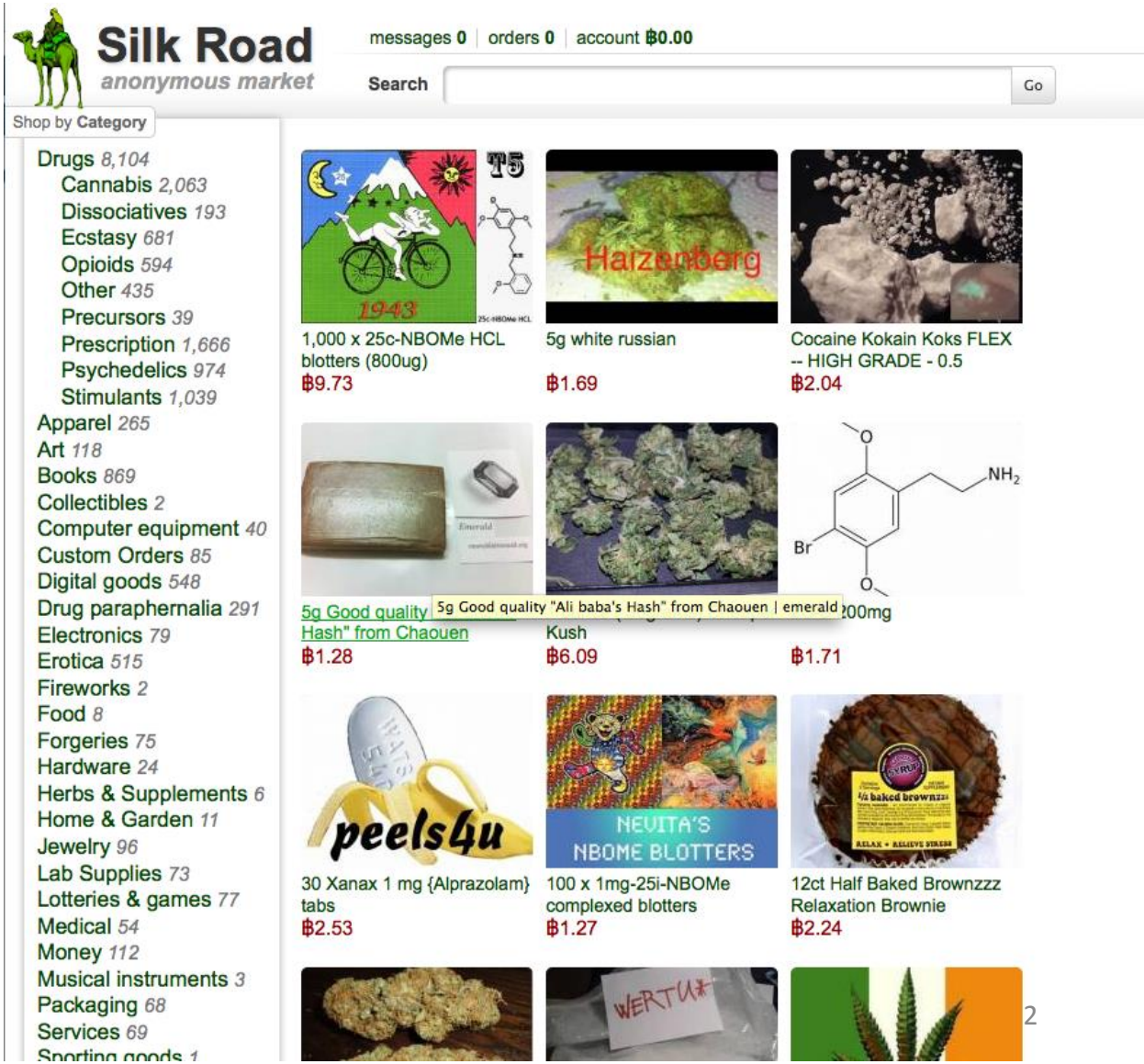
Anonymity of E-Cash Protocols

Erman Ayday

Disclaimer

It is debatable that anonymous e-cash protocols are also useful for black market and money laundering



Bitcoin

S. Nakamoto, 2008

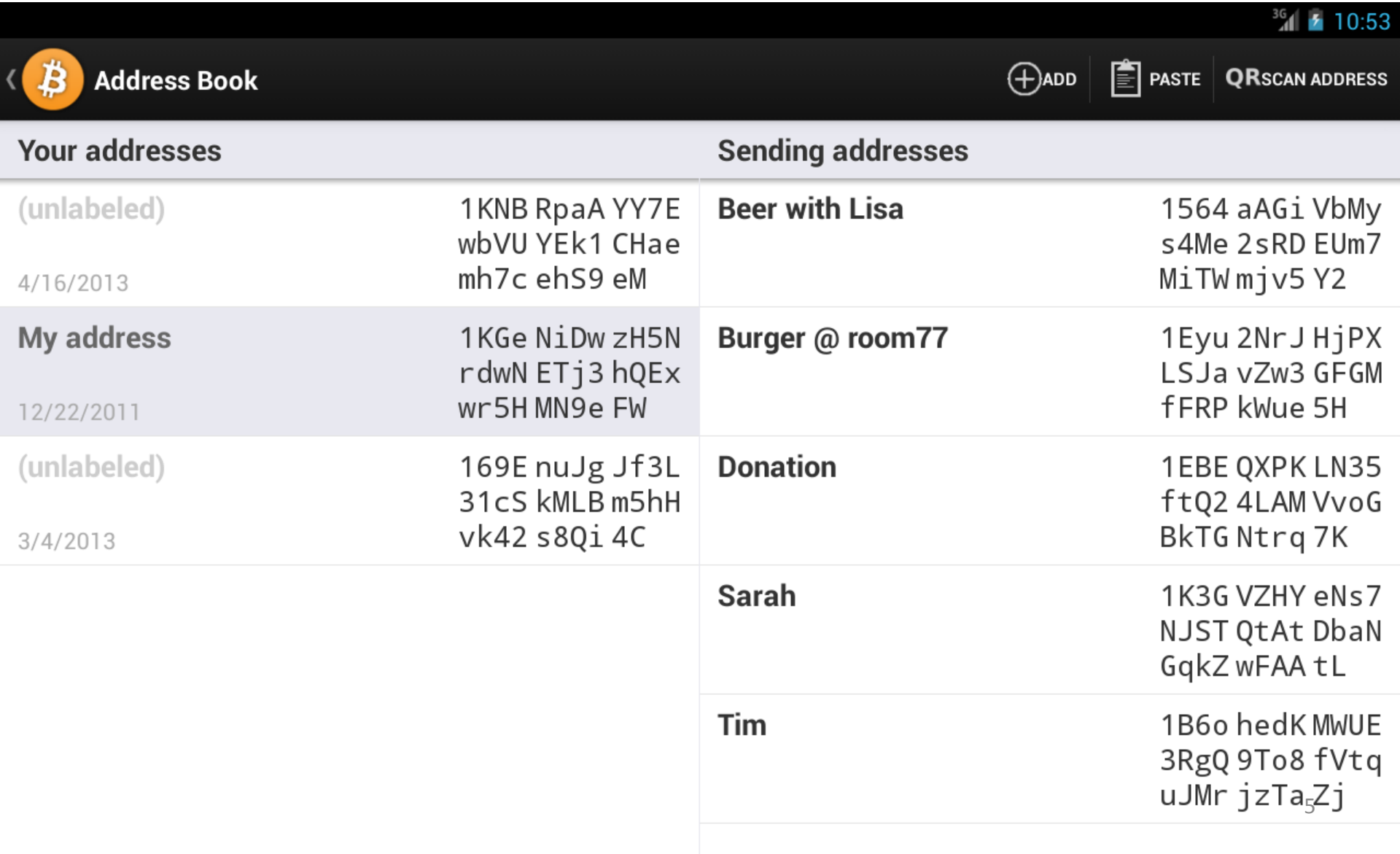
- A software-based online payment system described by Satoshi Nakamoto
- Decentralized digital currency:
  - Payments work peer-to-peer without a central repository or single administrator
- Merchants have an incentive to accept the digital currency
  - Transaction fees are lower than credit cards
  - Some mainstream websites began accepting bitcoins
- Payments are recorded in a **public ledger (or blockchain)**

Digital Wallet

- Bitcoin uses public-key cryptography
  - Public key: an account number or name
    - Bitcoin address: H(public key)
  - Private key: ownership credentials
- Wallet is a collection of these keys
  - Also generates these keys

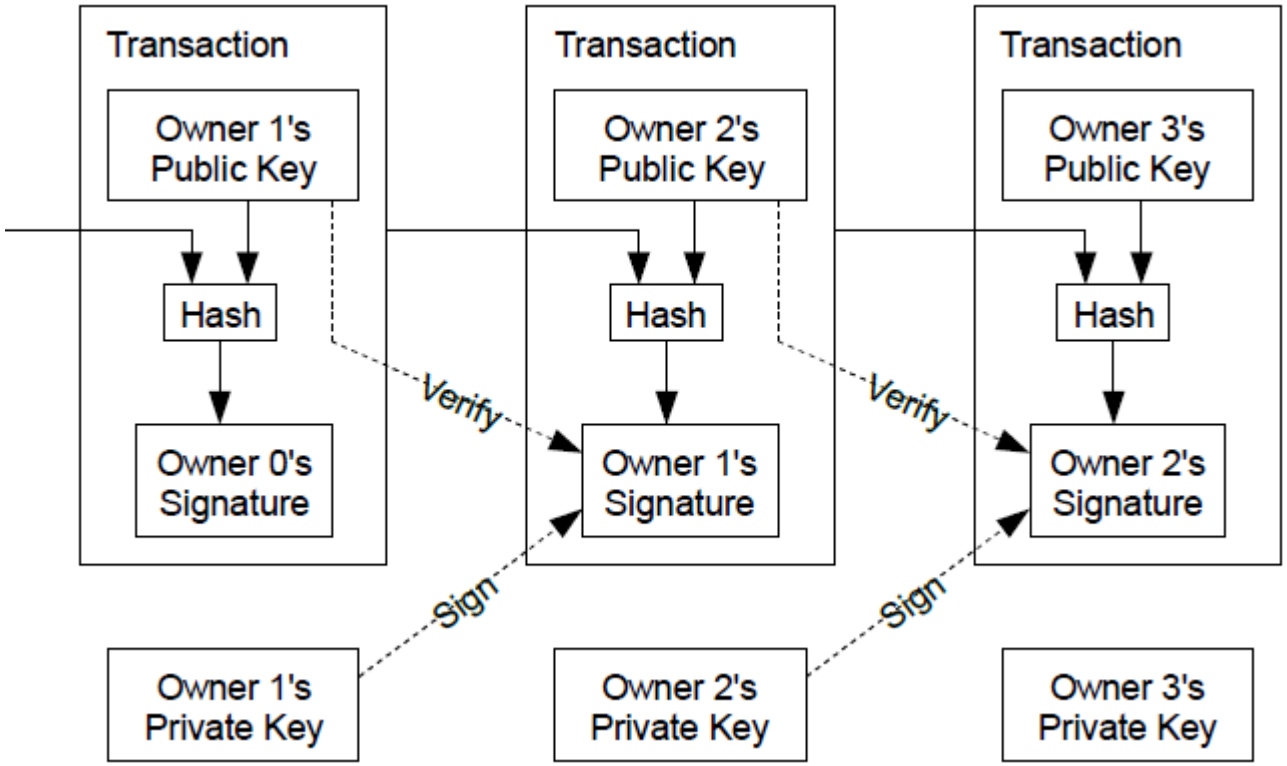
- Ownership of Bitcoins associated with a certain address is demonstrated with knowledge of the private key
- If a private key is lost, the user cannot prove ownership
- Risk of theft can be reduced by
  - Generating keys offline on an uncompromised computer
  - Saving the keys on external storage or paper printouts

Digital Wallet (Android)



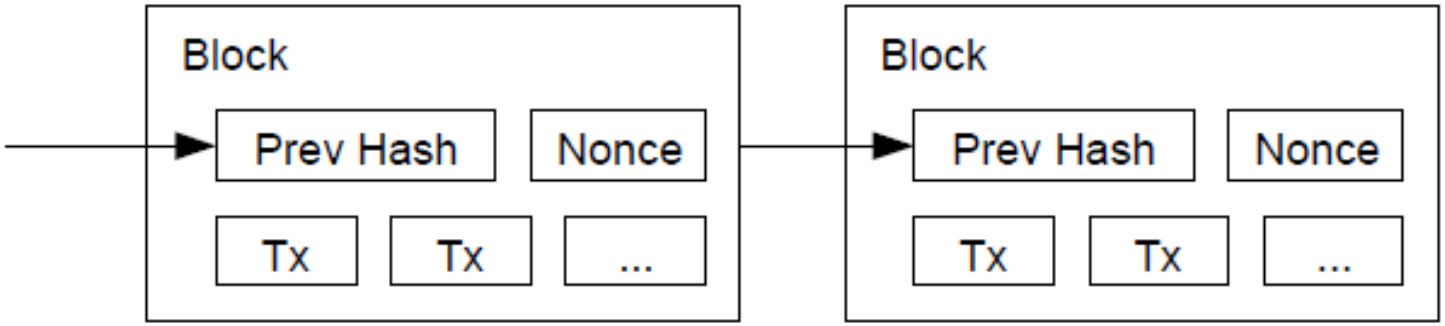
Transactions

- Permanently transfers ownership to a new address
- Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner
- A payee can verify the signatures to verify the chain of ownership

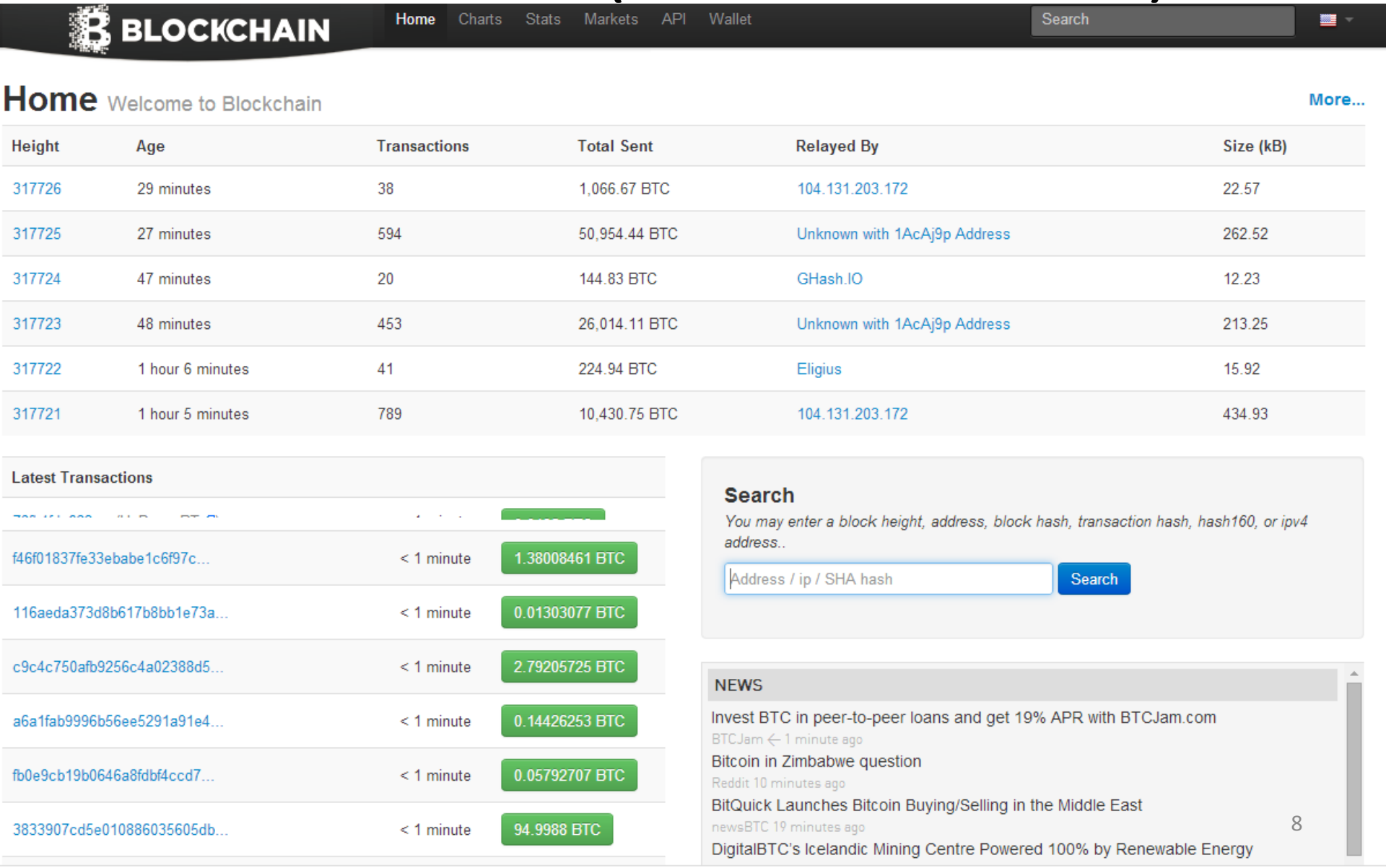


Public Ledger – The Blockchain

- Bitcoin servers can validate the transactions (*mining*), add them to their copy of the ledger, and then broadcast these ledger additions to other servers
  - Solution for preventing double-spends
- Six times per hour, a group of accepted transactions (a *block*) is added to the blockchain



Blockchain (blockchain.info)



Mining

- Maintaining the blockchain
- Miners process payments by verifying each transaction and adding it to the blockchain
- Individuals or companies engage in this activity in exchange for transaction fees and newly created bitcoins
  - Transaction fee is optional, but may speed up confirmation
- Reward will be removed when an arbitrary limit of 21 million Bitcoins is reached (approximately in 2140)
  - There are roughly 13 million Bitcoins as of August 2014
- Recently became very competitive (specialized technology)

How a ₿itcoin transaction works

Bob, an online merchant, decides to begin accepting bitcoins as payment. Alice, a buyer, has bitcoins and wants to purchase merchandise from Bob.

WALLETS AND ADDRESSES

Bob and Alice both have Bitcoin "wallets" on their computers.

Wallets are files that provide access to multiple Bitcoin addresses.

An address is a string of letters and numbers, such as 1HULMwZEP kjEPeCh 43BeKJL1yb LCWrfDpN.

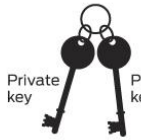
CREATING A NEW ADDRESS

Bob creates a new Bitcoin address for Alice to send her payment to.

Each address has its own balance of bitcoins.

SUBMITTING A PAYMENT

Alice tells her Bitcoin client that she'd like to transfer the purchase amount to Bob's address.



Public Key Cryptography 101
When Bob creates a new address, what he's really doing is generating a "cryptographic key pair," composed of a private key and a public key. If you sign a message with a private key (which only you know), it can be verified by using the matching public key (which is known to anyone). Bob's new Bitcoin address represents a unique public key, and the corresponding private key is stored in his wallet. The public key allows anyone to verify that a message signed with the private key is valid.

It's tempting to think of addresses as bank accounts, but they work a bit differently. Bitcoin users can create as many addresses as they wish and in fact are encouraged to create a new one for every new transaction to increase privacy. So long as no one knows which addresses are Alice's, her anonymity is protected.

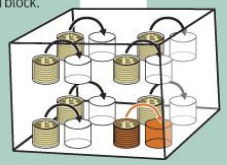
Alice's wallet holds the private key for each of her addresses. The Bitcoin client signs her transaction request with the private key of the address she's transferring bitcoins from.

Anyone on the network can now use the public key to verify that the transaction request is actually coming from the legitimate account owner.

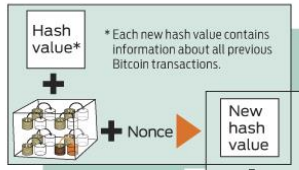
VERIFYING THE TRANSACTION

Gary, Garth, and Glenn are Bitcoin miners.

Their computers bundle the transactions of the past 10 minutes into a new "transaction block."



The miners' computers are set up to calculate cryptographic hash functions.



The mining computers calculate new hash values based on a combination of the previous hash value, the new transaction block, and a nonce.

Cryptographic Hashes
Cryptographic hash functions transform a collection of data into an alphanumeric string with a fixed length, called a hash value. Even tiny changes in the original data drastically change the resulting hash value. And it's essentially impossible to predict which initial data set will create a specific hash value.

The root of all evil	6d0a 1899 086a... (56 more characters)
The root of all evil	486c 6be4 6dde...
The root of all veil	b8db 7ee9 8392...

Nonces
To create different hash values from the same data, Bitcoin uses "nonces." A nonce is just a random number that's added to data prior to hashing. Changing the nonce results in a wildly different hash value.

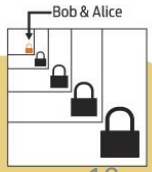
The root of all evil ???	0000 0000 0000 ...

Creating hashes is computationally trivial, but the Bitcoin system requires that the new hash value have a particular form—specifically, it must start with a certain number of zeros.

The miners have no way to predict which nonce will produce a hash value with the required number of leading zeros. So they're forced to generate many hashes with different nonces until they happen upon one that works.

Each block includes a "coinbase" transaction that pays out 50 bitcoins to the winning miner—in this case, Gary. A new address is created in Gary's wallet with a balance of newly minted bitcoins.

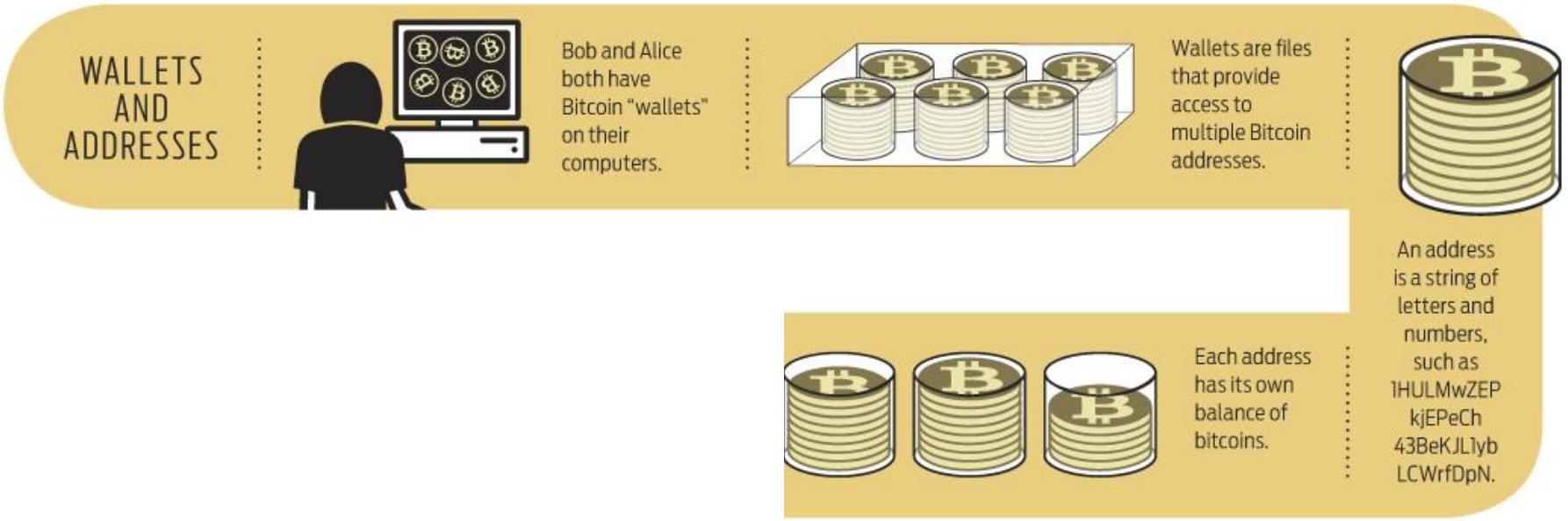
TRANSACTION VERIFIED



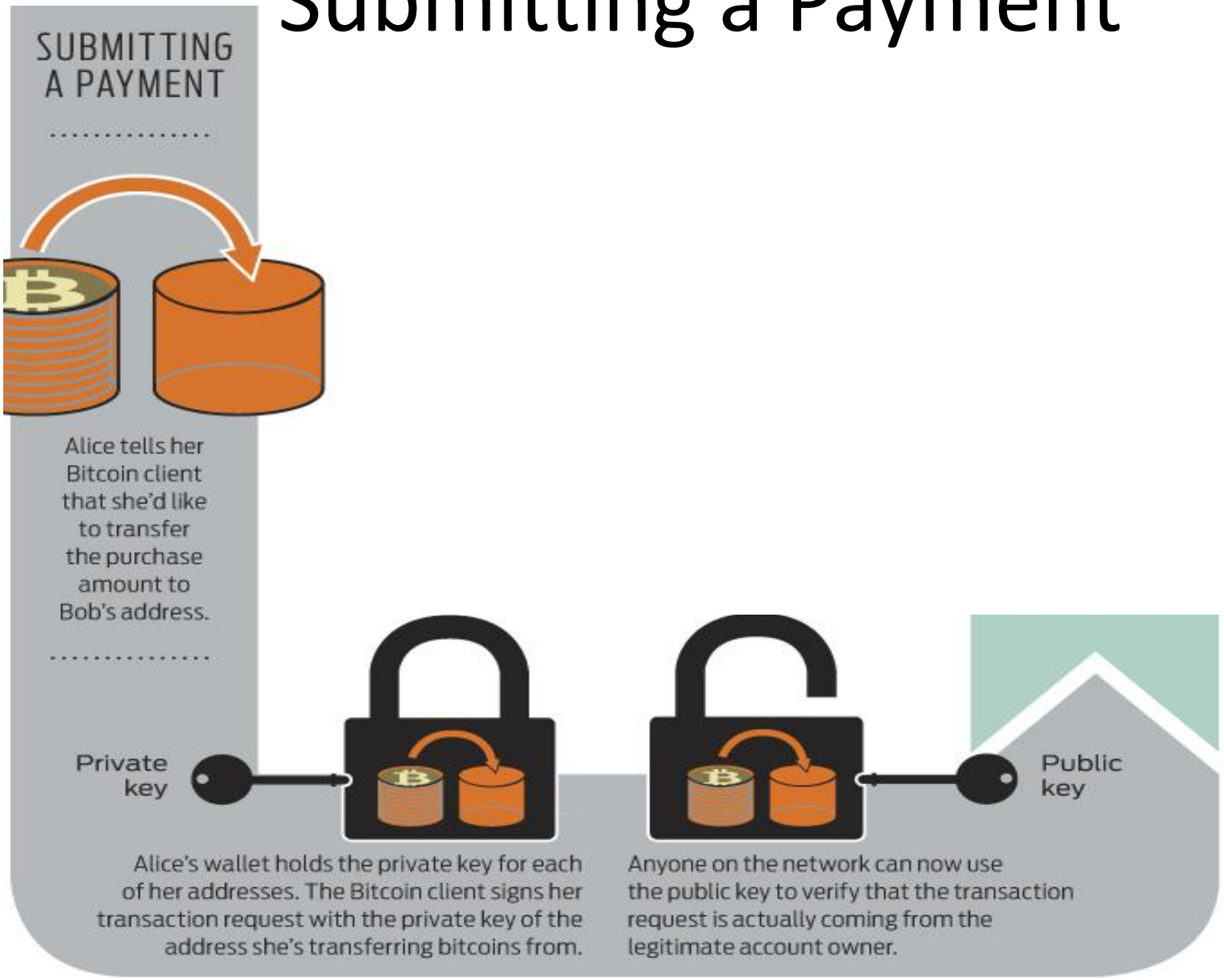
As time goes on, Alice's transfer to Bob gets buried beneath other, more recent transactions. For anyone to modify the details, he would have to redo the work that Gary did—because any changes require a completely different winning nonce—and then redo the work of all the subsequent miners. Such a feat is nearly impossible.

How a Bitcoin Transaction Works

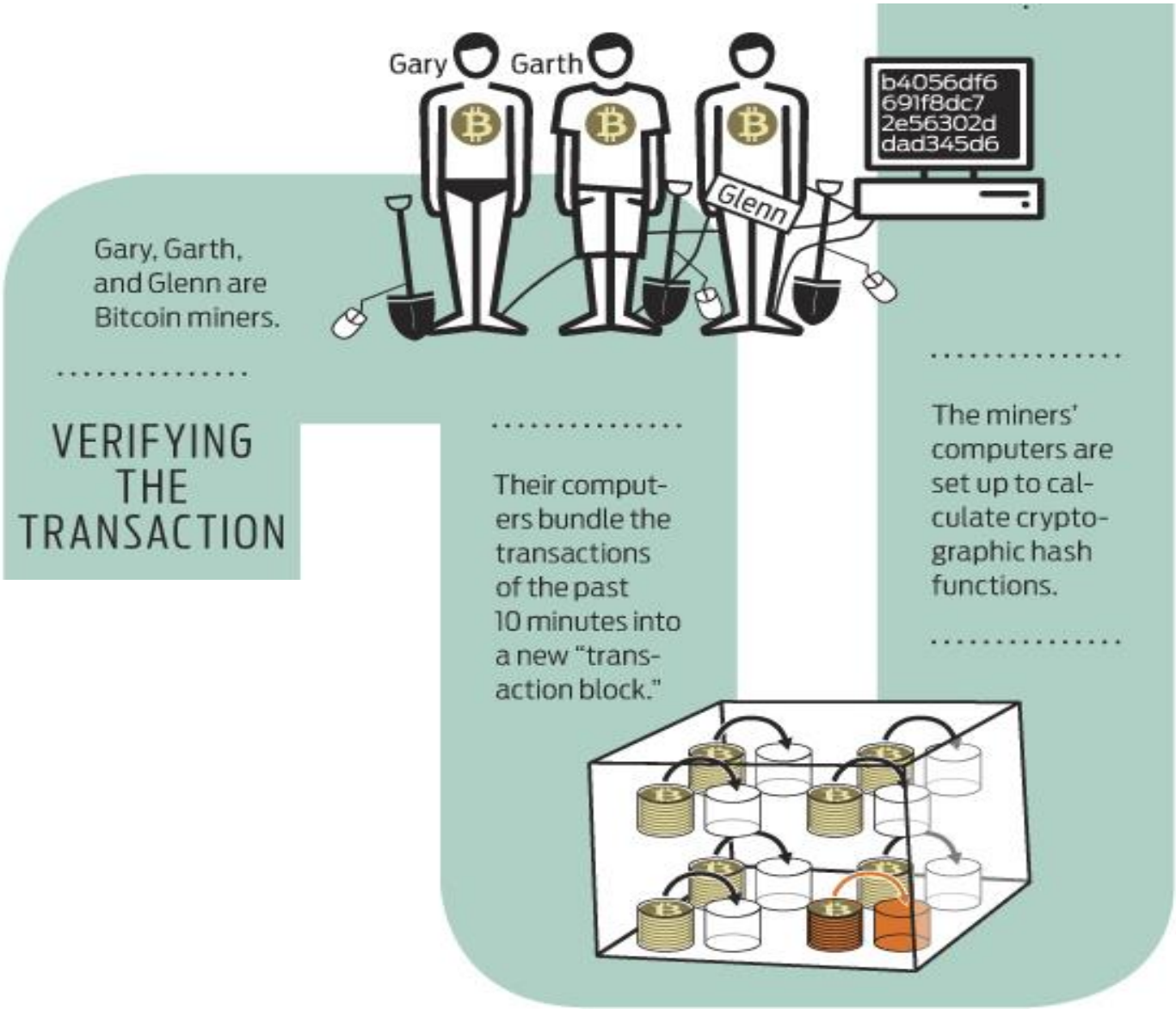
- Alice wants to purchase merchandise from Bob using Bitcoins



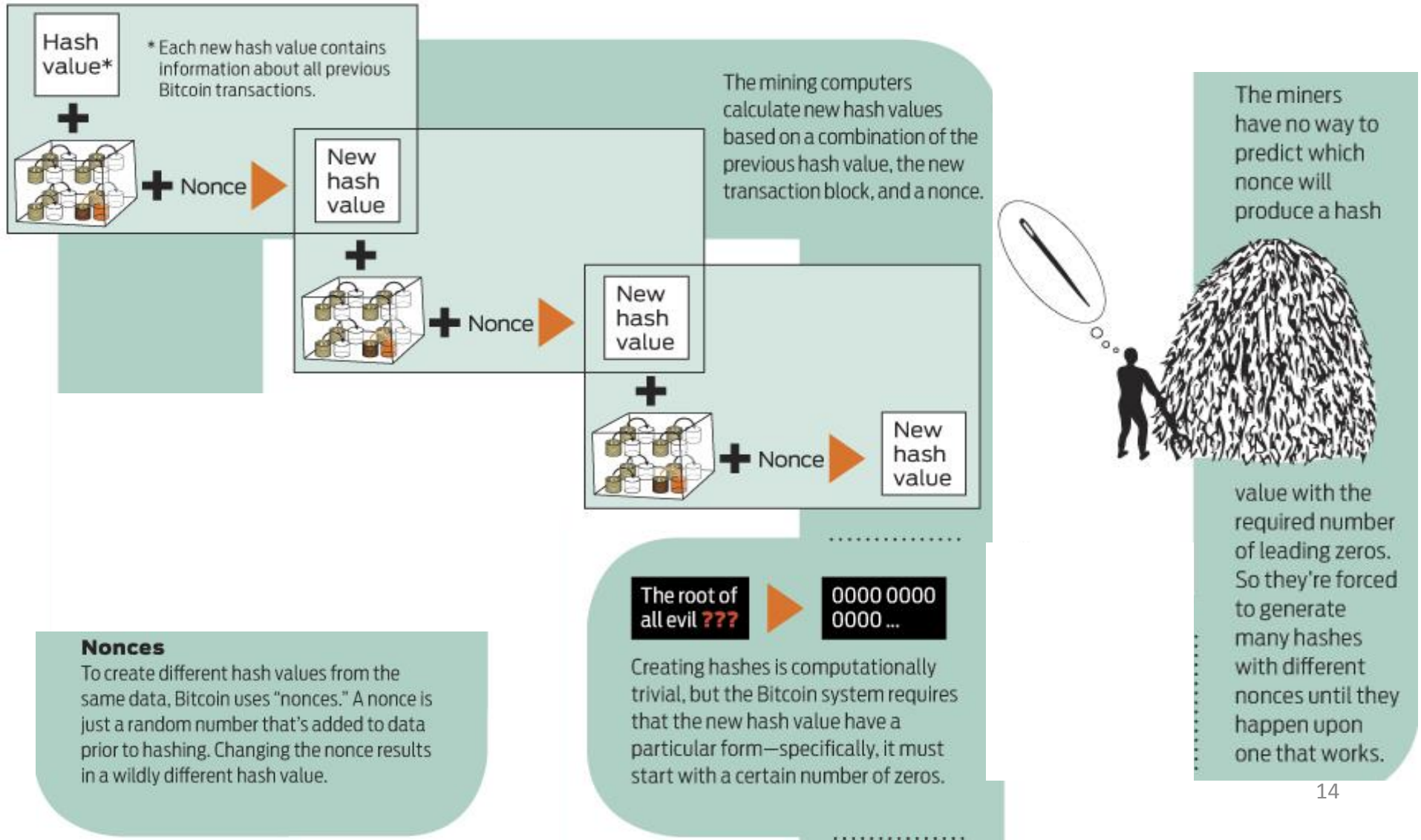
Submitting a Payment



Verifying a Transaction



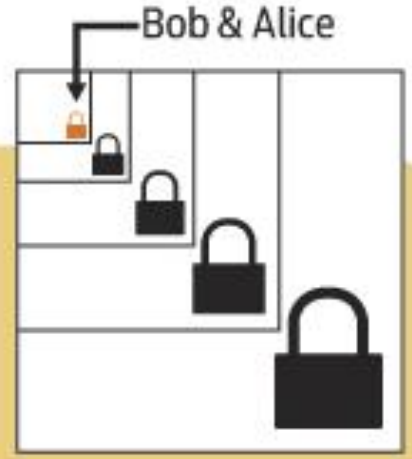
Maintaining the Blockchain



Verified Transaction

TRANSACTION VERIFIED

As time goes on, Alice's transfer to Bob gets buried beneath other, more recent transactions. For anyone to modify the details, he would have to redo the work that Gary did—because any changes require a completely different winning nonce—and then redo the work of all the subsequent miners. Such a feat is nearly impossible.



Security of Bitcoin

- Protects the blockchain by using digital signatures and cryptographic hashes
- The Addition Attack
  - An attacker Eve steal money from Alice by adding fake transactions to the blockchain
    - E.g., Alice pays Eve 100 bitcoins
  - Prevented by requiring every transfer to be digitally signed with the payer's private key
- The Modification Attack
  - Eve can modify that blockchain after a transaction
    - E.g., Eve pays Alice 100 bitcoins -> Eve pays Alice 1 bitcoin
  - Prevented by requiring
    - Transactions to be entered to the blockchain in blocks
    - Each block accompanied by a cryptographic hash of the previous block and a nonce
- Double Spending
  - Payer can spend the same money again and again
  - Alice can check the blockchain to verify Eve actually owns required Bitcoins
  - Alice may not deliver the goods until Eve's payment to Alice appears in the blockchain
    - Typically involves waiting about ten minutes
    - Might be less with transaction fees

Anonymity on Bitcoin

- Many people using Bitcoin don't want others to know who they are
- Connect to Bitcoin network via Tor
- To obfuscate the link between individual and transaction, a different bitcoin address for each transaction can be used
  - One person could hold multiple addresses in his wallet, and in theory, there would be nothing to link those addresses together
- External laundry (mixing) services
  - Allow users to trade bitcoins whose transaction history implicates them for coins with different transaction histories
  - E.g. BitLaundry

BitLaundry

BitLaundry - For all your Bitcoin washing needs.

How it works

How BitLaundry works

BitLaundry is designed to help unlink accounts from each other. It does that by providing a well-known, and hopefully popular service. Here's how it works:

1. Imagine that Alice wishes to send Bitcoins to Bob.
2. Bob, sadly, is not well liked. Alice would rather not have anyone know that she sent Bob Bitcoins.
3. So, Alice enters Bob's Bitcoin address into the form at BitLaundry, and selects a delivery schedule.
4. Alice gets a one-time-use address from BitLaundry.
5. Alice sends her Bitcoins to that address, and they get all mixed up with BitLaundry's other Bitcoins.
6. BitLaundry waits until Alice's Bitcoins are received with 10 confirmations.
7. BitLaundry deletes the database link between the one-time-use address and Bob's address.
8. BitLaundry sends Bitcoins out to Bob according to the delivery schedule.

Tips

Send Bitcoins to yourself to obscure their history.

Use multiple recipient addresses and/or spread the transactions over a number of days to thwart correlation attacks!

Fees

2.4900% of the total you send, plus **BTC 0.00149** per outgoing transaction

Example: Let's distribute BTC 10 to 7 recipients over 3 days with 2 transactions per recipient per day:

The base fee:
BTC 10 * 0.0249 = BTC 0.2490

plus the transaction fee:

(Not So Much) Anonymity on Bitcoin

- Everything that happens in the Bitcoin world is trackable
- Every Bitcoin-based transaction is logged in the blockchain
  - If you choose to engage in sensitive transactions on Bitcoin, you should be aware that a record will be preserved for all eternity
- If your Bitcoin address goes public, everyone in the world will know your bitcoin balance and transactions
  - Can also be inferred using auxiliary information about a specific target (time of transaction, merchant, etc.)
  - Inferring personal information by analyzing large datasets is not far fetched (remember Netflix or MA Governor)
- Privacy is not enforced by the Bitcoin protocol design

What happens in the blockchain stays in the blockchain

Why Existing Methods are not Good Enough?

- Create a lot of Bitcoin addressed
  - Who has time for that?
  - Correlation between the addresses can be inferred
- Use an external laundering service
  - Laundry must be trusted
  - The service can be malicious (no anonymity)
  - The service can go out of business
  - Bitcoins can be stolen
- Users usually don't want to expand continual effort in protecting their privacy
- Users are typically not sufficiently aware of their compromised privacy

Evaluating User Privacy in Bitcoin

Androulaki et al. 2013

- **Goal:** Evaluate the privacy that is provided by Bitcoin
- Investigates the behavior of Bitcoin client and exploiting its properties
- Through a novel simulator that mimics the use of Bitcoin as the primary currency within a university setting

Adversary Model

- Adversary *A* does not only have access to public log, but is also part of the Bitcoin system
- Can also incur one or more transactions through Bitcoin
- Can have access to the (public) addresses of some vendors along with (statistical) information
  - Such as the pricing of items or the number of their clients within a specified amount of time
- Computationally bounded

Quantifying Privacy in Bitcoin

- *Activity unlinkability*
  - An adversary *A* should not be able to link two different addresses (address unlinkability) or transactions (transaction unlinkability)
- *Profile indistinguishability*
  - (in-)ability of *A* to reconstruct the profiles of *all the users* that participate in pubLog
- Defined both definitions as a game between the adversary and a challenger
- Quantified user privacy in terms of adversary's advantage in winning the games

Tools - Heuristics

- Multi-input transactions
  - When the BTC amount is not enough in one address, multiple addresses (in one's wallet) are used for a transaction
  - If these BTCs are owned by different addresses, then the input addresses belong to the same user
- Shadow addresses
  - Automatically created and used to collect back the "change" that results from any transaction issued by the user
  - When a Bitcoin transaction has two output addresses *x* and *y*, and if *x* has appeared in the public log before:
    - *y* is a shadow address

Tools – Behavioral Analysis

- *Adversary A* also uses behavior-based clustering techniques
  - K-Means (KMC), Hierarchical Agglomerative Clustering (HAC) algorithms
- Goal: Output a group of clusters of addresses that approximates the Bitcoin users the best
  - Utilizing the similarities between Bitcoin addresses
  - Time of transaction, value, etc.
- HAC is applied on top of the results received using heuristics
- KMC is applied on top of results obtained via HAC
- Refer to the paper for details

Results and Limitations

- Behavior-based clustering techniques can unveil the profiles of 40% of Bitcoin users
  - With 80% accuracy
  - Even if these users try to enhance their privacy by manually creating new addresses
- Limitations:
  - Experimental setup (not real data)
  - Laundry service is not considered as a privacy tool

Zerocoin: Towards Anonymous Bitcoin

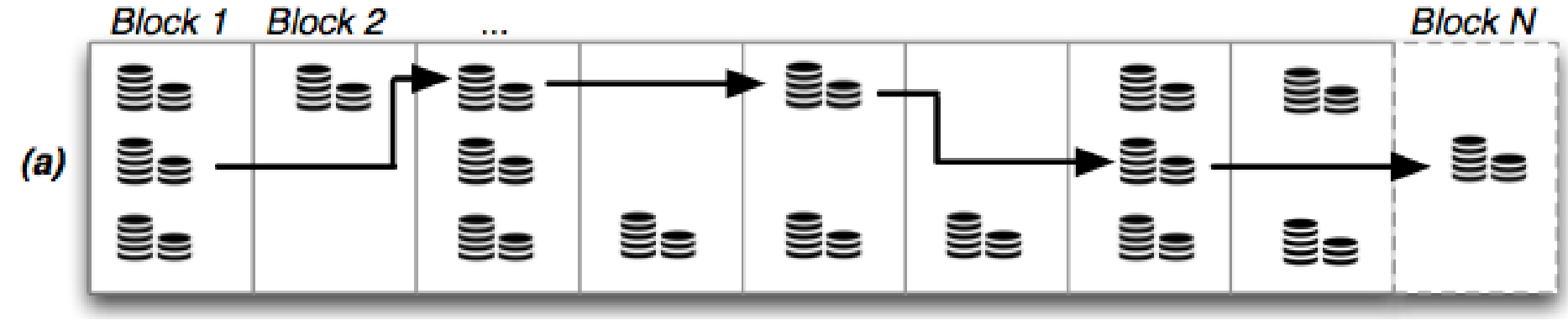
Miers et al. 2013

- Effectively builds a money-laundering service into a crypto currency at the *protocol level*
  - Eliminates any reliance on trusted third parties
- Zerocoins are purchased with Bitcoin in fixed denominations
  - *Zerocoin mint*
- System takes original Bitcoins, turns them into Zerocoins, and then turns them back into new Bitcoins in another wallet
- Anyone with a Bitcoin can spend it to create a Zerocoin
  - There is a serial number inside of every Zerocoin
  - Each Zerocoin is like the encryption of that serial number
- Users can come back at any time to redeem their Bitcoins

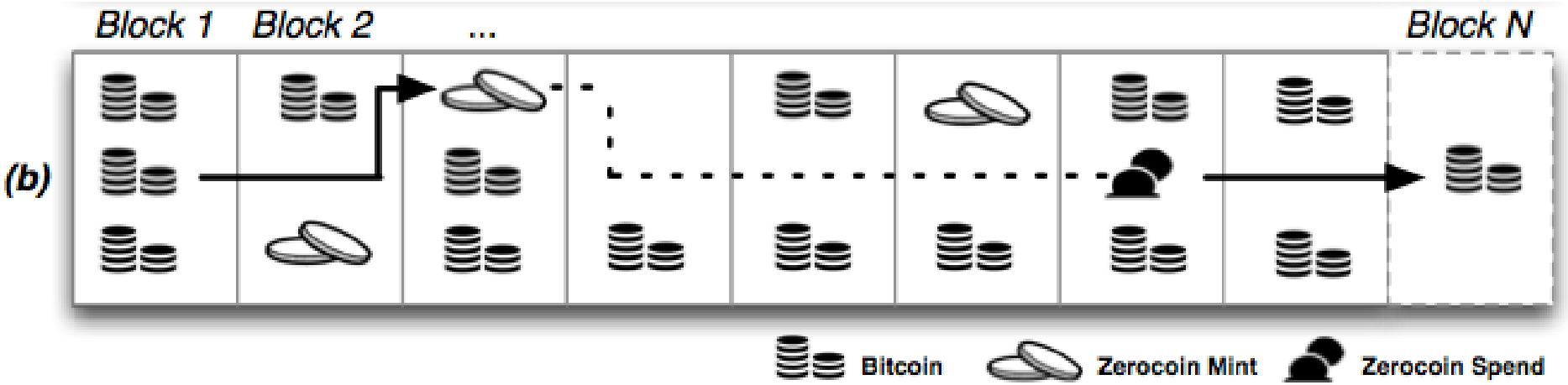
Zerocoin - Simplified (by Matthew Green)

- People throw dollars into a hat
- Each time they throw a dollar, they get a token
  - All the tokens look exactly the same
- Bob comes back with a mask on
  - Or gives his token to a friend and he goes back
- Bob exchanges his token, and he takes out a totally different dollar

Zerocoin vs. Bitcoin



Bitcoin block chain. Each transaction is tied to the one that precedes it



Bitcoin/Zerocoin block chain. A user transforms Bitcoins into a Zerocoin, then "Spends" it to redeem the Bitcoins. The linkage between Mint and Spend (dotted line) cannot be determined from the blockchain data

Figure: Miers et al. Zerocoin: Anonymous Distributed E-Cash from Bitcoin.

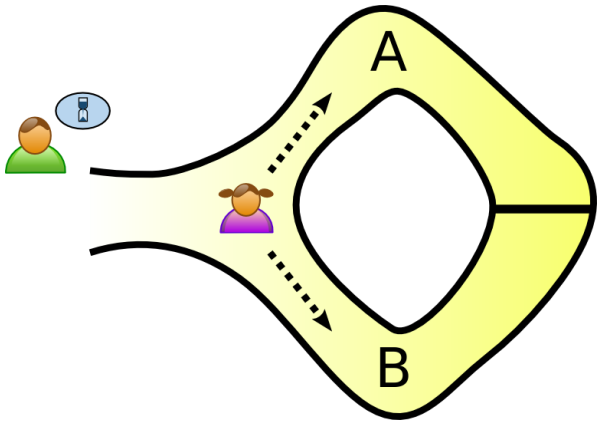
Zerocoin - Challenge

- How do people get their coins back without leaving their fingerprints all over it

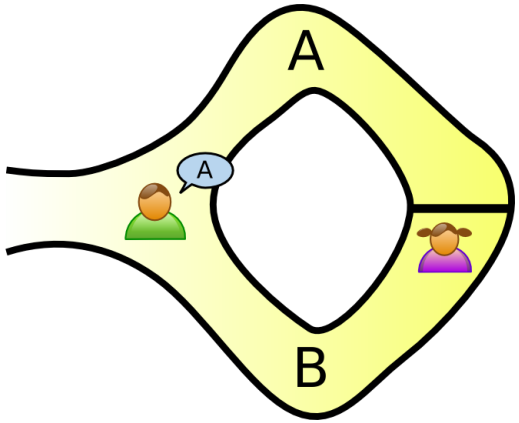
- **Solution:** zero-knowledge proof
  - A party can prove to another that a given statement is true, without conveying any additional information apart from the fact that the statement is indeed true
- Zero-knowledge proof says two things:
  - I am an owner of a Zerocoin and I know a serial number that is inside of the coin I made
  - I actually paid for the Zerocoin

Reminder: Zero-Knowledge Proofs

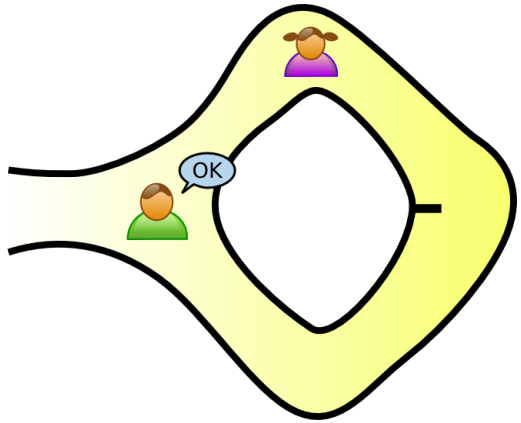
- Peggy has uncovered the secret word used to open a magic door in a cave
- Victor wants to know whether Peggy knows the secret word; but Peggy, being a very private person, does not want to reveal the fact of her knowledge to the world in general



Peggy randomly takes either path A or B, while Victor waits outside



Victor chooses an exit path



Peggy reliably appears at the exit Victor names

Figure: Wikipedia

Why Not Another Anonymous E-Cash System?

- Other anonymous cash systems rely on distributing the work of anonymizing users amongst a set of parties
  - Or to a trusted central authority
- Works well if all parties are fully available but can be subject to "denial of service" attacks
- Zerocoin system can remain available even when many nodes are compromised

Zerocoin - Protocol

- Goal: build a crypto currency where your neighbors, friends and enemies can't see what you bought or for how much
- Zerocoin *mint* and *spend* transactions
- Routine day-to-day transactions must be conducted via Bitcoin
  - Due to performance and functionality drawbacks

Minting a Zerocoin

- Bob generates a random serial number S, and encrypts this into a coin C by use of second random number r
- Coin C is added to a cryptographic accumulator by miners
- The amount of the base currency (Bitcoin) equal in value to the denomination of the Zerocoin is added to a Zerocoin escrow pool

Redeeming (Spending) a Zerocoin

- Bob needs to prove two things via a zero-knowledge proof
  - He knows a coin C that belongs to the set of all other minted Zerocoins $(C1, C2, \ldots\ Cn)$, without revealing which coin it is
    - By use of a one-way accumulator that does not reveal the members of the set
  - He knows a number r, that along with the serial number S corresponds to a Zerocoin
- The proof and serial number S are posted as a Zerocoin spend transaction
- Miners verify the proof and that the serial number S has not been spent previously
- If verified, the transaction is posted to the blockchain
- The amount of the base currency equal to the Zerocoin denomination is transferred from the Zerocoin escrow pool

Zerocoin - Anonymity

- Anonymity in the transaction is assured because the minted coin C is not linked to the serial number S used to redeem the coin
  - Serial number S is only publicly revealed after a redeem operation
- Limitations:
  - Size of the anonymity set depends on the number of coins minted by honest users
  - Reveals number of minted and spent coins to all users

Zerocoin - Drawbacks

- Efficiency Issues:
  - Extra computation time required by the process (to be performed by the miners)
    - The verification time for a block increases
    - Authors show that that verification time for an entire block would not exceed five minutes
  - If the proofs were posted to the blockchain, the size of the blockchain dramatically increases
    - Authors stated the proofs can be stored outside of the blockchain
- Not adapted by Bitcoin
  - Above efficiency issues
  - Political reasons…

Zerocoin - Drawbacks

- Anonymity problems:
  - Reveals payments' destinations and amounts
- Functionality problems:
  - Does not support payments of exact values
  - Does not provide a functionality to divide coins
- Does not support direct transfer of Zerocoins between users

Zerocash

Ben-Sasson et al. 2014

- Functions on top of any ledger-based base currency, such as Bitcoin
- More efficient than Zerocoin
  - Zerocash transactions are less than 1KB and take less than 6ms to verify
  - Zerocoin transactions exceeds 45 kB and require 450 ms to verify
- Hides the amount of the payment and destination
- All transactions can be made in terms of Zerocoins
- Users can:
  - Convert from Bitcoins to Zerocoins
  - Send Zerocoins to other users
  - Split or merge Zerocoins they own

Zerocash – Protocol

- *Mint transactions* and *pour transactions*
- Zerocash does not use any zero-knowledge proof
- Leverages ***zero-knowledge Succinct Non-interactive ARguments of Knowledge*** (zk-SNARK) systems
  - Zero-knowledge proofs that are particularly short and easy to verify

Mint Transactions - Overview

- Certificates of deposit
- Allows a user to convert a specified number of Bitcoins into the same number of Zerocoins belonging to a specified Zerocash address
- Consists of a cryptographic commitment to a new coin, which specifies the **coin's value**, **owner address**, and (unique) serial number
- The commitment is based on the SHA-256 hash function, and hides both the coin's value and owner address

Zerocash - Definitions

- COMM: statistically-hiding non-interactive commitment scheme
  - E.g., a hash function
- $f := COMM_r(m)$
  - f is opened by revealing r and m
  - One can verify that $COMM_r(m)$ equals f
- Three pseudorandom functions for a seed x
  - $PRF_x^{addr}(.)$, $PRF_x^{sn}(.)$, and $PRF_x^{pk}(.)$
- Each user u generates an address key pair (a_{pk}, a_{sk})
  - Coins of u contain the value a_{pk} and can be spent only with knowledge of a_{sk}
  - A user can generate and use any number of address key pairs

Mint Transactions

- To mint a coin c of a desired value v:
- User u determines the coin's serial number sn
  - $sn := PRF_{a_{sk}}^{sn}(\rho)$
  - ρ is a secret value sampled by the user
- u commits to the tuple (a_{pk}, v, ρ) in two phases
  - u computes $k := COMM_r(a_{pk}||\rho)$ for a random r
  - u computes $cm := COMM_s(v||k)$ for a random s
  - Nested commitments
- Resulting coin $c := (a_{pk}, v, \rho, r, s, cm)$
- Resulting mint transaction $tx_{Mint} := (v, k, s, cm)$
  - tx_{Mint} is sent to the ledger
  - tx_{Mint} is appended to the ledger only if u has paid v BTC to a backing escrow pool
    - Assuming 1 BTC = 1 Zerocoin

Nested Commitments

- Public $tx_{Mint} := (v, k, s, cm)$
- Anyone can verify that cm in tx_{Mint} is a coin commitment of a coin of value v
  - Checking that $COMM_s(v||k) = cm$
- Noone can know the owner (a_{pk}) or serial number (derived from ρ)

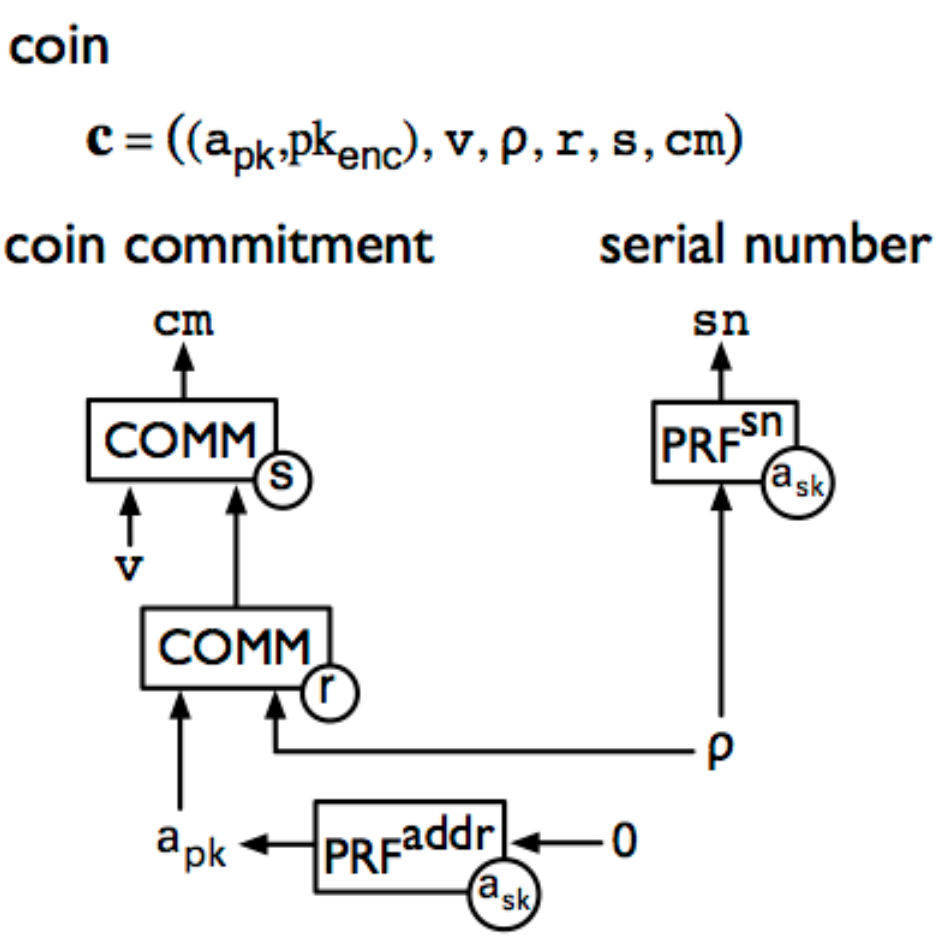


Figure: Ben-Sasson et al. Zerocash: Decentralized anonymous payments from Bitcoin.

Pour Transactions - Overview

- Allows a user to make a private payment
  - By consuming some number of coins to produce new coins
  - Takes a set of input coins, to be consumed, and "pours" their value into a set of fresh output coins
- Involves proving, in zero knowledge, that:
  - The user owns the input coins
  - Each one of the input coins appears in some previous mint transaction or as the output coin of some previous pour transaction
  - The total value of the input coins equals the total value of the output coins

Pour Transactions

- u with address key pair $(a_{pk}^{old}, a_{sk}^{old})$ wishes to consume his coin $c_{old} = (a_{pk}^{old}, v^{old}, \rho^{old}, r^{old}, s^{old}, cm^{old})$
- Operation produces two new coins c_1^{new} and c_2^{new}, with total value $v_1^{new} + v_2^{new} = v^{old}$
- New coins are targeted at address public keys $a_{pk,1}^{new}$ and $a_{pk,2}^{new}$
  - These may belong to u or other user
- For each coin 1 and 2 user u:
  - Samples serial number randomness ρ_i^{new}
  - Computes $k_i^{new} := COMM_{r_i^{new}}(a_{pk}^{new} || \rho_i^{new})$ for a random r_i^{new}
  - Computes $cm_i^{new} := COMM_{s_i^{new}}(v_i^{new} || k_i^{new})$ for a random s_i^{new}
- User u generates:
  - $c_1^{new} := (a_{pk,1}^{new}, v_1^{new}, \rho_1^{new}, r_1^{new}, s_1^{new}, cm_1^{new})$
  - $c_2^{new} := (a_{pk,2}^{new}, v_2^{new}, \rho_2^{new}, r_2^{new}, s_2^{new}, cm_2^{new})$

Pour Transactions – zk-SNARK

- u produces a **zk-SNARK** proof π_{POUR} for the following NP statement:

Given the Merkle-tree root rt, serial number sn^{old}, and coin commitments cm_1^{new}, cm_2^{new}, I know coins $c^{old}, c_1^{new}, c_2^{new}$, and address secret key a_{sk}^{old} such that:

- The coins are well-formed (commitments)
- The address secret key matches the public key a_{pk}^{old}
- The serial number sn^{old} is computed correctly
- The coin commitment cm^{old} appears as a leaf of a Merkle tree with root rt
- The values add up: $v_1^{new} + v_2^{new} = v^{old}$

Pour Transactions - Properties

- Pour transaction $tx_{Pour} := (rt, sn^{old}, cm_1^{new}, cm_2^{new}, \pi_{POUR})$ is appended to the public ledger

- If u does not know the address secret key $a_{sk,1}^{new}$ that is associated with the public key $a_{pk,1}^{new}$, u cannot spend c_1^{new}
- When a user that knows $a_{sk,1}^{new}$ does spend c_1^{new}, the user u cannot track it, because he doesn't know its revealed serial number
  - $sn_1^{new} := PRF_{a_{sk,1}^{new}}^{sn}(\rho_1^{new})$
  - u knows ρ_1^{new}, but not $a_{sk,1}^{new}$

- **Anonymity:** tx_{Pour} reveals no information about
  - How the value of the consumed coin was divided among the two new fresh coins
  - Which coin commitment corresponds to the consumed coin
  - The address public keys to which the two new fresh coins are targeted

Sending Coins

- Appending tx_{Pour} to the ledger is not enough
- Suppose $a_{pk,1}^{new}$ is the address public key of user u_1
- u must somehow send the secret values in c_1^{new} to u_1

- **Solution:** Assign a public/private key pair (pk_{enc}, sk_{enc}) to each user
- u computes $C_1 = Enc(v_1^{new}, \rho_1^{new}, r_1^{new}, s_1^{new})$ under $pk_{enc,1}^{new}$
- u includes C_1 in tx_{Pour}

Zerocash – Other Properties

- User u can redeem his coins (convert back to Bitcoin)
  - Via a public output in the pour operation
- Protocol is non-malleable
  - An attacker cannot modify the pour transaction
  - Using digital signatures

Discussion– Too Much Anonymity?

- **Concern:** decentralized anonymous payments will facilitate laundering of ill-gotten funds by criminals
  - Anonymity offered by Zerocash may facilitate illegal activity
- Arguments against the concern (by the authors):
  - Main difficulty with money laundering does not lie in how to privately transfer money from one person to another, but in how to make the eventual income appear legitimate
  - Even without the "help" of Zerocash, criminal users can already anonymize their activities via existing financial systems (e.g., by using cash)
- **A simple solution:** A backdoor could be added to allow police to track money laundering

References

- S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System, 2008
- E. Androulaki, G. Karame, M. Roeschlin, T. Scherer, and S. Capkun. Evaluating User Privacy in Bitcoin. In Proceedings of Financial Cryptography 2013.
- I. Miers, C. Garman, M. Green, and A. D. Rubin. Zerocoin: Anonymous Distributed E-Cash from Bitcoin. In Proceedings of the IEEE Symposium on Security and Privacy, 2013.
- Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M. Zerocash: Decentralized anonymous payments from Bitcoin. In Proc. of the 35th Symposium on Security and Privacy. S&P'14.