Schnorr Signature
&
MimbleWimble

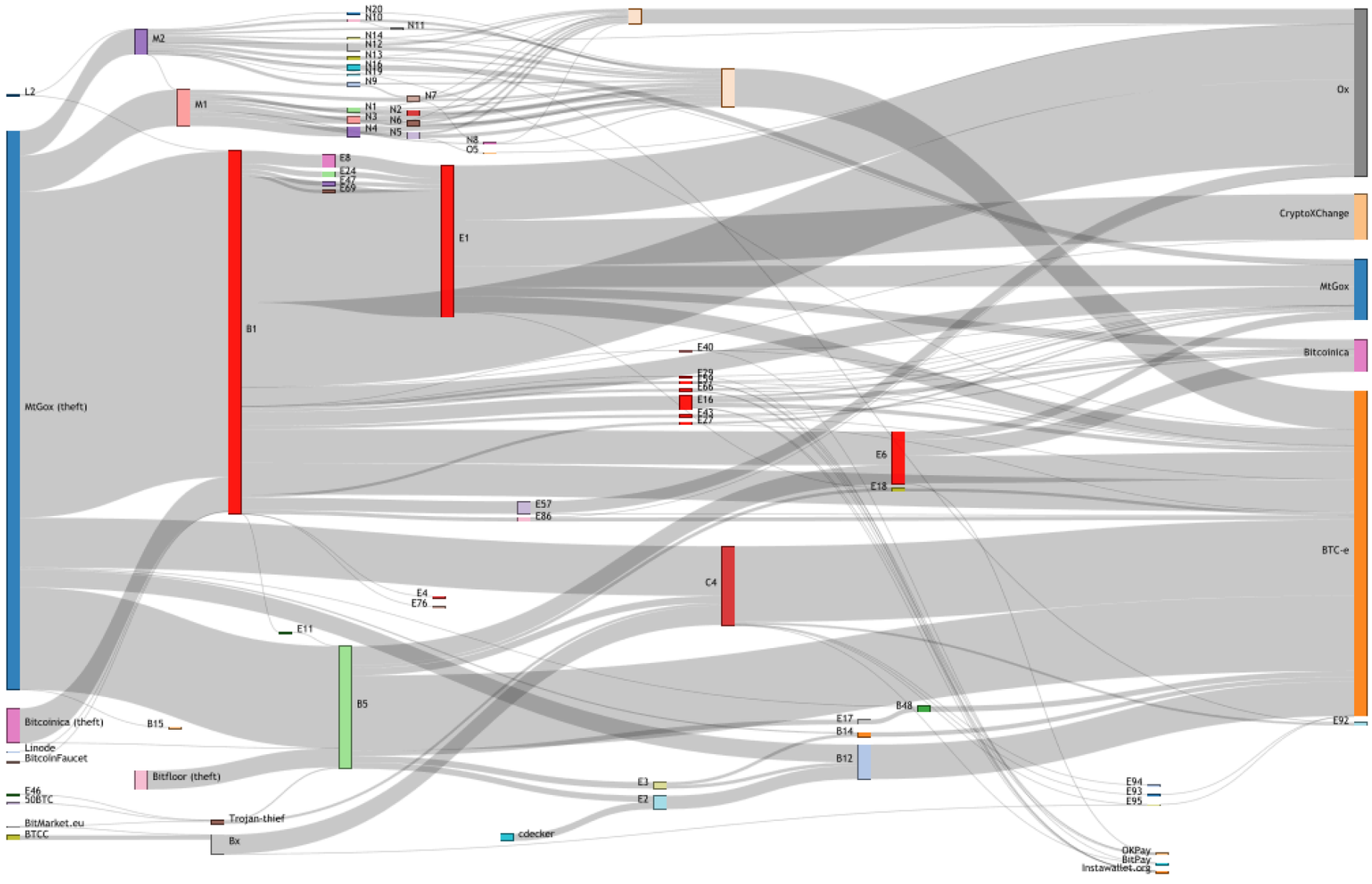
Oct. 5, 2019

Overview of today

- Lack of Privacy in Bitcoin
- MimbleWimble cryptocurrency
- ECC math
  - Schnorr’s signatures scheme
  - Pedersen Commitments

Motivation

- Bitcoin is decentralized and anonymous, but not private
  - Everybody can see the amounts transferred
  - We can trace payments and money



MimbleWimble

- MimbleWimble, a Tongue-Tying Spell from Harry Potter
  - The protocol is unable to spill details about a transaction

- MW is build on ECC
  - Pedersen Commitments
    - to hide amounts
  - Schnorr signatures (as opposed to ECDSA)
    - To prove that transactions are correct

Signatures

- A signature proofs that the owner of a private key created some input-dependent data
  - $s = \text{sign}(sk, \text{document})$
- Everybody can verify this using the public key
  - $\text{verify}(pk, \text{document}, s) \in \{\text{True}, \text{False}\}$

Signatures with ECC

- For elliptic curve cryptography, (at least) 3 types of signatures exist
  - ECDSA
  - Schnorr - signature
  - BLS (Boneh–Lynn–Shacham)

- Schnorr’s signature are easier to understand and implement correctly than ECDSA

- Schnorr’s signatures are extendable
  - A property we will use today

Timeline of ECC-based Signature

1985 1990 1995 2000 2005 2010 2015 2020

ECC proposed

C.-P. Schnorr patents a signature scheme on ECC

S. Nakamoto develops Bitcoin (ECDSA)

NIST develops ECDSA signatures to circumvent Schnorr's patent

patent

ECC becomes popular

Schnorr's sig. patent expires

Now

ECDSA

- Widely used and researched
    - Malleable, i.e. an attacker can change the document and the signature without knowing the private key
        - several extra checks have to be performed to prevent attacks

- In short:
    - On the way out. Not a focus of this course

Schnorr’s signature

- Relatively new (popularized only recently)
- Non-malleable
- Concepts, security proofs, and implementation easier
- In short:
  - Schnorr’s signatures are often better than ECDSA

BLS signature

- Relatively new (popularized only recently)
- Require special elliptic curves
- Believed to be secure

- In short:
  - Potentially very useful for complex applications
  - Security proofs and trusted implementations not yet widely accepted

Elliptic point math
(Recap)

- Capital letters: points on a curve $P, Q, R, \ldots \in \mathbf{Z}_p \times \mathbf{Z}_p$
- lower case letters: integers $a, b, c \ldots \in \mathbf{Z}$
- Points can be added
  - P+Q, P+P+P+Q+Q+Q
- Points can be multiplied with a numbers
  - aG, (b+c)P
- Commutative and associative rules are preserved
  - $a\left((b+c)P + d(e+f)G\right) = adeG + adfG + abP + acP$

Schnorr's signature

- Global parameters:
  - Base-point G publicly known
  - non-invertible hash function $\mathscr{H}$
- User specific parameters
  - Private key: integer p
  - Public key: Point $P = pG$
- One-time parameters
  - document to sign
  - random (secret) number r
  - Public point $R = rG$

Schnorr's signature

- sign(document, private key = p)
  1. generate random number r
  2. compute $s = r + \mathscr{H}\left(R \,|\, P \,|\, \text{document}\right) p$
  3. return (s, rG)

$P = pG$ public key

$rG = R$ random one-time nonce

- verify(document, signature = (s,R), public key=P):

$$sG \stackrel{?}{=} R + \mathscr{H}\left(R \,|\, P \,|\, \text{document}\right) P$$

$$= rG + \mathscr{H}\left(R \,|\, P \,|\, \text{document}\right) pG$$

$$= \left(r + \mathscr{H}\left(R \,|\, P \,|\, \text{document}\right) p\right) G$$

Schnorr’s multi-signature

- We can easily extend this scheme to multi-signatures
  - We can prove that a group of people all signed it

- Alice (private/public key a / aG), random secret point n
- Bob (private/public key b / bG), random secret point m

- multisig: (s, nG+mG)

- $s = n + m + \mathscr{H}\left(nG + mG \,|\, aG + bG \,|\, \text{document}\right)(a + b)$

Schnorr's multi-signature
communication protocol

- Goal, compute
  - $s = n + m + \mathscr{H}\left(nG + mG \,|\, aG + bG \,|\, \text{document}\right)(a + b)$
  - without revealing secrets a, b, n, m to other party

- Alice: $s_a = n + \mathscr{H}\left(nG + mG \,|\, aG + bG \,|\, \text{document}\right) a$
- Bob: $s_b = m + \mathscr{H}\left(nG + mG \,|\, aG + bG \,|\, \text{document}\right) b$
- Multi-sig:

$$\begin{aligned} s_a + s_b &= n + m + \mathscr{H}(\ldots)b + \mathscr{H}(\ldots)a \\ &= n + m + \mathscr{H}(\ldots)\left(a + b\right) \end{aligned}$$

Summary Schnorr's signatures

- A number and a point
  - $(r + \mathscr{H}(rG \,|\, pG \,|\, \text{text})p$, rG)
- Easy to compute and to verify
- Linear, i.e. we can aggregate signatures into one
  - e.g. 200 aggregated signatures are still only one number and one point

Pedersen Commitments

- Instead of one base point, we use 2: G,H
  - secret value s
  - use random value γ
    - Let's use Greek letters for random values
    - Also called blinding factors
- Pedersen commitment of s is $sG + \gamma H$

T. P. Pedersen

Note: There is a value z so that $H = zG$

It is important that no one knows this value

Pedersen Commitments

$$X = rG + \gamma H$$

- Impossible to separate X into the part generated by G and the part generated by H

Proving properties of Pedersen Commitments

$$X = rG + \gamma H$$

- Alice can prove $r = 0$ by using X in a signature

- Alice sends

$$s = m + \mathscr{H}\left(X \,|\, M \,|\, \text{"Alice"}\right) \gamma \text{ and } mH$$

- Bob verifies

$$sH \stackrel{?}{=} mH + \mathscr{H}\left(X \,|\, M \,|\, \text{"Alice"}\right) X$$

$$sH \stackrel{?}{=} mH + yX \qquad \text{with } y = \mathscr{H}(\ldots)$$

Proving that one part is 0

- Bob knows that
  - $\left(mH + yX\right)$ is a term only generated by H
    - since $sH = mH + yX$
  - Alice does not know z with $H = zG$

- Therefore, X does not have any G

Summary Pedersen Commitments

- We use 2 base points (G,H)
  - Also called generators

- We can commit value r using blinding factor γ
  - $X = rG + \gamma H$

- We can prove that $r = 0$ without revealing γ
  - Using Schorr's signature scheme

MimbleWimble

(A cryptocurrency protocol)

2 implementations: Beam and Grin

Cryptocurrency with Pedersen Commitments

$$X = rG + \gamma H$$

Amount, e.g. $r = 12C$

Secret Key, e.g. $\gamma = 623475638542338746 5$

Transactions in MW

- Alice has r=12 coins in $A = rG + \gamma H$
- Alice wants to send 4 coins to Bob
- Alice and Bob publish equation

$$A - B - C$$

$$\underbrace{(12G + \gamma H)}_{\text{Alice's input}} - \underbrace{\left(4G + \beta H\right)}_{\text{Bob's output}} - \underbrace{(8G + \alpha H)}_{\text{Alice's return}}$$

- Only Alice knows γ, α
- Only Bob knows β
- The blockchain removes A from the UTXO and adds B,C

Transactions in MW

- Verify correctness of a transaction:
  - given eq. $(xG + \gamma H) - \left(yG + \beta H\right) - (zG + \alpha H)$
  - A transaction is valid, if inputs = outputs
    - All coefficients of G sum up to 0, i.e. $x - y - z = 0$
- $(xG + \gamma H) - \left(yG + \beta H\right) - (zG + \alpha H) = (\gamma - \beta - \alpha)H$

Transactions in MW

- A transaction $T = A - B - C$ is
  - a point on the curve
  - a Pedersen commitment
- We can use Schnorr’s signature to proof that T is only made out of H components

Transactions in MW

- A transaction $T = A - B - C$ is
  - a point on the curve
  - a Pedersen commitment
- We can use Schnorr's signature to proof that T is only made out of H components
  - provide a point M and value s, so that

$$sH = M + \mathscr{H}\left(M \,|\, T \,|\, \text{sometext}\right) T$$

could be the empty string

Transactions in MW

- There is one more hole to plug. Consider transaction

$$T = (12G + \gamma H) - \left(400G + \beta H\right) - (-388G + \alpha H)$$

Input Alice **Output Bob** **Return Alice**

- This is valid
  - 12-400-(-388) = 0
  - Bob now has 400 coins
- This is a problem
- We need to prove that each Pedersen Commitment is >0
  - Range proofs are outside the scope of today's lecture
  - Bulletproofs (later)

Transactions in MW

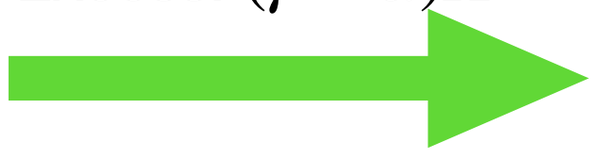
- We now have all the ingredients for a MW transaction.
  - Alice owns a known Pedersen commitment A
  - Alice and Bob interact (off chain) to produce
    - $T = A - B - C$
    - (s, M), so that $sH = M + \mathscr{H}(\dots)T$
    - A range proof coefficient of G in B > 0
    - A range proof coefficient of G in C > 0

Creating Transactions

Alice (owns $A = 12G + \gamma H$)

- $C = 8G + \alpha H$
- random $M = mH$
- Range proof for C

Alice In: A
Alice out: C
Amount: 4
Random nonce: M
Excess: $(\gamma - \alpha)H$



Bob

- $B = 4G + \beta H$
- random $N = nH$
- $T = A - B - C$
- $h = \mathscr{H}\left(M + N \,|\, T \,|\, ""\right)$
- $s_b = n + h(-\beta)$
- Range proof for B

Bob Out: B
Range proof: r(B)
partial sig: s_b
Random nonce: N

- $T = A - B - C$
- $h = \mathscr{H}\left(M + N \,|\, T \,|\, ""\right)$
- $s_a = m + h(\gamma - \alpha)$

Alice publishes $A, B, C,\ (s_a + s_b, M + N),\ r(B),\ r(C)$

Summary of MW Transactions

- All values are hidden in Pedersen Commitments
- Transactions given as an equation

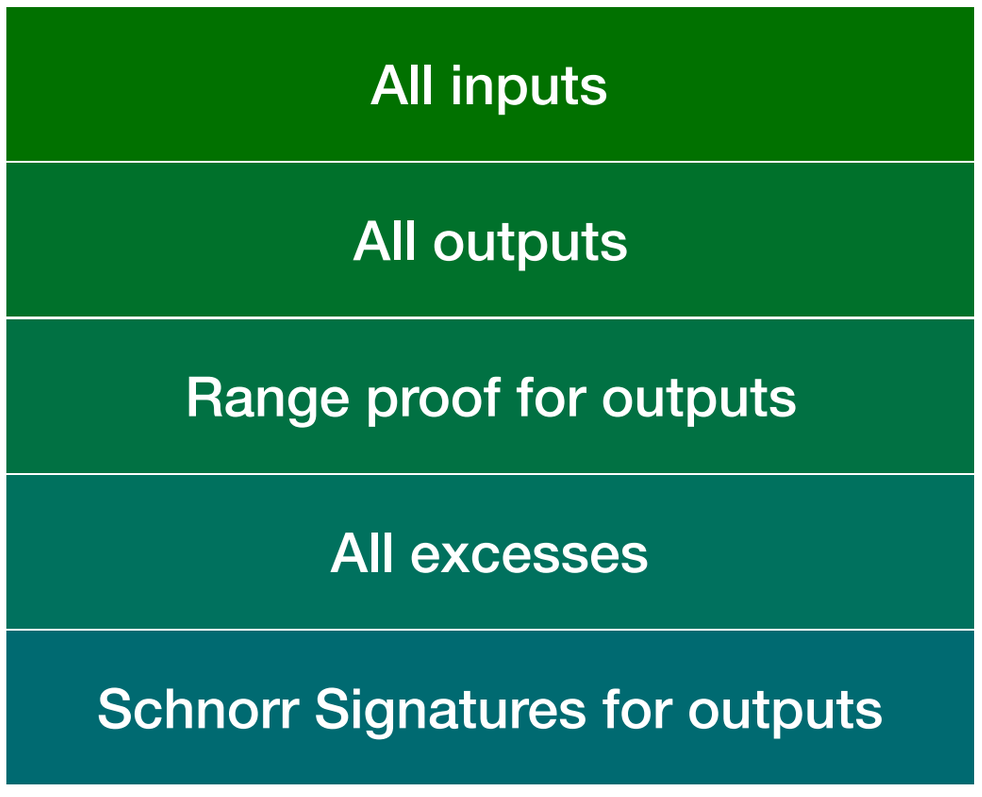
$$Inputs - Output = Excess$$

- To prove that transactions are valid
  - use Schnorr's signature scheme to show that the $Excess$ can be expressed by only using generator H
  - use range proofs to show that all amounts >0

Extra Security

- The MW miner sees all these values

- An honest miner can obfuscate the block by disassociating inputs and outputs



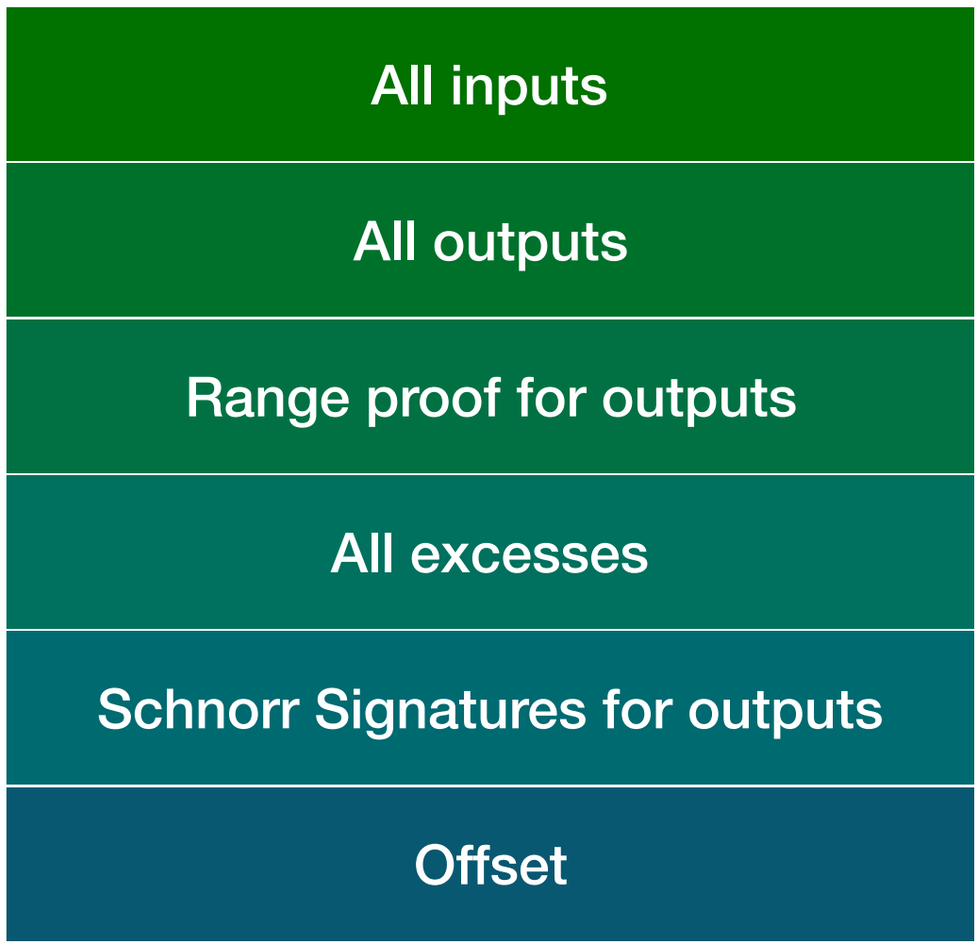
Verify via
sum(inputs)-sum(outputs) = sum(excess)

Kernel Offset for extra obfuscation

- An attacker can easily match inputs with outputs

- Thus, add an arbitrary value to each tx

  In1 - Out1 = Excess1 + offset1
  In2 - Out2 = Excess2 + offset2
  …



Efficiency Gains

- If
  - A sends money to B
  - B sends money to C
- we don’t need to store any information about B

Cut Through

- Given 2 transactions
  - $T_1 = A + B + C - D - E$
    - $T_2 = D + G + H - J - K$

output used directly as input for another tx

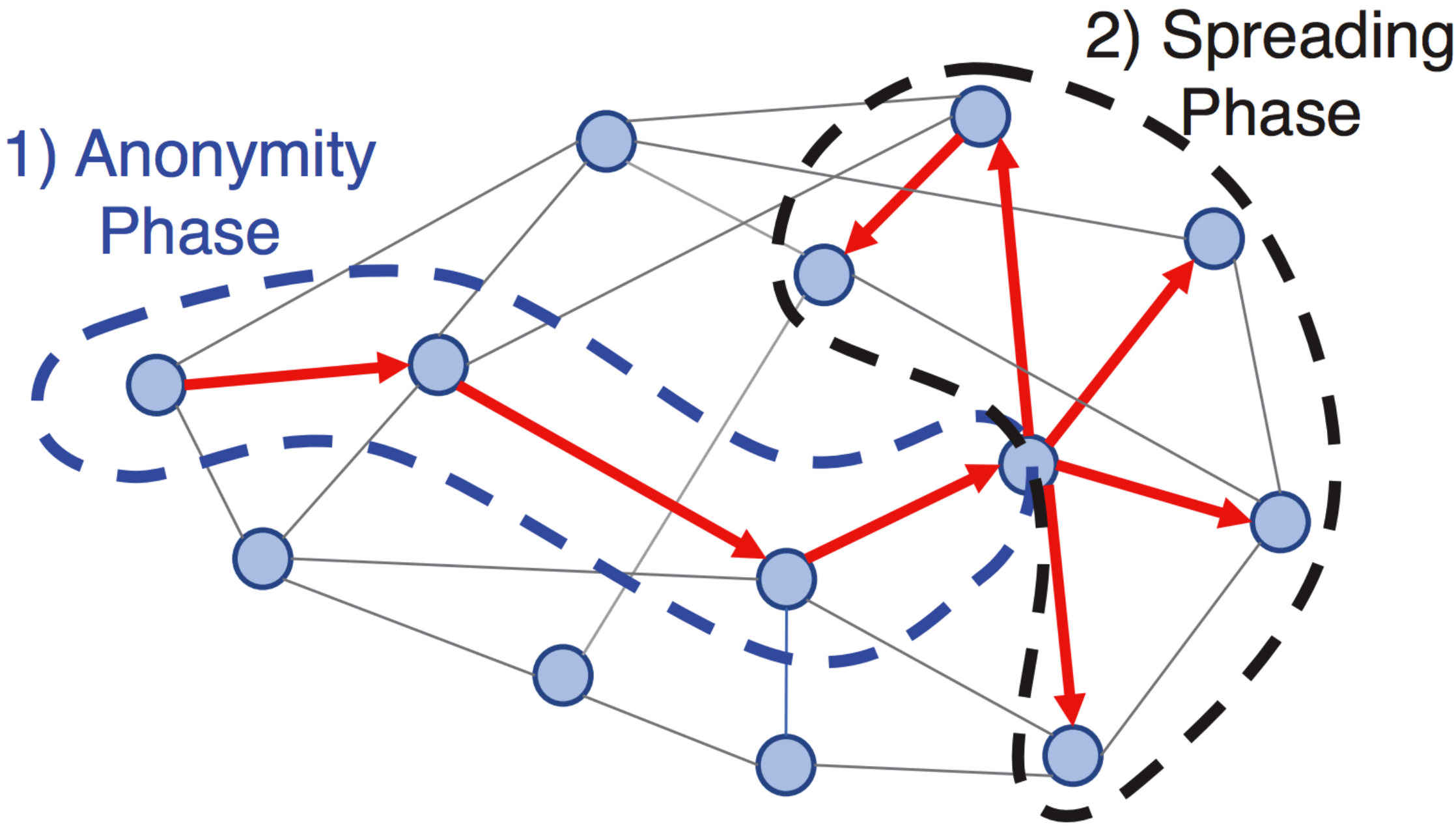
- $T_1 + T_2 = A + B + C + G + H - E - J - K$ is also valid

- Only final inputs and outputs need to be published

MimbleWimble

- The 2 popular implementations use proof-of-work
  - ASIC resistance through algorithms that use a lot of memory
- Mining Fees are special transactions added to each block
  - Similar to bitcoin

Dandelion Anonymity



Verifying the entire MW state

- The total amount of coins created by mining in the chain.
    - Easy: #block * mining_reward_per_block
- The complete set of unspent outputs
    - A Pedersen Commitment is a group element (64 bit)
- The transactions signatures for each transaction
    - Signature is (number, group element)-tuple (96 bits)
- Range proof ~ 1.5kb

Verifying the entire MW state

- The transactions signatures for each transaction contain the entire history
    - Contains information about every coin, even long after the coin was spend
    - Grows over time
- Consider 2 Schnorr signatures
    - ($r + \mathscr{H}(rG \,|\, pG \,|\, \text{text})p$, rG)
    - ($m + \mathscr{H}(mG \,|\, qG \,|\, \text{text})q$, qG)
    - Can not be combined without interaction
- Transaction signatures can not be compressed

Future of MimbleWimble

- Alternative signature scheme (BLS)
  - $S_{sum} = S_1 + S_2$
    - We can compress all transaction signatures of the past in one signature
- Verifying one signature is sufficient to proof that throughout the entire history, all transactions were correct