the amount to be publicly spent (e.g., to redeem coins or pay transaction fees).

€€

$Setup$

$CreateAddress$

$Mint$

$Pour$

$VerifyTransaction$

$Receive$

Zcash Operations

The $Pour$ algorithm outputs two new coins and a transaction containing the commitments, the values, the merkle tree root, and the old serial numbers.

- $Setup$
- $CreateAddress$
- $Mint$
- $Pour$
- $VerifyTransaction$
- $Receive$

Zcash Operations

The $VerifyTransaction$ algorithm is run by the miners.

It takes the public parameters, a (mint or pour) transaction, and the current state of the ledger as input.

It outputs $0/1$.

- $Setup$
- $CreateAddress$
- $Mint$
- $Pour$
- $VerifyTransaction$
- $Receive$

Zcash Operations

The *Receive* algorithm scans the ledger and retrieves unspent coins paid to a particular user address.

It takes as input a public address, its secret key, and the current state of the ledger as input.

It outputs a set of unspent coins.

Setup

CreateAddress

Mint

Pour

VerifyTransaction

Receive

Zcash General Stats

As of 15/03/2017 (coinmarketcap.com),

- There are 203 565 known accounts
- Market Cap is $39, 864, 183 (Bitcoin's is $20, 265, 831, 377 and Ethereum's is $3, 417, 689, 177)
- One ZEC is worth $45.26 (Bitcoin's is $1245.93 and Ethereum's is $29.57)
- There are 868,569 total ZEC's in circulation

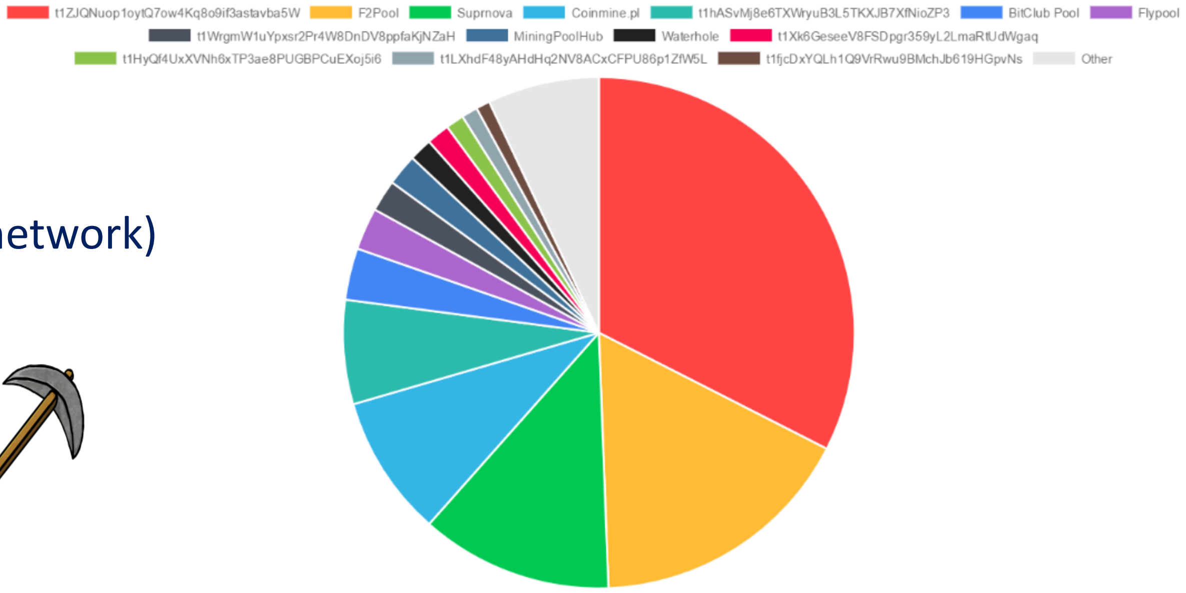
Zcash Mining Stats

As of 15/03/2017 (explorer.zcha.in/statistics/network),

- The miners reward for one block is 10 ZEC
- The max block size is 2MB

Zcash Mining Stats

As of 15/03/2017,
(explorer.zcha.in/statistics/network)



Top 15 miners, by count of blocks mined.

Zcash Founders Reward Stats

As of 15/03/2017 (explorer.zcha.in/statistics/network),

- The cumulative Founders reward is 173773.75 ZEC
- This is about 7 million USD worth.