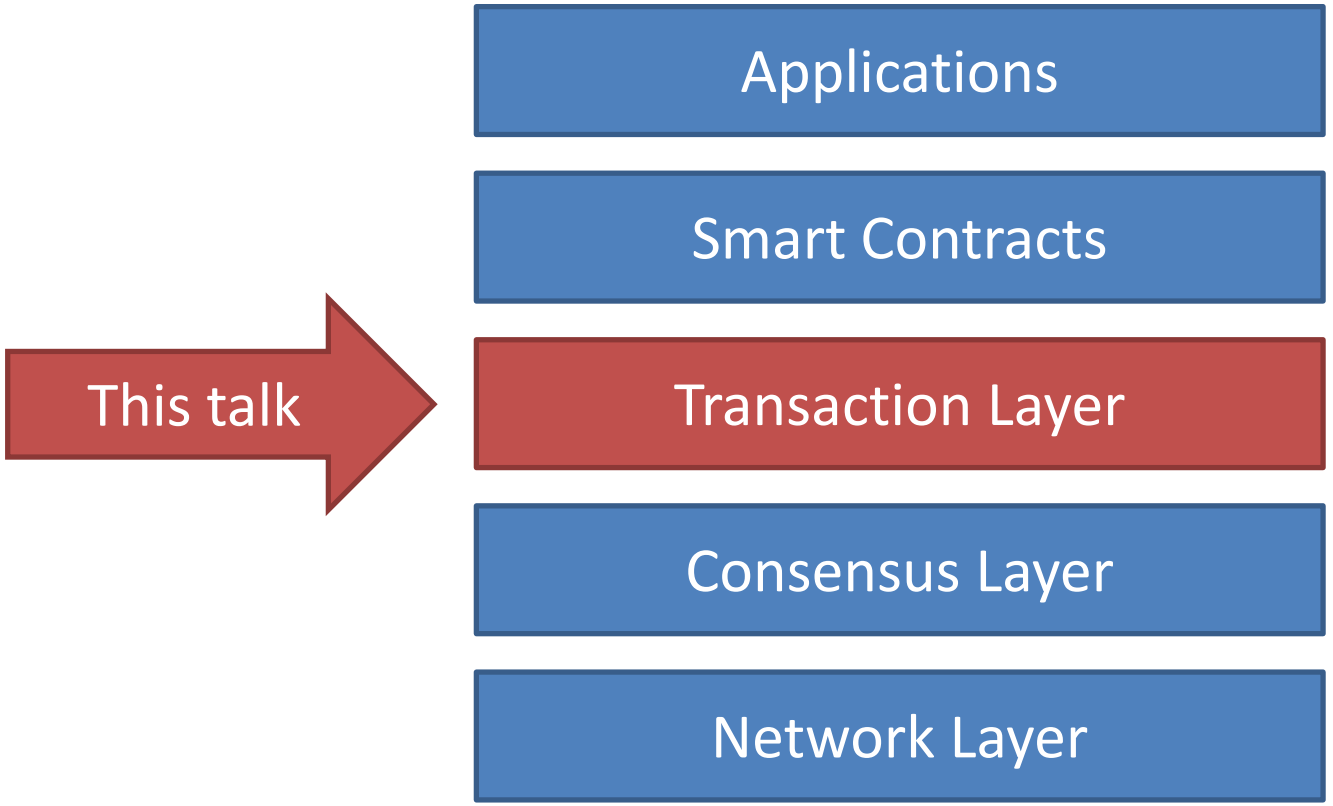
QuisQuis: A New Design for Anonymous Cryptocurrencies

Prastudy Fauzi, Aarhus University

Joint work with: Claudio Orlandi (AU),
Sarah Meiklejohn (UCL), Rebekah Mercer (AU)

*Original slides by Claudio Orlandi
(modified by Prastudy Fauzi)

Blockchain Research



Outline

- Bitcoin and Anonymity
- Anonymous Cryptocurrencies and Limitations
- Basic QuisQuis (1 public key = 1 coin)
  - Updatable public keys
  - N-to-N transactions without interaction
- Full QuisQuis (accounts with variable balance)
- Benchmarking & Conclusions

Outline

- Bitcoin and Anonymity
- Anonymous Cryptocurrencies and Limitations
- Basic QuisQuis (1 public key = 1 coin)
  - Updatable public keys
  - N-to-N transactions without interaction
- Full QuisQuis (accounts with variable balance)
- Benchmarking & Conclusions

“Bitcoin is like Twitter for your bank account.”
(Ian Miers)

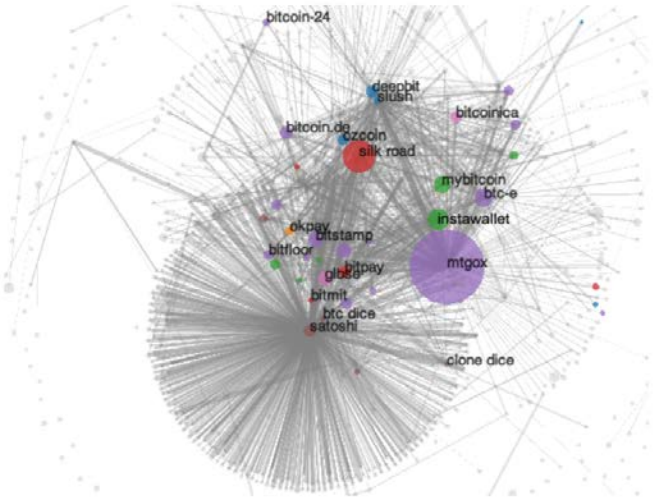


Figure 6: A visualization of the user network. The area of the cluster represents the external incoming value; i.e., the bitcoins received from other clusters but not itself, and for an edge to appear between two nodes there must have been at least 200 transactions between them. The nodes are colored by category: blue nodes are mining pools; orange are fixed-rate exchanges; green are wallets; red are vendors; purple are (bank) exchanges; brown are gambling; pink are investment schemes; and grey are uncategorized.

A Fistful of Bitcoins (Meiklejohn et al)

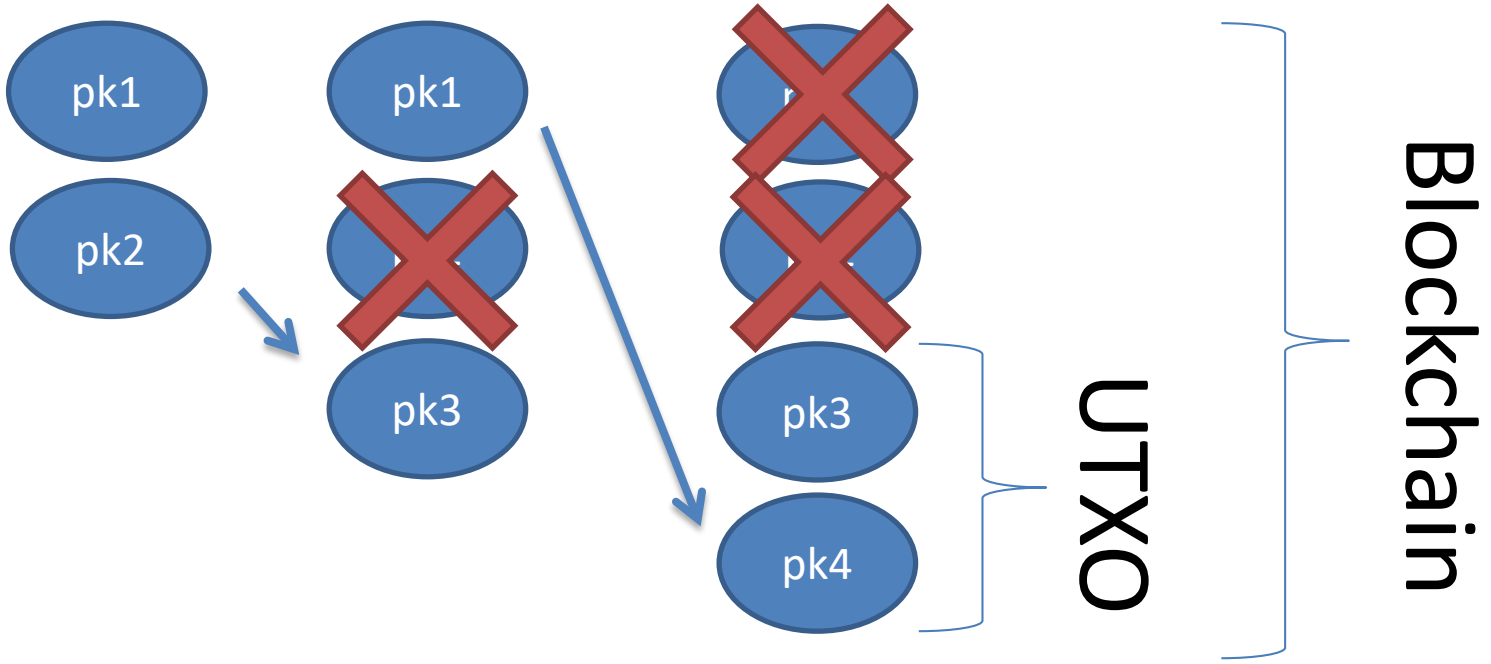
CRIME IN REAL TIME

The hackers behind the WannaCry ransomware attack have finally cashed out

By Keith Collins | August 03, 2017

CHAINALYSIS

Basic Transactions (e.g., Bitcoin)



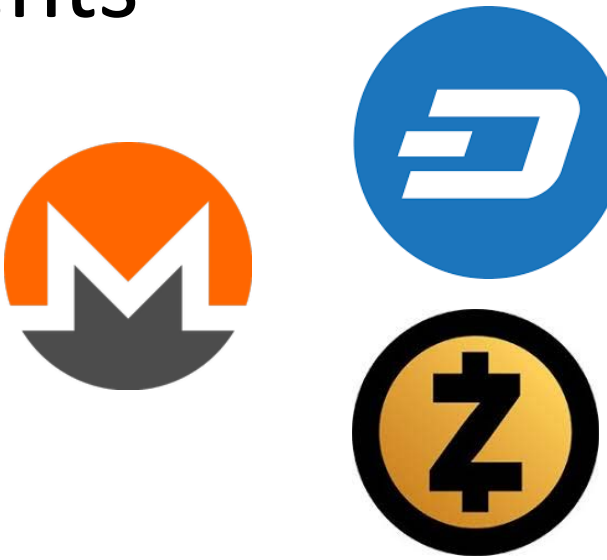
- For context, in January 2019
  - Blockchain 190+ GB
  - UTXO ~3 GB

Outline

- Bitcoin and Anonymity
- Anonymous Cryptocurrencies and Limitations
- Basic QuisQuis (1 public key = 1 coin)
  - Updatable public keys
  - N-to-N transactions without interaction
- Full QuisQuis (accounts with variable balance)
- Benchmarking & Conclusions

Existing Alternatives for Anonymous Payments

- Dash
- Monero
- Zcash



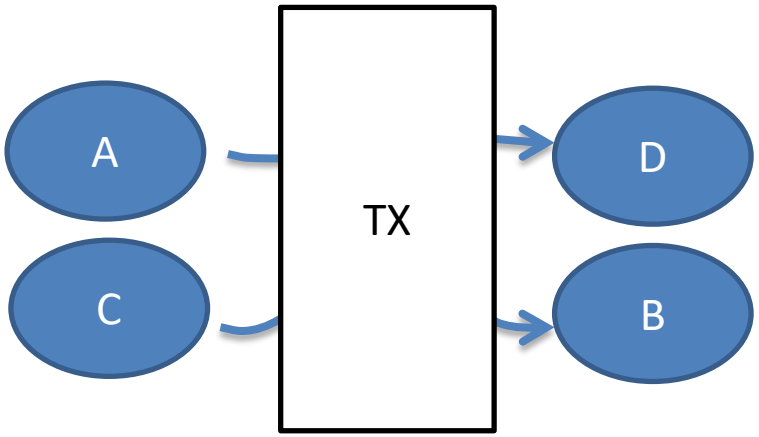
- … but, I’m a theoretician! For the rest of the talk I will address ”abstract technologies” not actual products (which are much more complicated).

Existing Techniques for Privacy

- Technologies
  - Tumblers 
  - Ring Signatures 
  - Zero-Knowledge/SNARKS 
- Questions
  - Need for coordination?
  - Plausible deniability?
  - Provable anonymity?
  - Trust in third parties?
  - Size of UTXO?

Tumblers (1/2)

- A wants to give 1 coin to B
- C wants to give 1 coin to D
- (A, C) create a 2-2 TX with receivers (B,D) in random order.
- An external observer cannot determine who sent to whom.
- Can be generalized to N senders and N receivers



Tumblers (2/2)

- Centralized Tumblers
  - ☺ Easy (trusted party performs transaction and matches users)
  - ☹ Need to trust central party for anonymity and security
- Decentralized Tumblers
  - ☹ Hard (how to find other users who want to mix their coins? + protocol require interaction)
  - ☺ Secure using cryptographic protocol
- (See e.g., TumbleBit by Heilman et al.)

Ring Signatures (1/3)

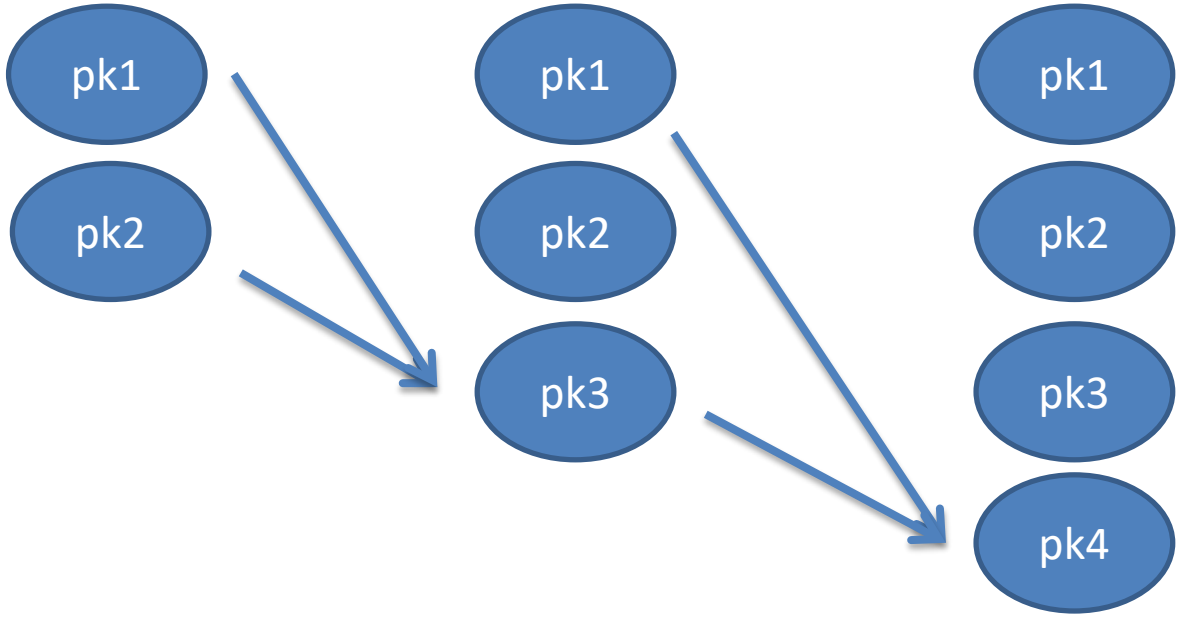
- Sign(pk_0,pk_1,sk_b, m)→ s
- Ver(pk_0,pk_1,m,s)→ accept

- Indistinguishability:
- Sign(pk_0,pk_1,sk_0, m) ≈ Sign(pk_0,pk_1,sk_1, m)

- (In general, there are N public keys)

Ring Signatures (2/3)

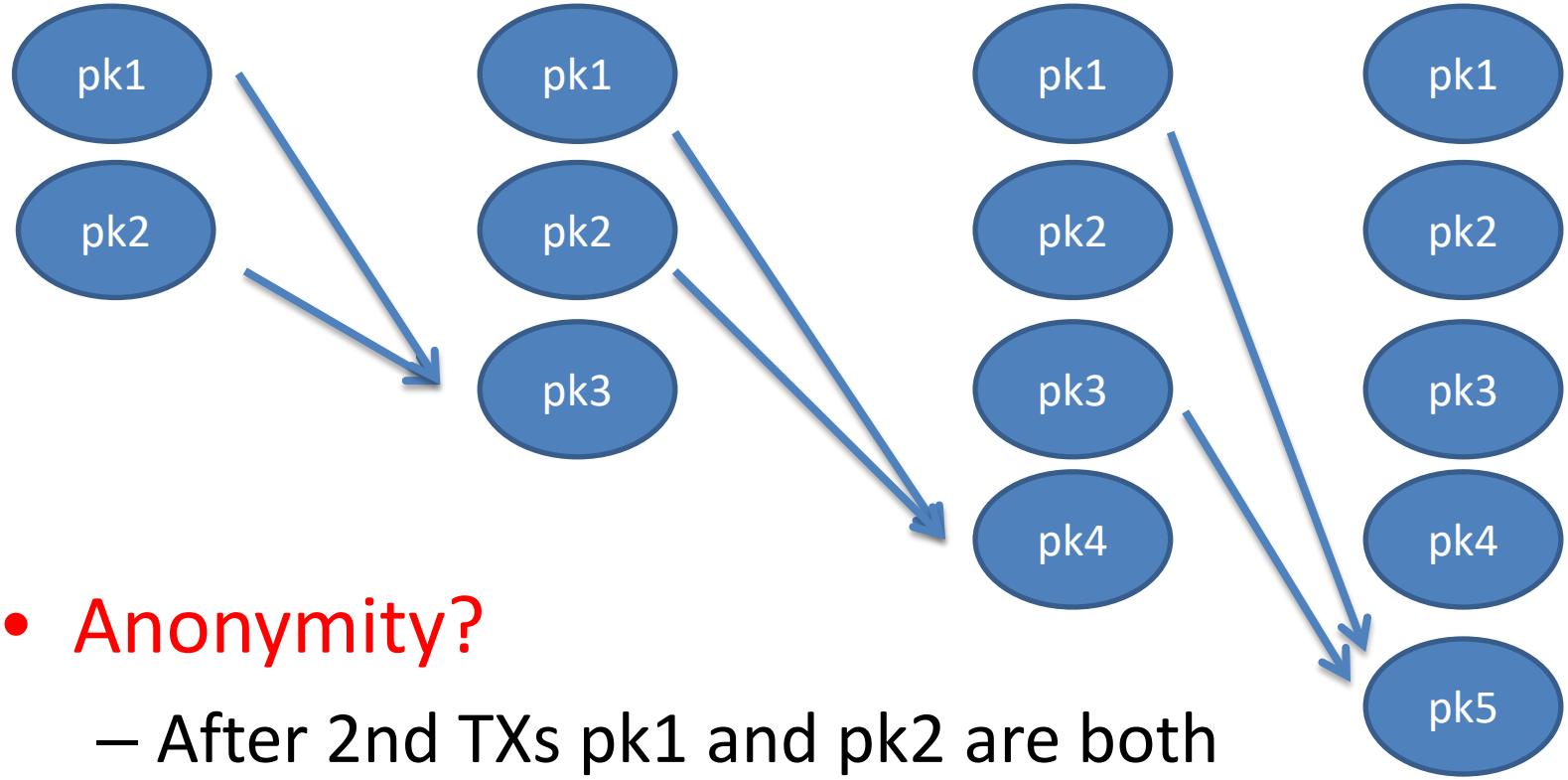
(Ignoring how to prevent double spending)



- Was pk1 spent? Can't tell! ☺
- Also means, cannot remove pk1 from UTXO ☹

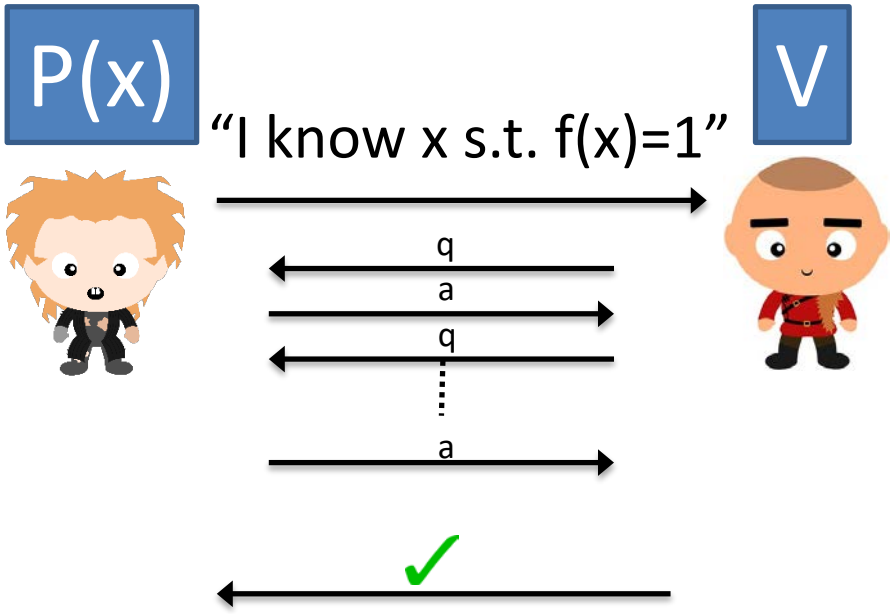
Ring Signatures (3/3)

(Ignoring how to prevent double spending)



- Anonymity?
  - After 2nd TXs pk1 and pk2 are both spent → 3rd transaction was made by pk3 with certainty

Zero-Knowledge (1/2)

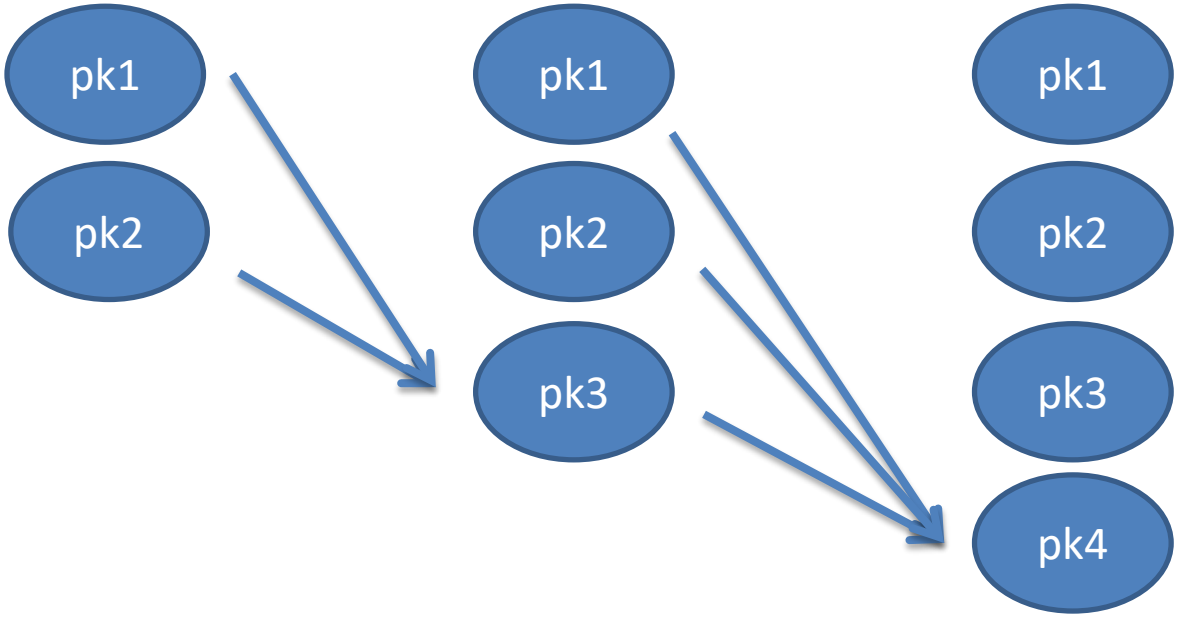


- **Completeness**
  - P,V honest → V accepts
- **Proof-of-Knowledge**
  - If P does not know x → V rejects
- **Zero-Knowledge**
  - V learns nothing about x

- **Non-Interactive:** proofs can be made non-interactive using CRS or the RO model.

Zero-Knowledge (2/2)

- Can be seen as extension of ring signatures, using advanced cryptographic protocols (SNARKS)
  - Can hide in sets of arbitrary size - “∞-to-1” transactions
  - Generation time for transaction high ☹
  - Need for trusted setup (CRS) ☹



	Security			Efficiency				
	Anonymity	Deniability	Theft prev.	UTXO growth	tx size		tx cost (ms)	
					big-$\mathcal{O}$	kB	prover	verifier
Tumblers	yes*	no	yes*	non-monotonic	low - high		slow	
Zcash	yes	no*	yes	monotonic	1	0.29	23,768	8.36
Monero	no	yes	yes	monotonic	$n + v$	14	238	46.4

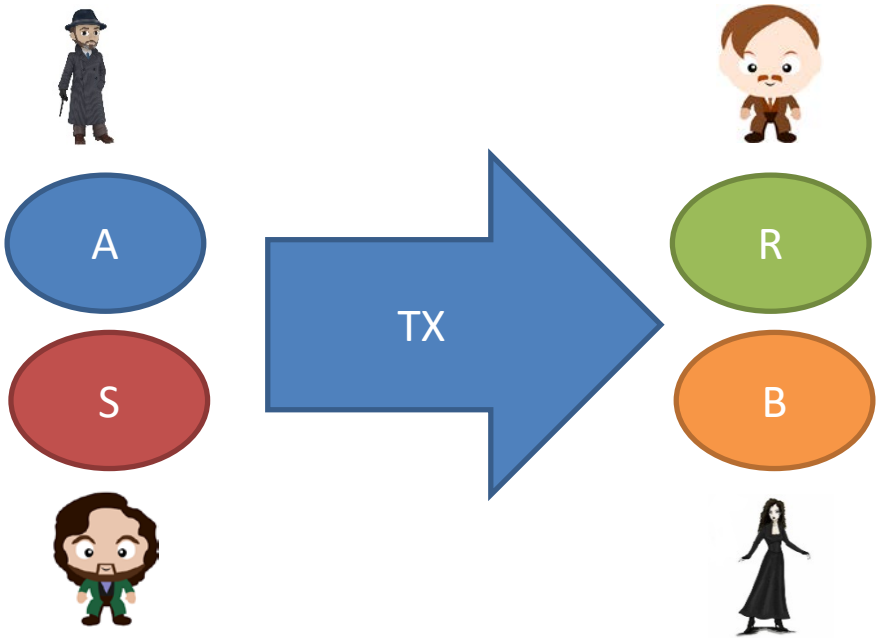
- Numbers taken from benchmarking in October 2018
- Machine: Core i7 server, 3.5 GHz CPU, 32 GB RAM

Outline

- Bitcoin and Anonymity
- Anonymous Cryptocurrencies and Limitations
- Basic QuisQuis (1 public key = 1 coin)
  - Updatable public keys
  - N-to-N transactions without interaction
- Full QuisQuis (accounts with variable balance)
- Benchmarking & Conclusions

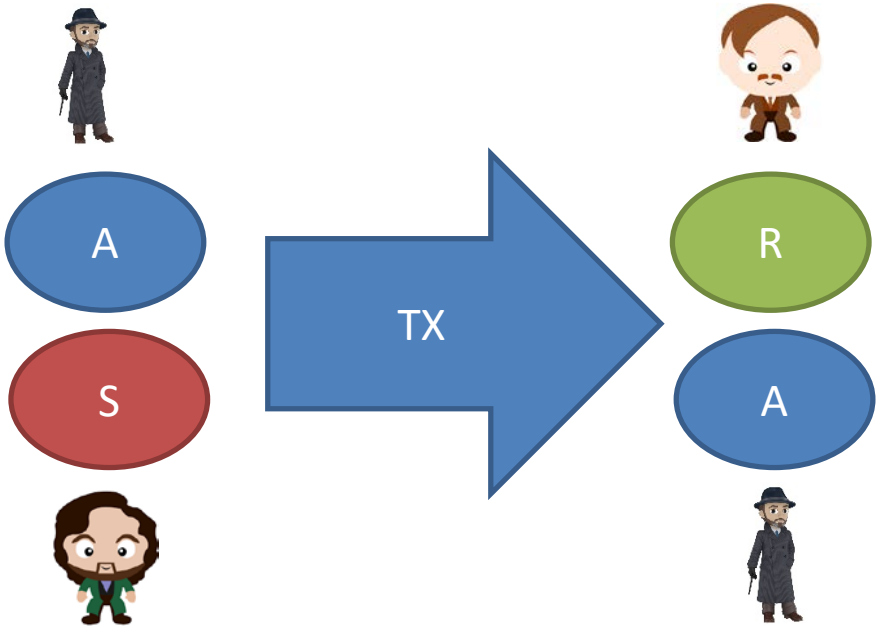
Basic QuisQuis idea

N-to-N transaction without interaction



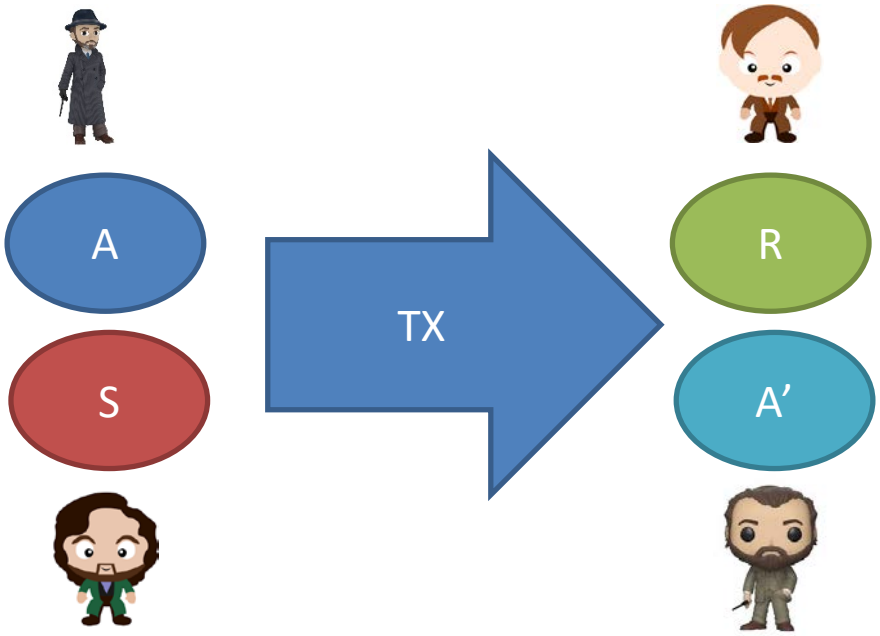
- S wants to send money to R
- Add transaction from A to B for anonymity
- Paradox?
  - Move other people money without their approval
  - While at the same time preventing theft?

Idea that does not work



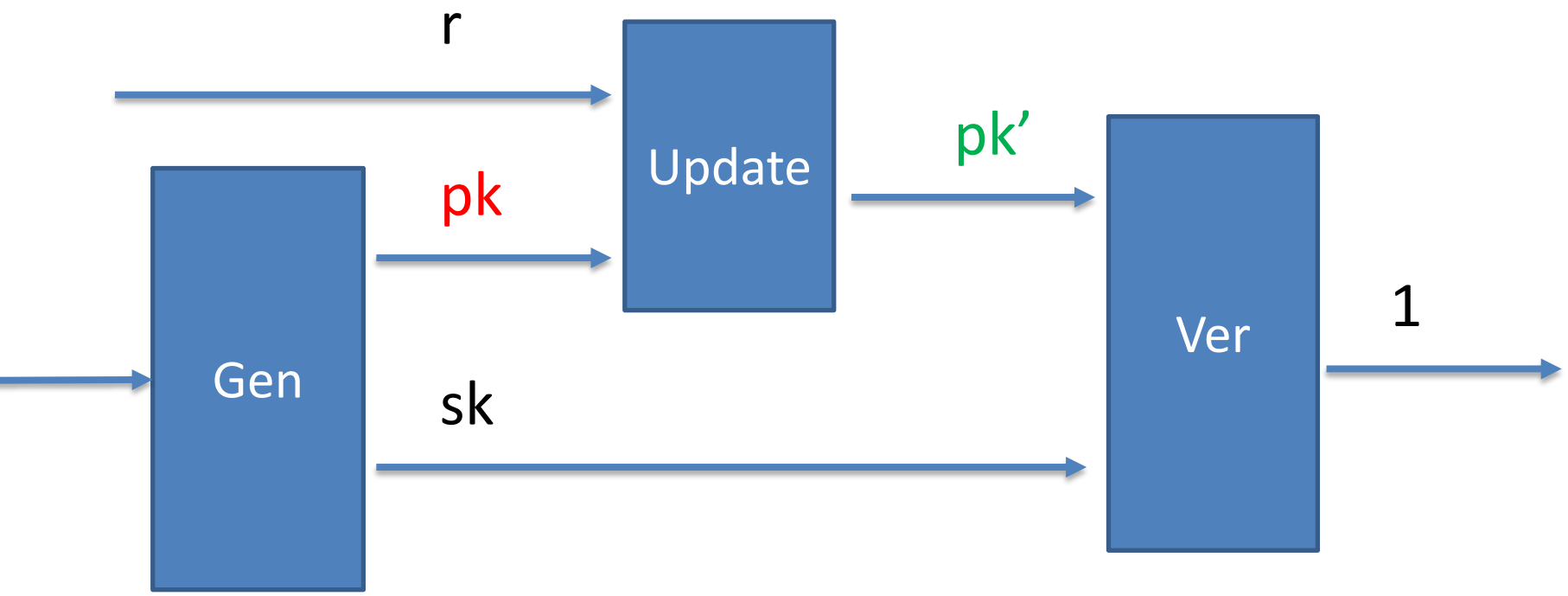
- Add transaction from A to A.
- No money stolen ☺
- No privacy ☹

Idea that might work



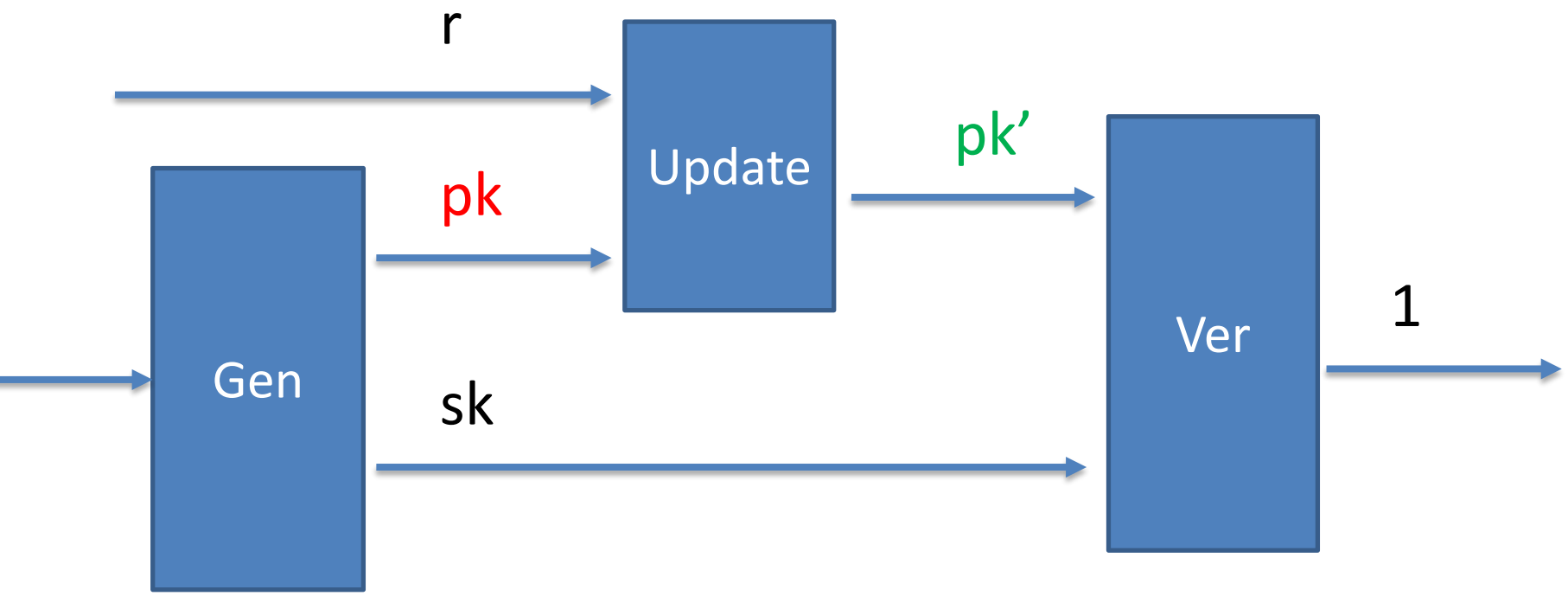
- What if I could move A’s money to a new ”random looking” address **which is also owned** by A?

Updatable Public Keys



- **Correctness**: (pk',sk) is a valid key pair

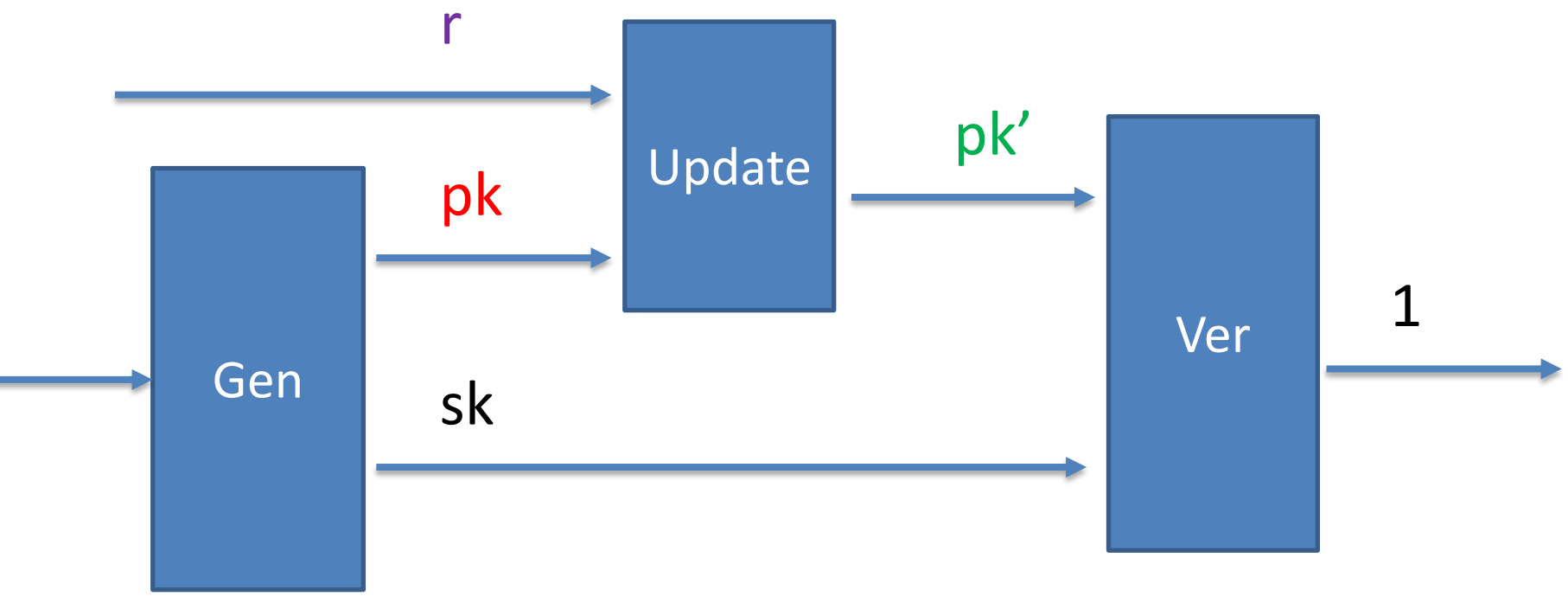
Updatable Public Keys



- **Indistinguishability**:

  pk and pk' are computationally indist.

Updatable Public Keys



- **Unforgeability**:

  Given pk, can't learn *sk of updated public key*
  pk' = ***Update***(*pk*, r)

Unforgeability

- No A(pk) can output (pk',sk,r) such that

  Update(*pk*,*r*) → *pk'*

  AND

  (pk',sk) is a valid pair

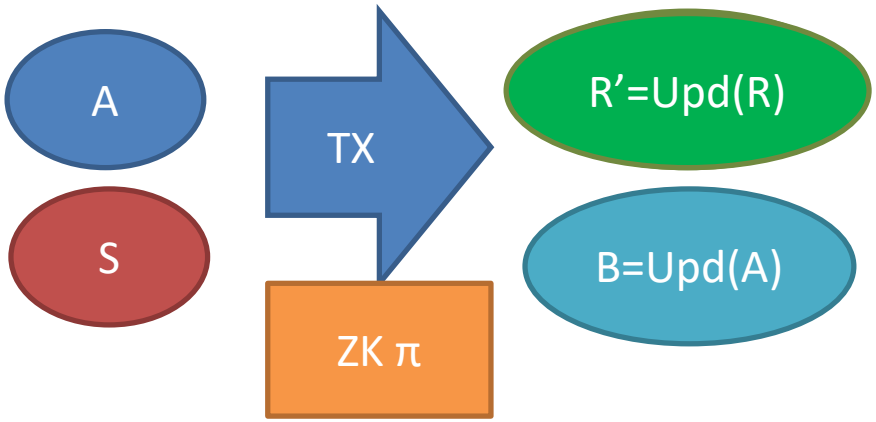
- **Idea**:
  - Output (pk',r): *trivial!* (run *Update*)
  - Output (pk',sk): *trivial!* (drop pk and run Gen)
  - Both at the same time should be hard!

Constructions of Updatable Public Key

- Gen → $pk=(g^s,g^{sx})=(u,v)$, $sk=x$
- Update(pk, r) → $pk' = (u^r,v^r)$

- **Correctness**: ✓
- **Indist.**: $(u,v, u^r,v^r) \sim (u,v,u^r,v^s)$ (assuming DDH)
- **Unforgeability**: output x = break DL

Basic QuisQuis Transaction



- Real Input: pk_S
- Real Output: pk_R
- Run **Update**(pk_R)→pk_R'
- Pick random pk_A from UTXO
- Run **Update**(pk_A)→ pk_B

- ZK proof π for the following statement:
  - ”N-1 public keys were updated correctly (hiding which ones)”
  - “I know the sk corresponding to the last public key (and I can therefore spend it)”

Outline

- Bitcoin and Anonymity
- Anonymous Cryptocurrencies and Limitations
- Basic QuisQuis (1 public key = 1 coin)
  - Updatable public keys
  - N-to-N transactions without interaction
- Full QuisQuis (accounts with variable balance)
- Benchmarking & Conclusions

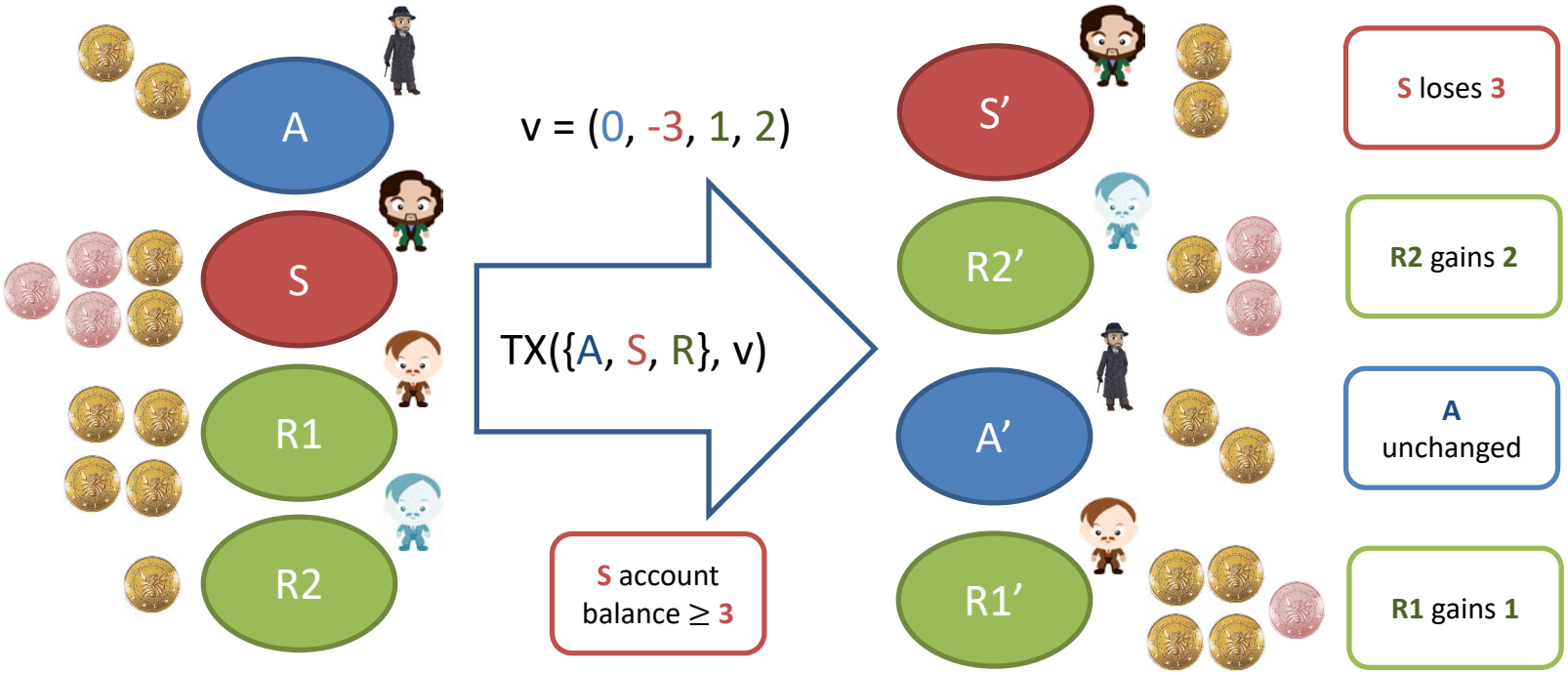
Towards the full construction…

- In the basic construction, we assumed every PK held exactly 1 coin
  - Unrealistic assumption
- We want to deal with variable amounts associated to keys
  - While hiding the amounts
  - Allowing anonymous transactions

QuisQuis: ideas

- Each public key has associated a *balance* bl
  - The balance is stored in committed (”encrypted”) format.
  - The commitment is homomorphic, the balance can be modified.
- Transactions are now ”redistribution of value”

QuisQuis: redistribution of value



Accounts

- Accounts are pairs of public keys and commitments to the balance of the public key

- E.g., acct = (pk, Com(pk,bl))

- In QuisQuis: $pk = (u,v)$
  $Com(pk,bl) = (u^r, g^{bl} v^r)$

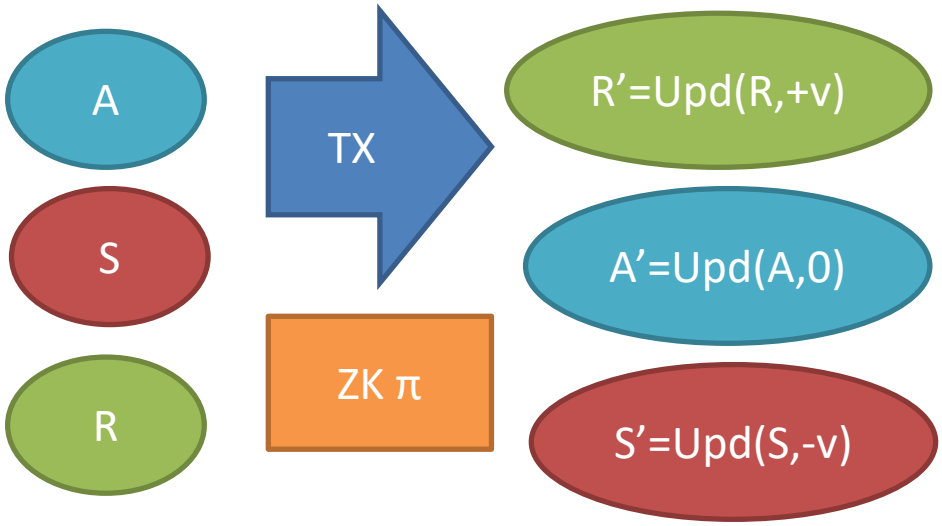
Accounts can be updated too

- Update(acct=(pk,c),v)
  - pk’ = Update(pk)
  - c’ = c * Com(pk,v)
  - Output acct’ = (pk’,c’)

- **Note:**
  - the secret key did not change
  - The value can be increased/decreased by v
  - acct’ cannot be linked to acct!

QuisQuis Transactions

”Redistribution of Value”



- Real Sender: $acct_S$ (loses v)
- Real Reciever: $acct_R$ (gains v)
- Pick random $acct_A$ from UTXO
- Run **Update**($acct_R$,+v)→$acct_R'$
- Run **Update**($acct_S$,-v)→$acct_S'$
- Run **Update**($acct_A$,0)→$acct_A'$

- Construct ZK π that everything was done correctly e.g.,
  - All accounts were updated correctly and with amounts ≥ 0.
  - Except one, for which I knew the sk, and whose balance ≥ v.

QuisQuis Transaction Properties

- **Non-growing UTXO:**
  - Only the last version of the accounts is stored
- **Theft prevention:**
  - You can withdraw from your account (e.g., if you know sk), as long as the balance is positive
  - The other accounts only receive non-negative updates
- **Anonymity**
  - Updated accounts in the output are unlinkable to the accounts in the input set
  - The commitments hide the value
  - The ZK proof hides the relationship between inputs/outputs, and the value which was transferred

Outline

- Bitcoin and Anonymity
- Anonymous Cryptocurrencies and Limitations
- Basic QuisQuis (1 public key = 1 coin)
  - Updatable public keys
  - N-to-N transactions without interaction
- Full QuisQuis (accounts with variable balance)
- Benchmarking & Conclusions

Performances

- See the paper for details on ZK proofs:
  - Combination of Sigma protocols for DL relations
  - Bayer-Groth Shuffle
  - Bulletproof (range proofs)
- Implemented in Go

$\|A\|$	Gen. (ms)	Verif. (ms)	Proof size	Proof size (bytes)
4	124±4%	25.6±3%	$122\mathbb{G} + 83\mathbb{F}_p$	6528
16	471±4%	71.6±3%	$244\mathbb{G} + 175\mathbb{F}_p$	13,408
64	2110±3%	251 ±4%	$624\mathbb{G} + 503\mathbb{F}_p$	36,064

Comparison

	Security			Efficiency				
	Anonymity	Deniability	Theft prev.	UTXO growth	tx size		tx cost (ms)	
					big-O	kB	prover	verifier
Tumblers	yes*	no	yes*	non-monotonic	low - high		slow	
Zcash	yes	no*	yes	monotonic	1	0.29	21,747	8.57
Monero	no	yes	yes	monotonic	$n + \log(v)$	2.71	982	46.3
Quisquis	yes	yes	yes	non-monotonic	$n + \log(v)$	13.4	471	71.6

- Monero: 2 new outputs (TXOs), ring size 10
- QuisQuis: 1 S, 3 R, 12 A, set size 16
- Monero's values updated: they now also use Bulletproofs, taking n + v to n + log(v)

Conclusions

- Quisquis shows an alternative approach to designing anonymous cryptocurrencies
- Open problems
  - Empirical analysis of anonymity
  - Theoretical and practical optimizations
  - Other applications for updatable public keys and Quisquis design principle?

Thank you!

Available at: eprint.iacr.org/2018/990