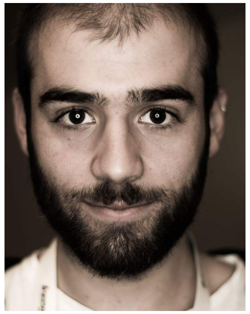
A cryptographic investigation of Mimblewimble

Georg Fuchsbauer

joint work with

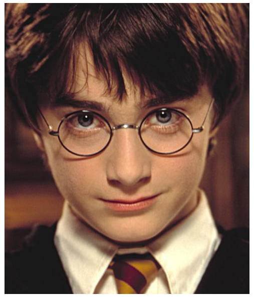
Michele Orrù



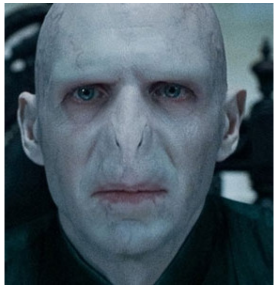
and Yannick Seurin

What is it?

- Proposal for a **cryptocurrency system**
  - **Privacy** (all amounts hidden; forget spent tx's)
  - **Scalability** (forget spent tx's)



- proposed by
  "Tom Elvis Jedusor"
  in 2016

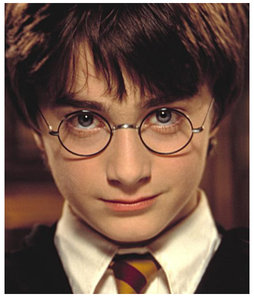


What is it?

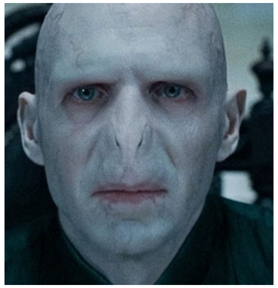
- Proposal for a **cryptocurrency system**
  - **Privacy** (all amounts hidden; forget spent tx's)
  - **Scalability** (forget spent tx's)

- implemented by *Grin*

- uses ideas from Gregory Maxwell



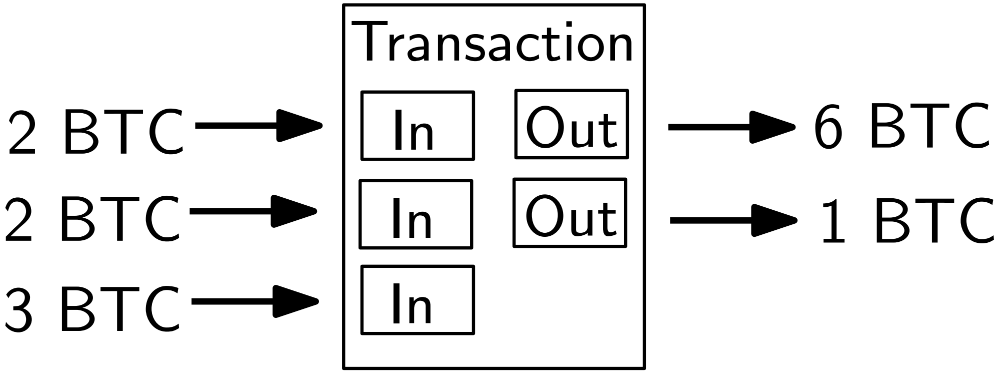
- proposed by
  "Tom Elvis Jedusor"
  in 2016



- further developed by Andrew Poelstra

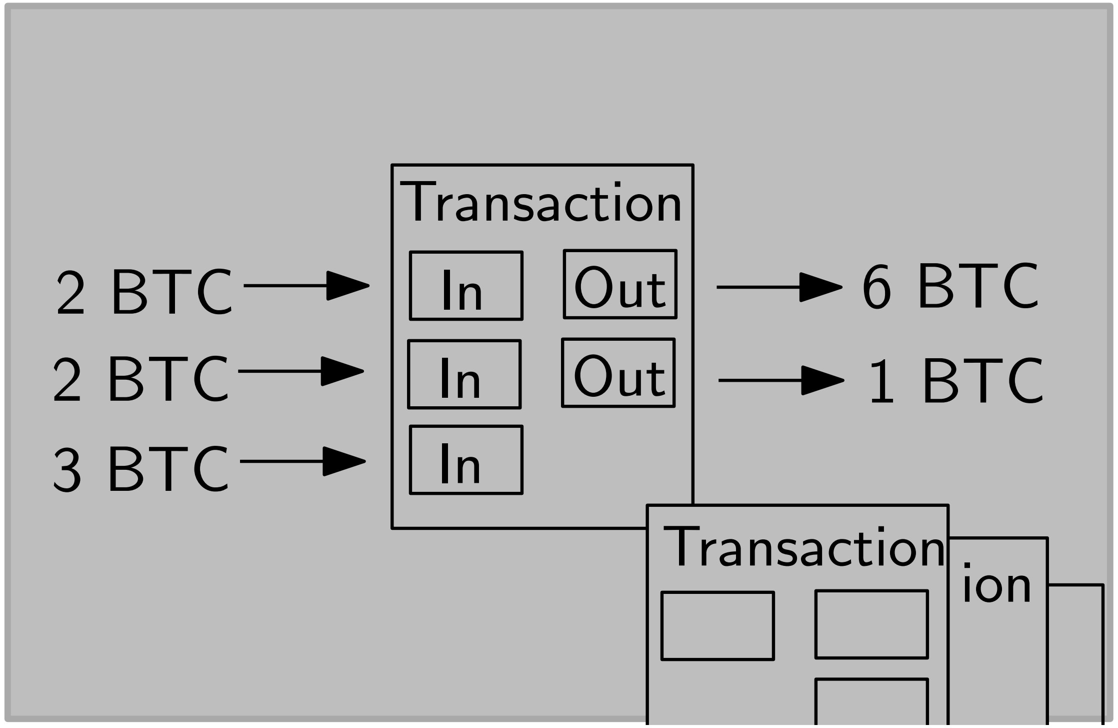
Bitcoin

- **Transactions**



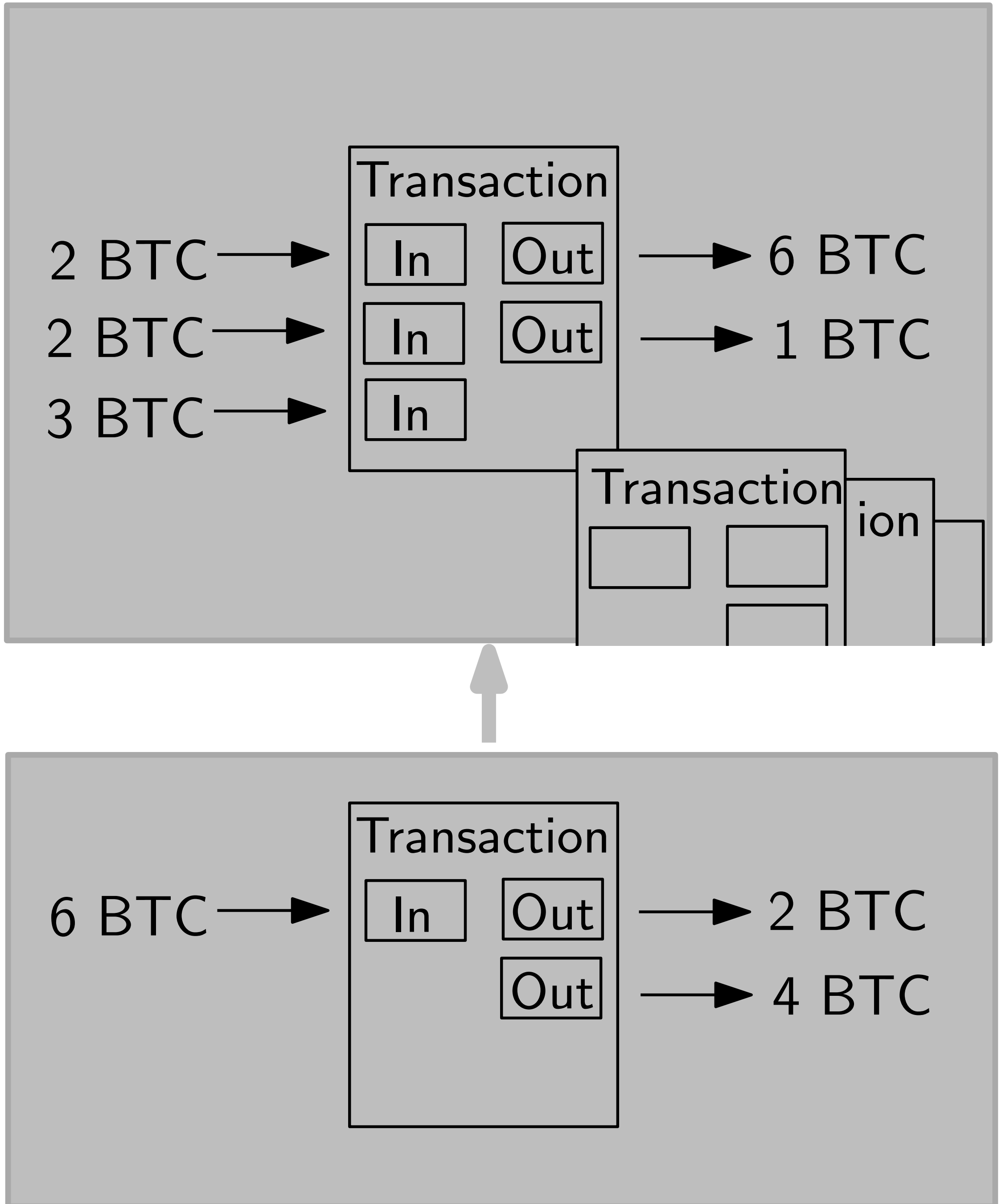
Bitcoin

- **Block**



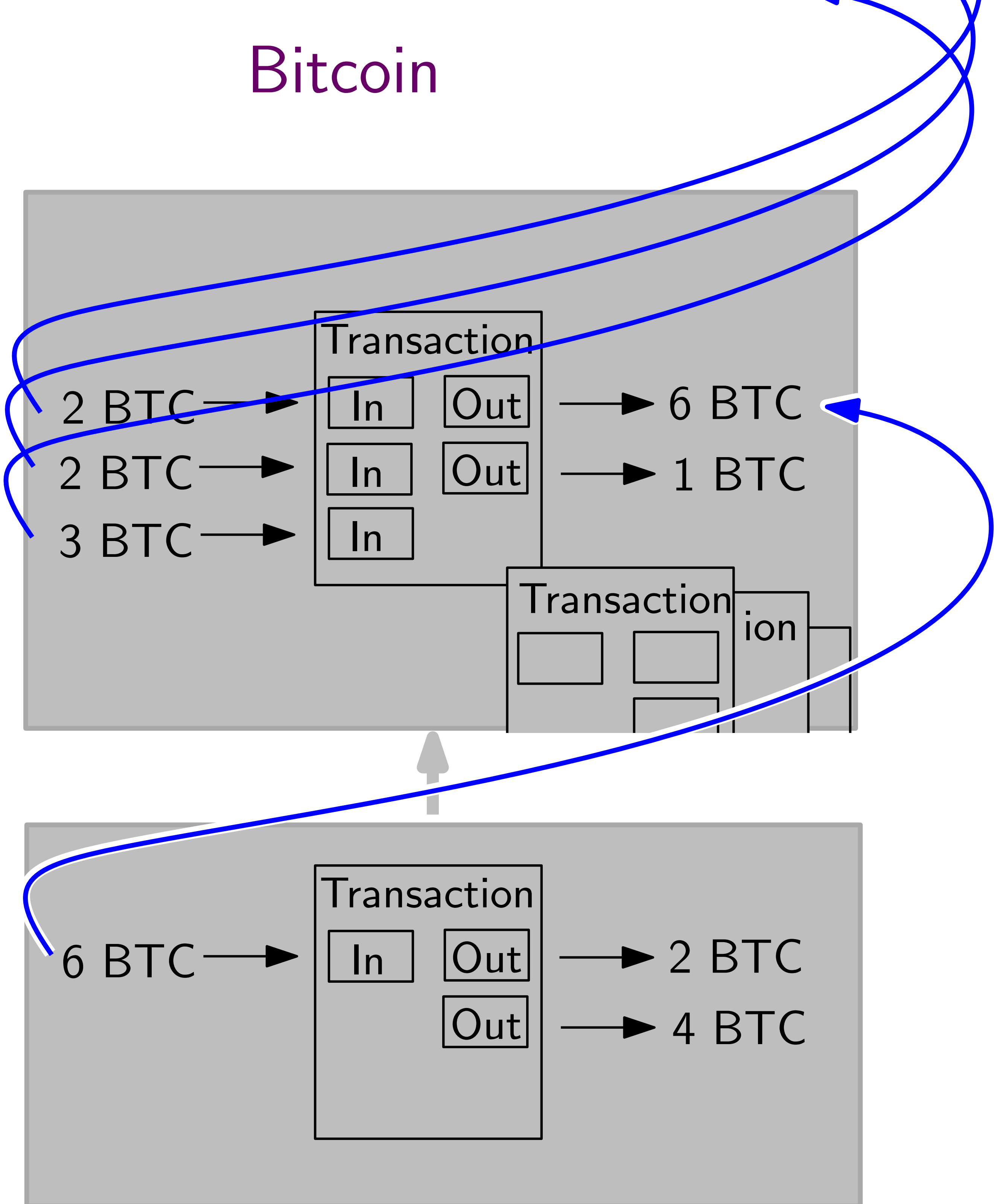
Bitcoin

- **Blockchain**



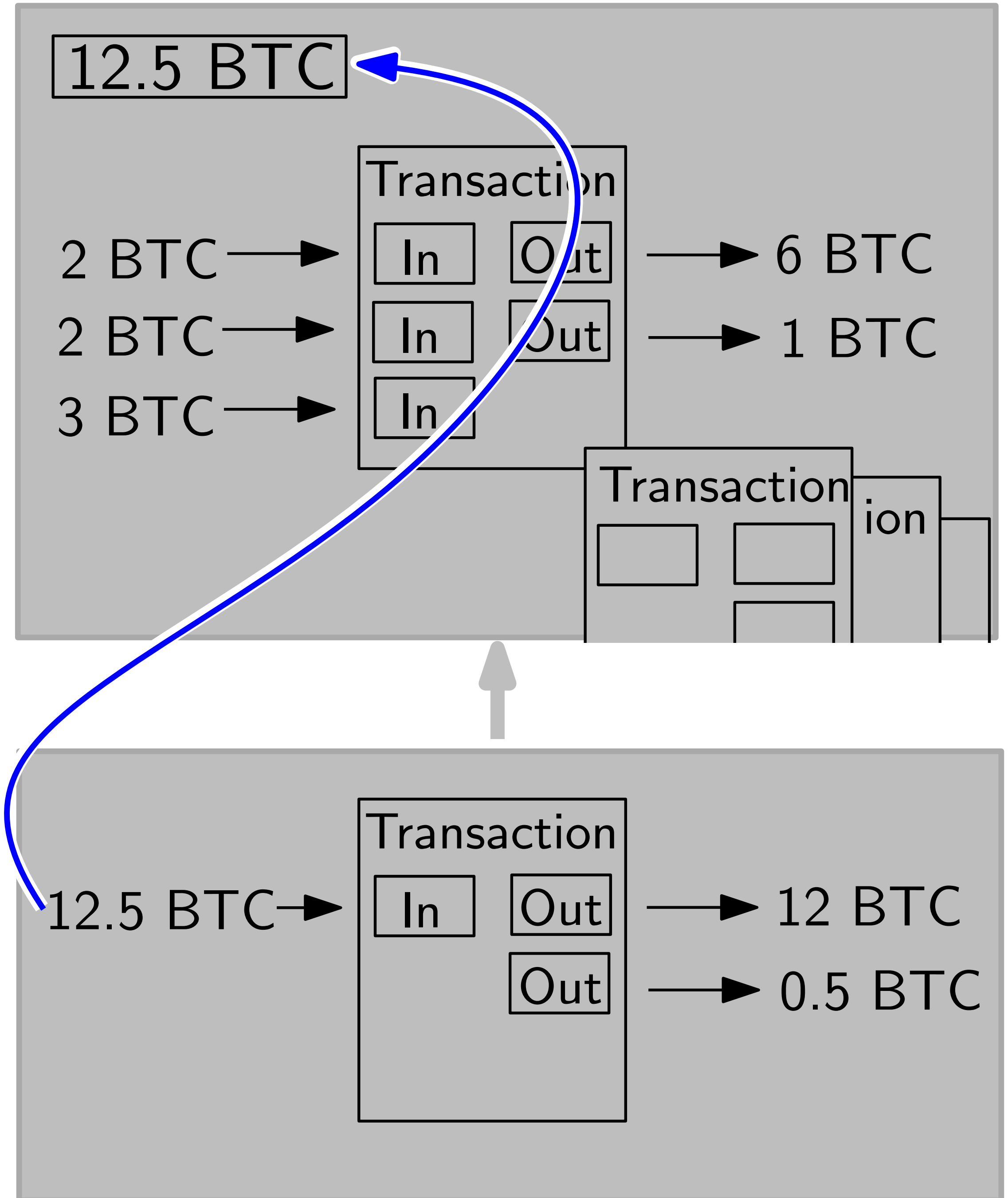
Bitcoin

- Reference to previous output

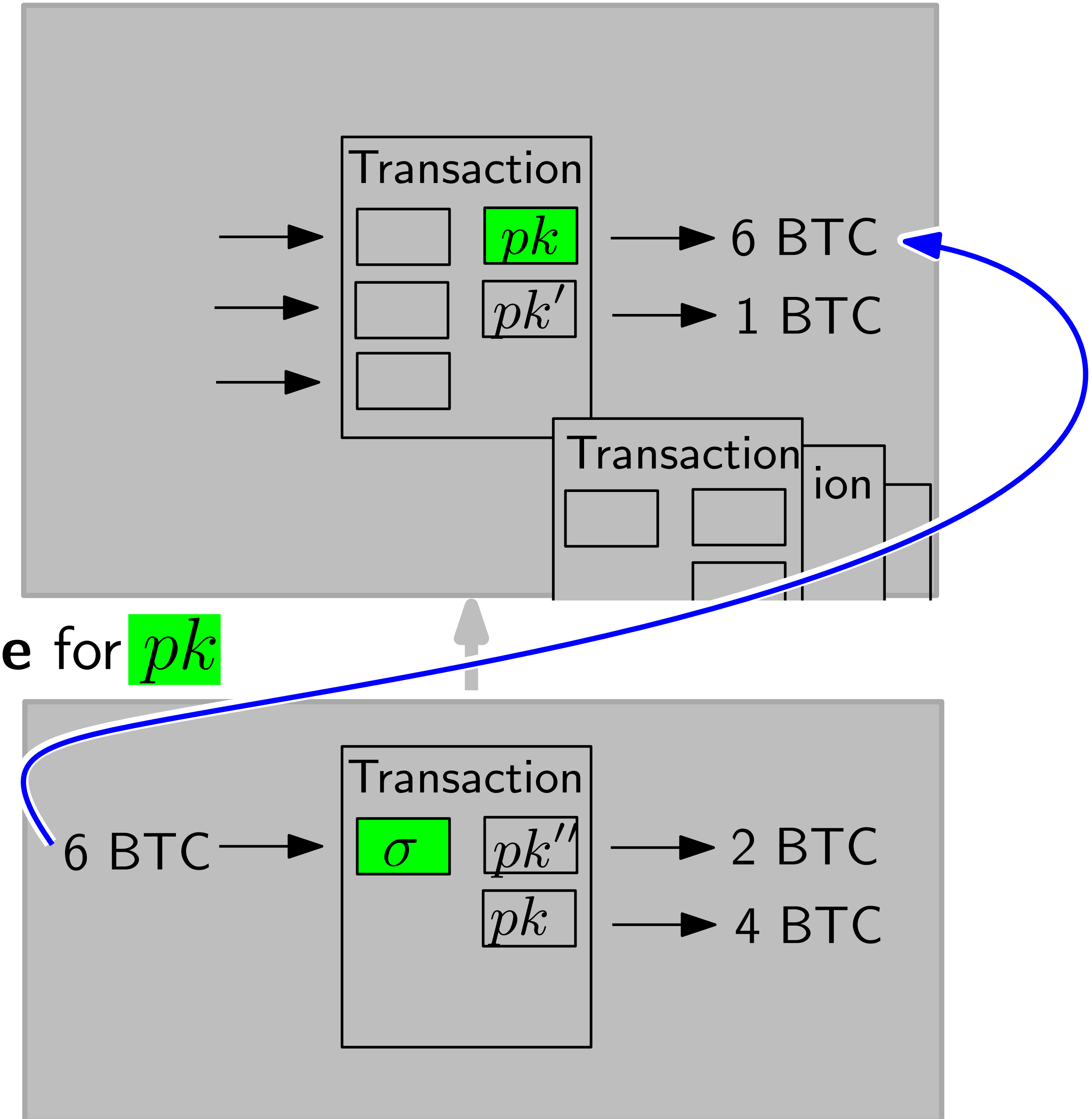


Bitcoin

- **Coinbase transaction**



Bitcoin



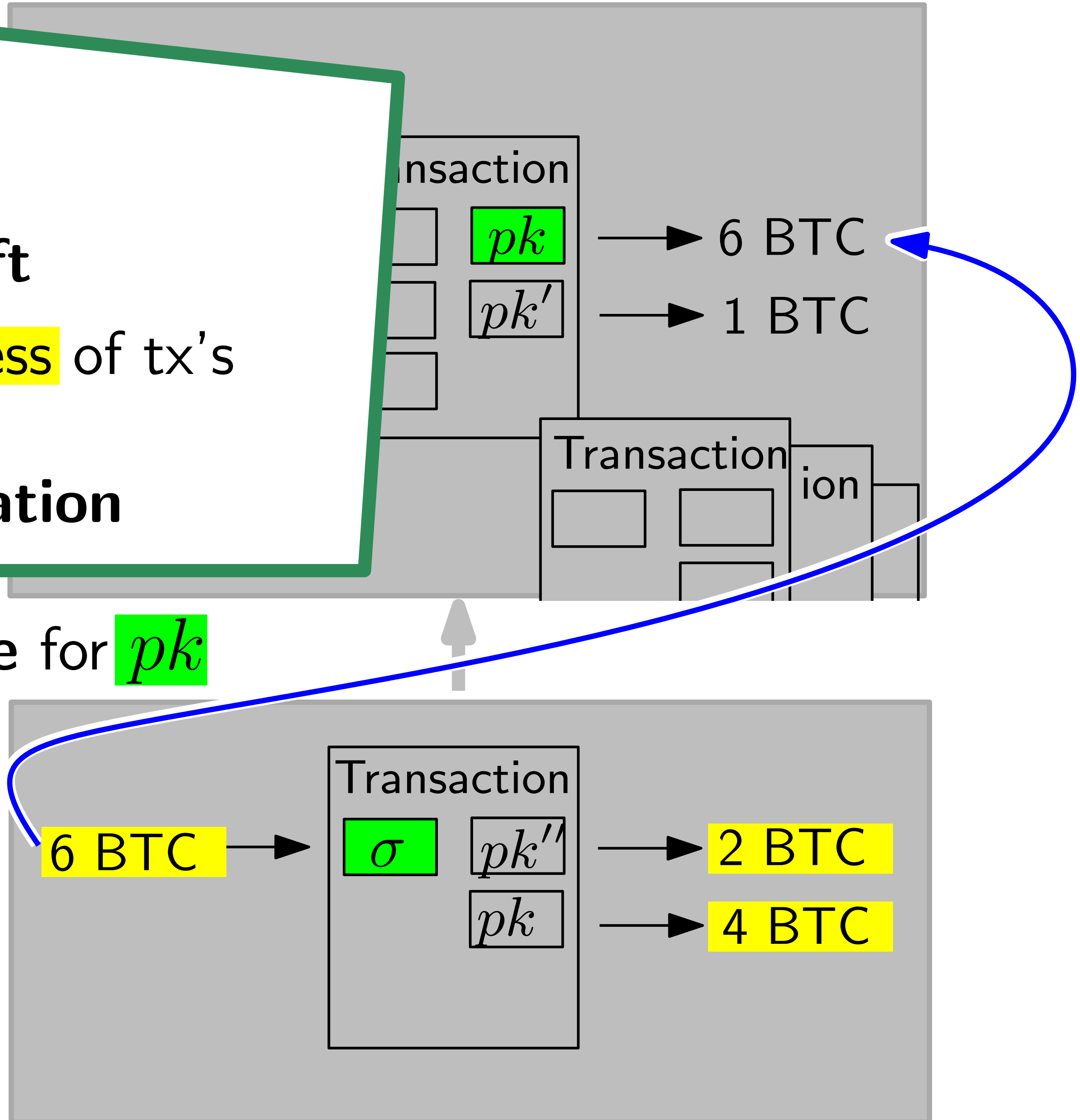
- σ is **signature** for pk

Bitcoin

Security

- signatures
  ⇒ **no theft**
- balancedness of tx's checkable
  ⇒ **no inflation**

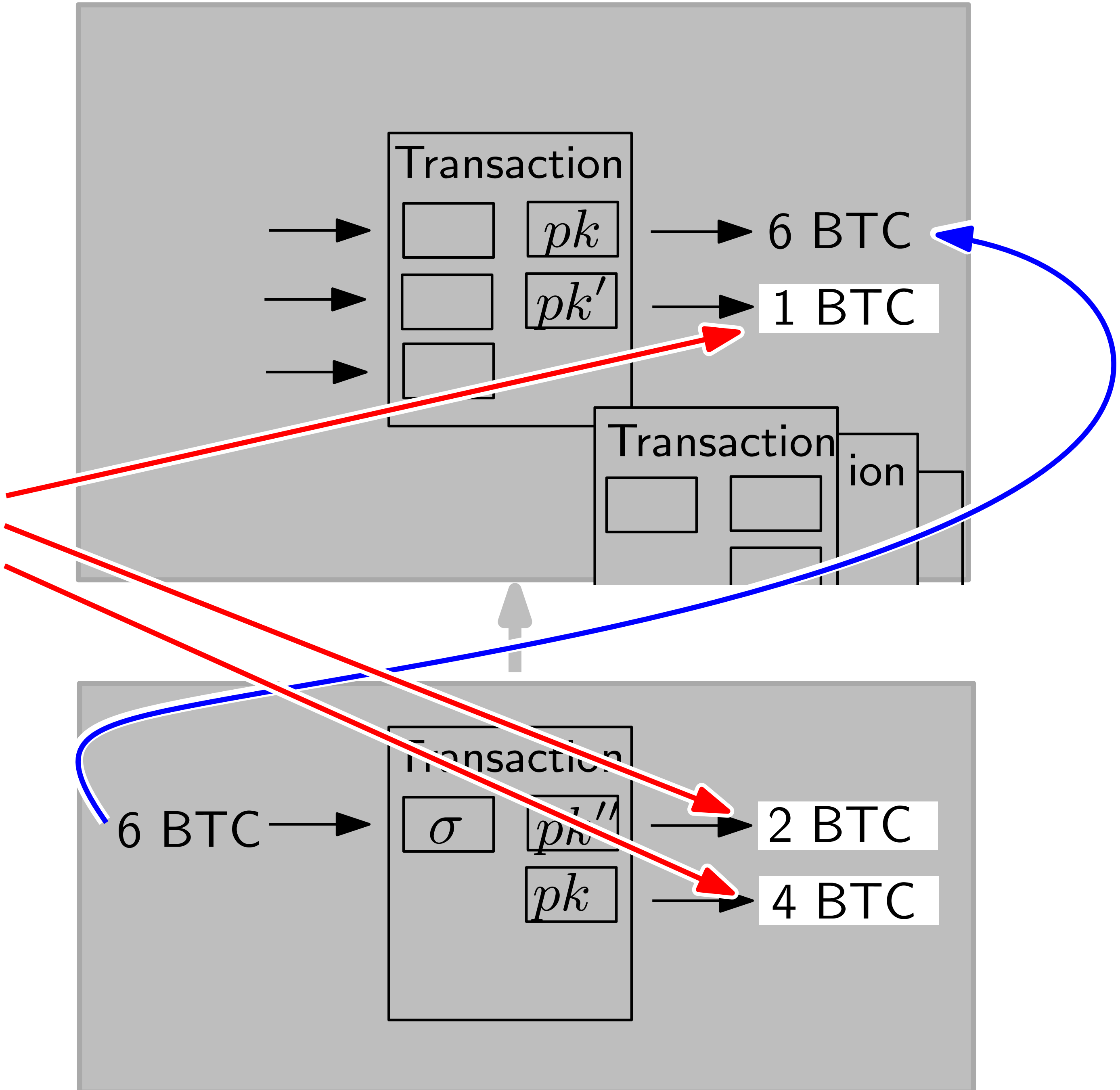
- σ is **signature** for pk



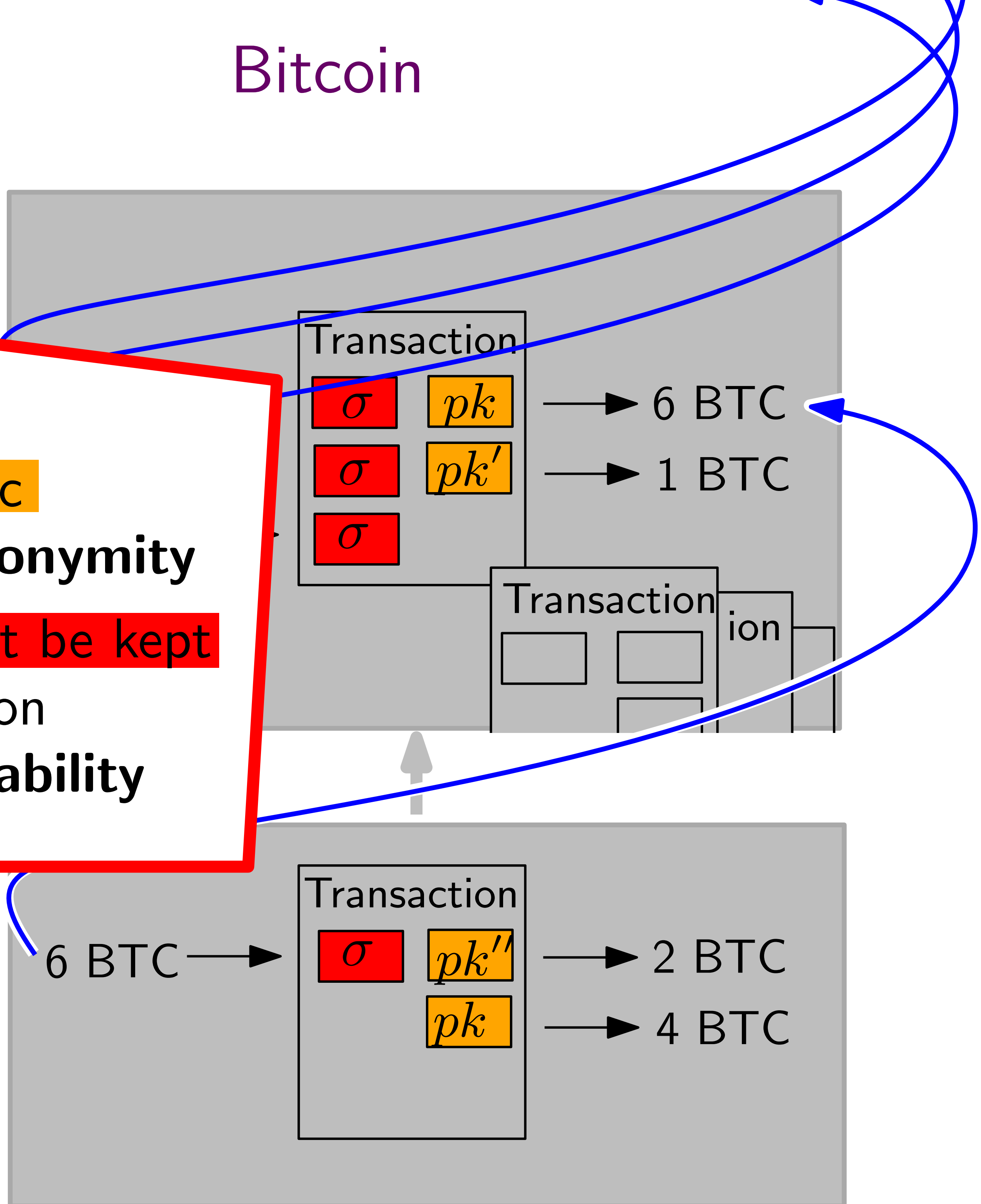
Bitcoin

Unspent transaction outputs (UTXO's)

= existing money in system



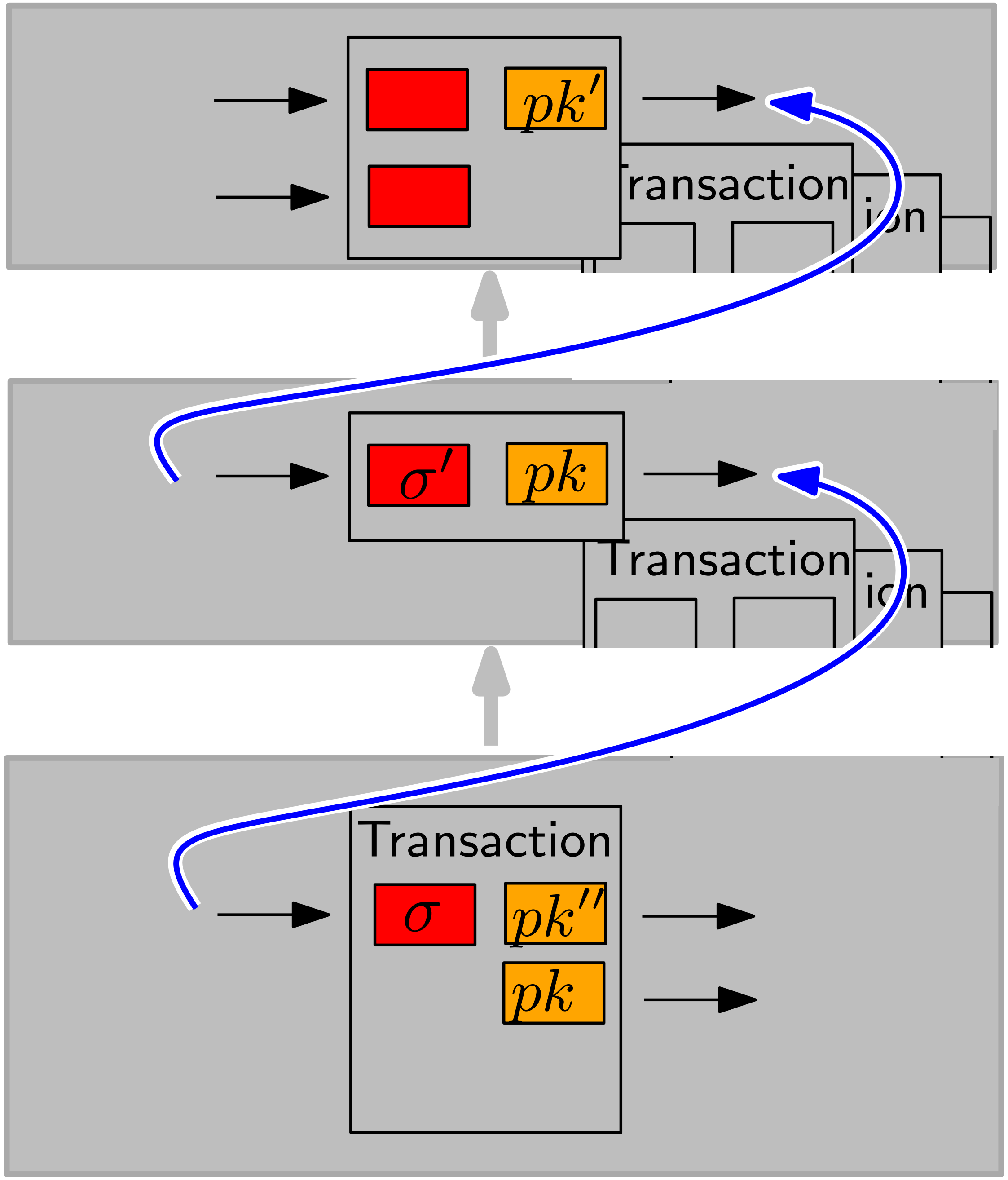
Bitcoin



Drawbacks

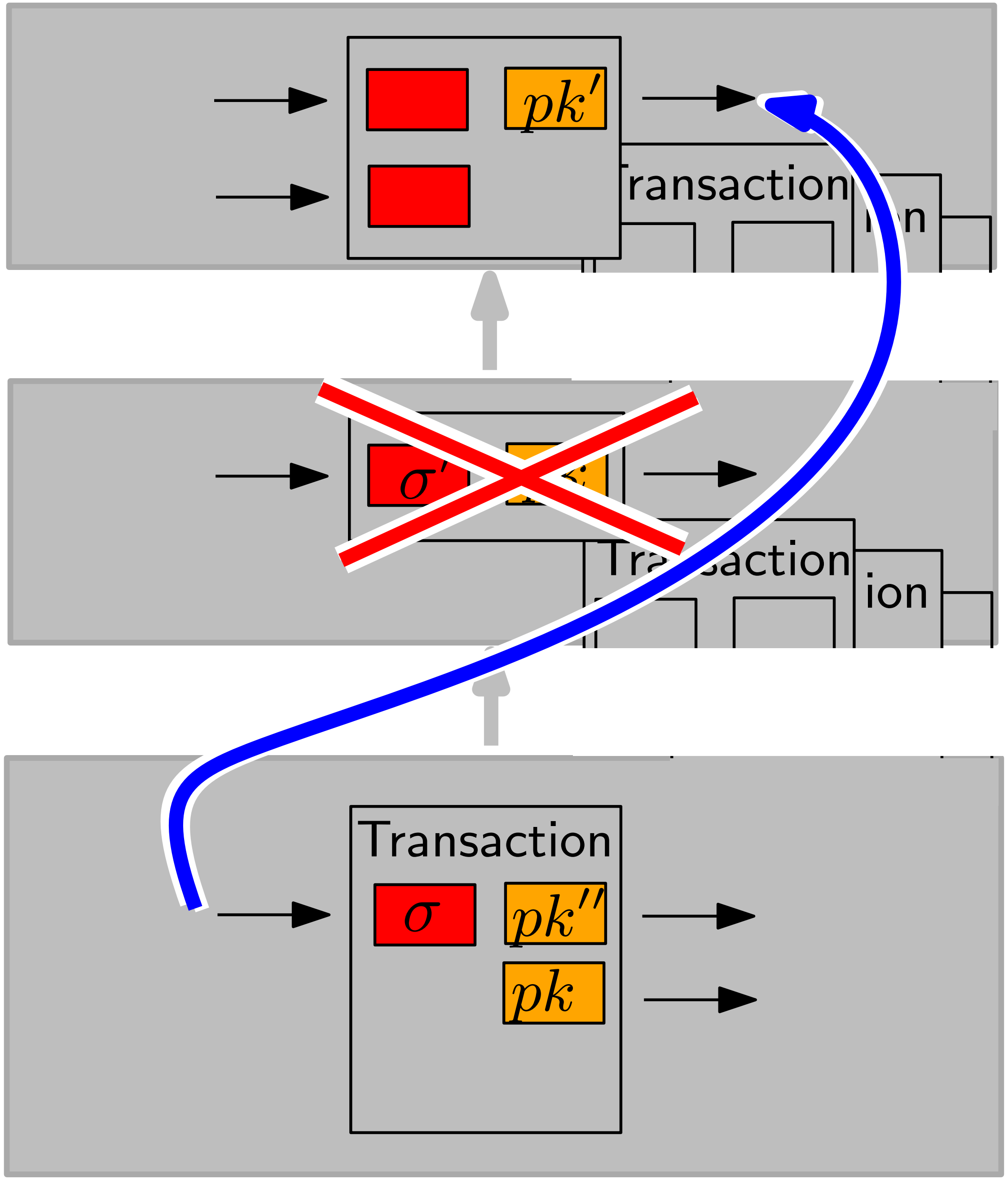
- all tx's public
  ⇒ **weak anonymity**
- all data must be kept for verification
  ⇒ **bad scalability**

Scalability



Scalability

"cut-through"



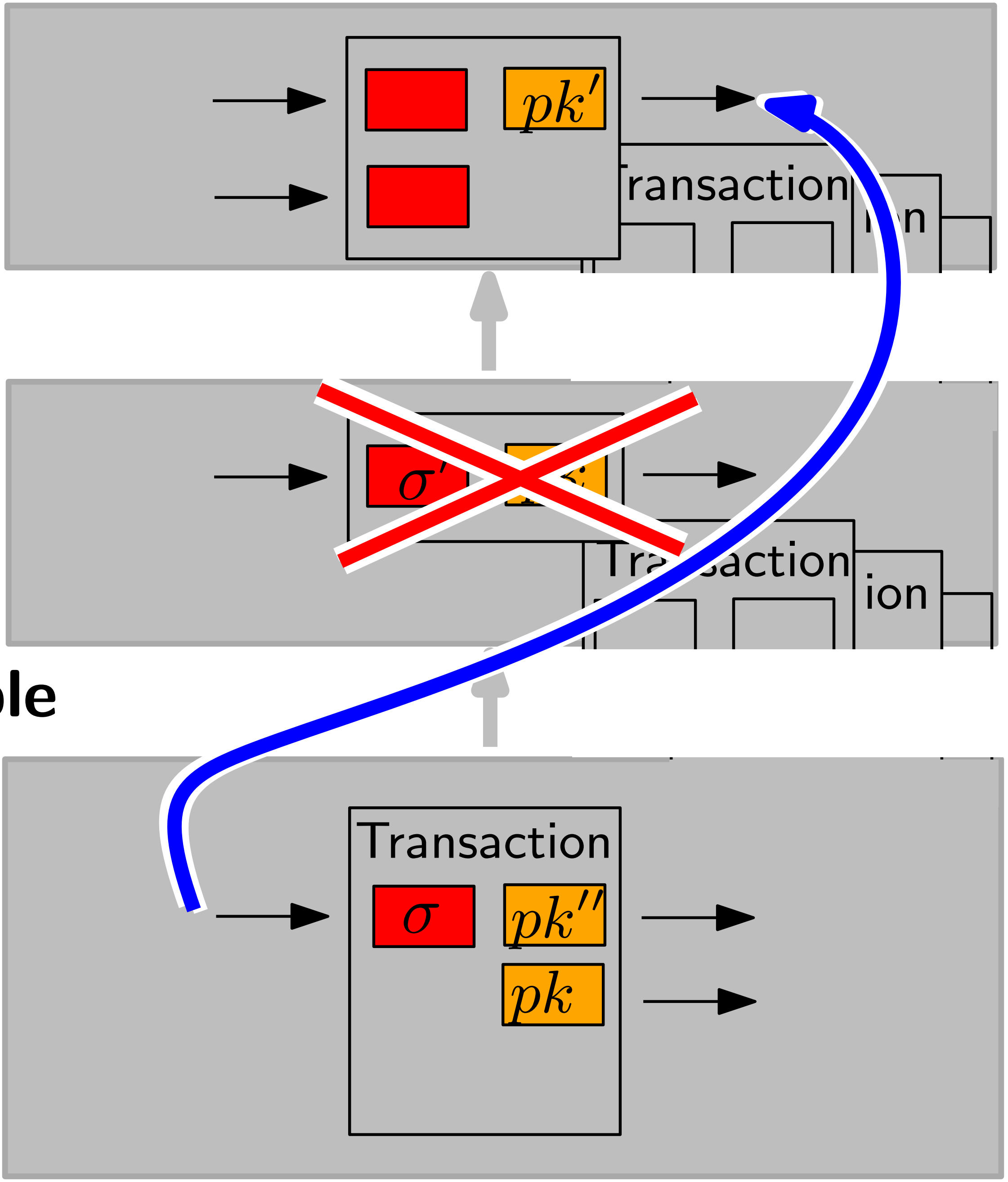
Scalability

"cut-through"

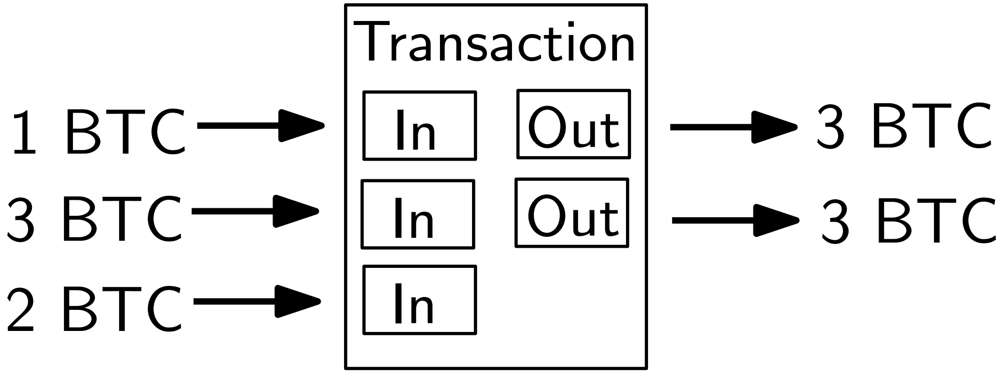
not possible in Bitcoin:

σ' is needed to verify validity

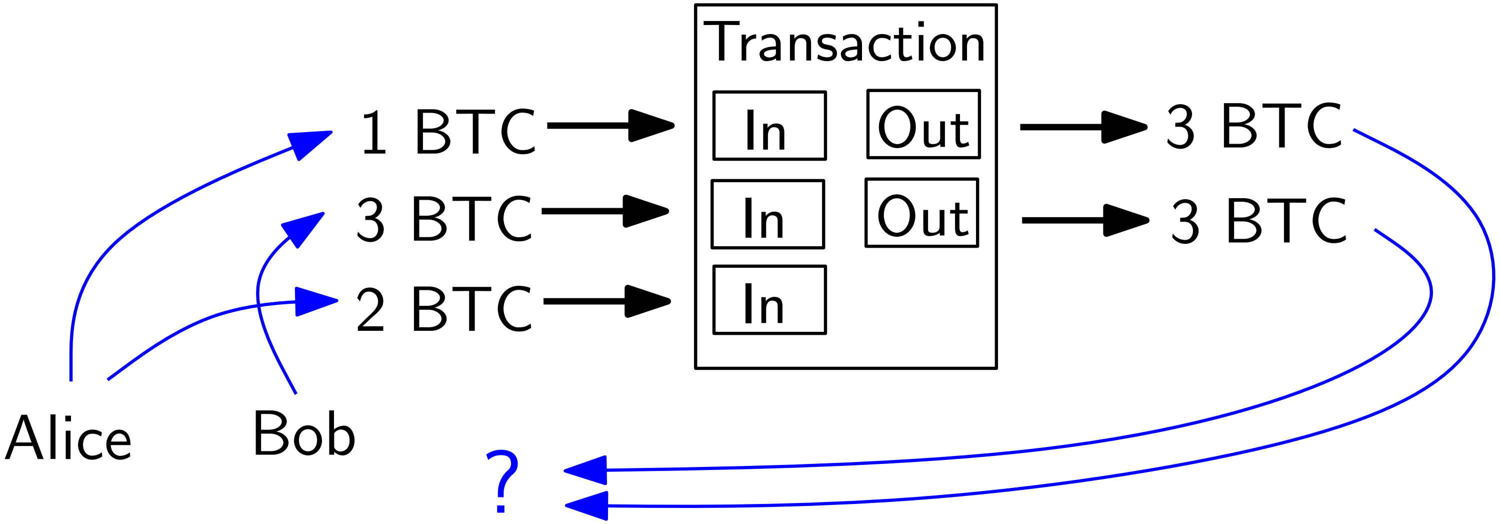
$\Rightarrow$ **Mimblewimble**



Anonymity

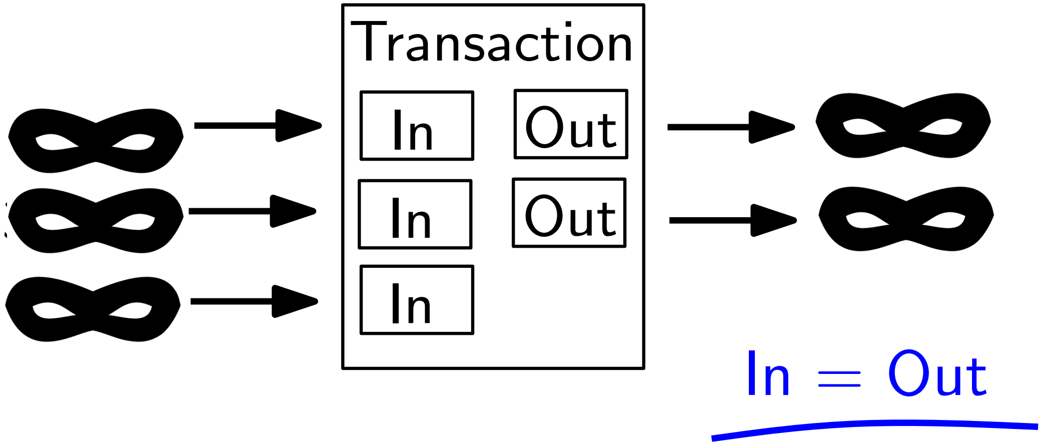


Anonymity



- **CoinJoin** [Maxwell'13]
  - no *link* between inputs and outputs
  - can we join many transactions together?
  - in Bitcoin: only interactively, since all inputs must sign tx

Anonymity



Transaction

In

Out

In

Out

In

In = Out

- **Confidential Transactions** [Maxwell]
  - hide the input and output *amounts*
  - not compatible with Bitcoin system
  - balancedness verifiable?

Anonymity

How can we get

- **Confidential transactions** (check balancedness)
- **Coin-join** (non-interactively)
- **Cut-through** (thus scalability)

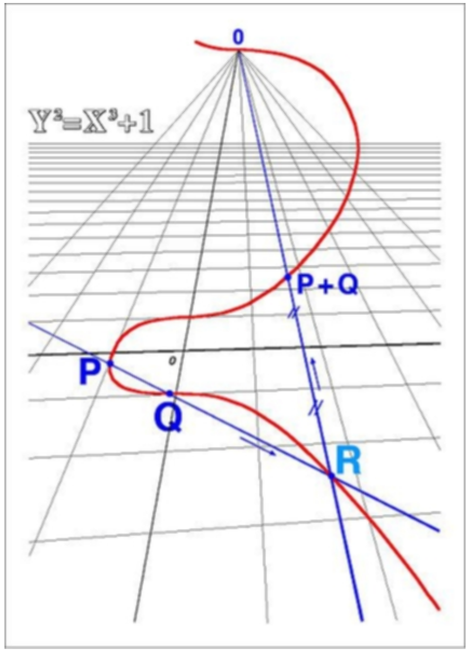
while **maintaining verifiability?**

- **Confider**
  - hide th
  - not co
  - balanc

Anonymity

- **Confider**
  - hide th
  - not co
  - balanc

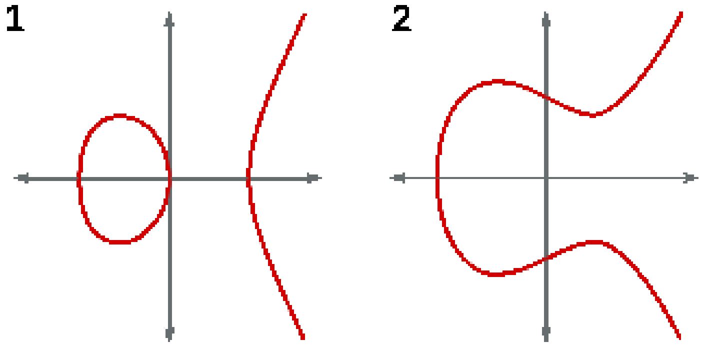
Mimblewimble



Some maths . . . and crypto!

Elliptic curves

- defined over finite field
- curve points can be added "$+$" $\Rightarrow$ group $\mathbb{G}$
  - generator G
  - $xG := \underbrace{G + \ldots + G}_{x \text{ times}}$

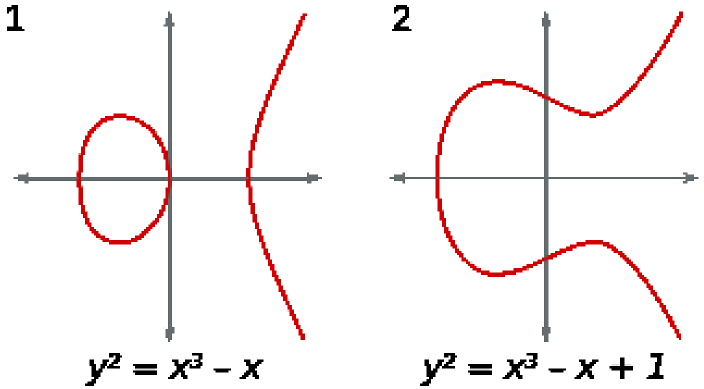


$y^2 = x^3 - x$

$y^2 = x^3 - x + 1$

Elliptic curves

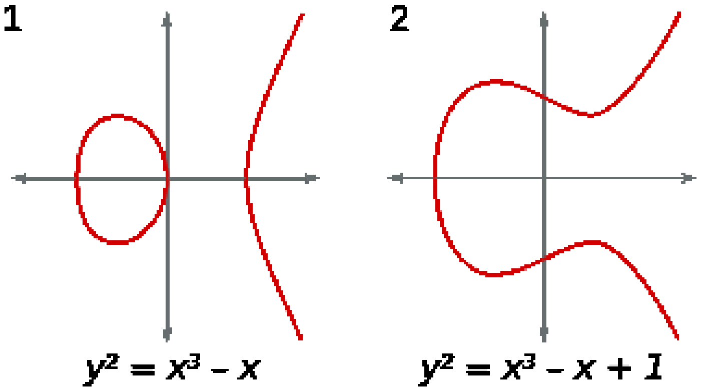
- defined over finite field
- curve points can be added "$+$" $\Rightarrow$ group $\mathbb{G}$
  - generator G
  - $xG := \underbrace{G + \ldots + G}_{x \text{ times}}$



- **Discrete logarithm** problem:
  - given $G, H \in \mathbb{G}$
  - find x such that $H = xG$

Elliptic curves

- defined over finite field
- curve points can be added "$+$" $\Rightarrow$ group $\mathbb{G}$
  - generator G
  - $xG := \underbrace{G + \ldots + G}_{x \text{ times}}$



- **Discrete logarithm** problem:
  - given $G, H \in \mathbb{G}$
  - find x such that $H = xG$

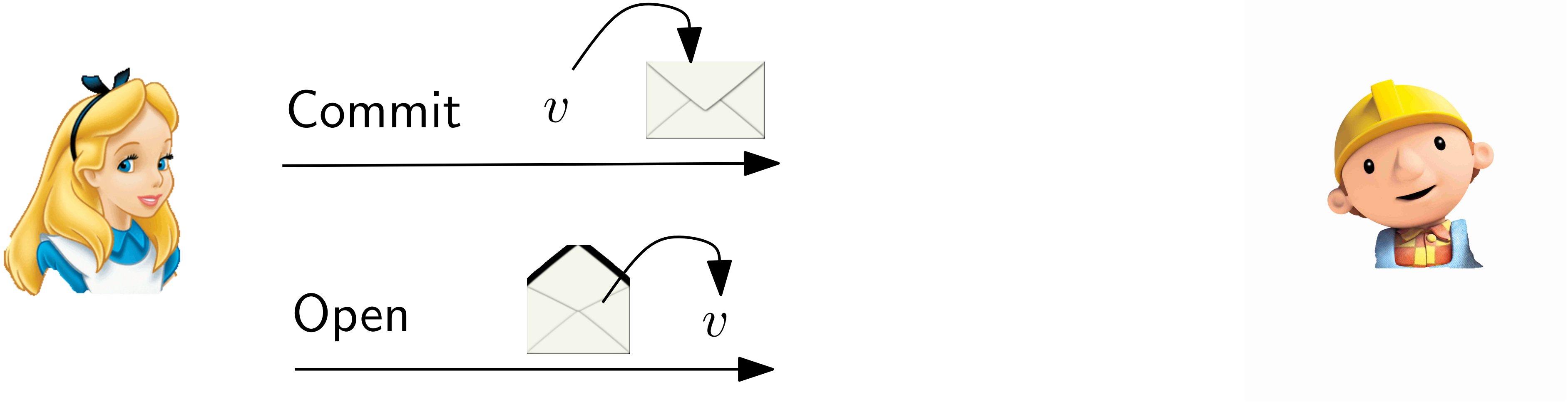
- used in **signature schemes**
  (e.g. ECDSA ₿,
  Schnorr)

  ○ secret key: x
  ○ public key: $X = xG$

Pedersen commitment

Commitment

- "digital envelope"



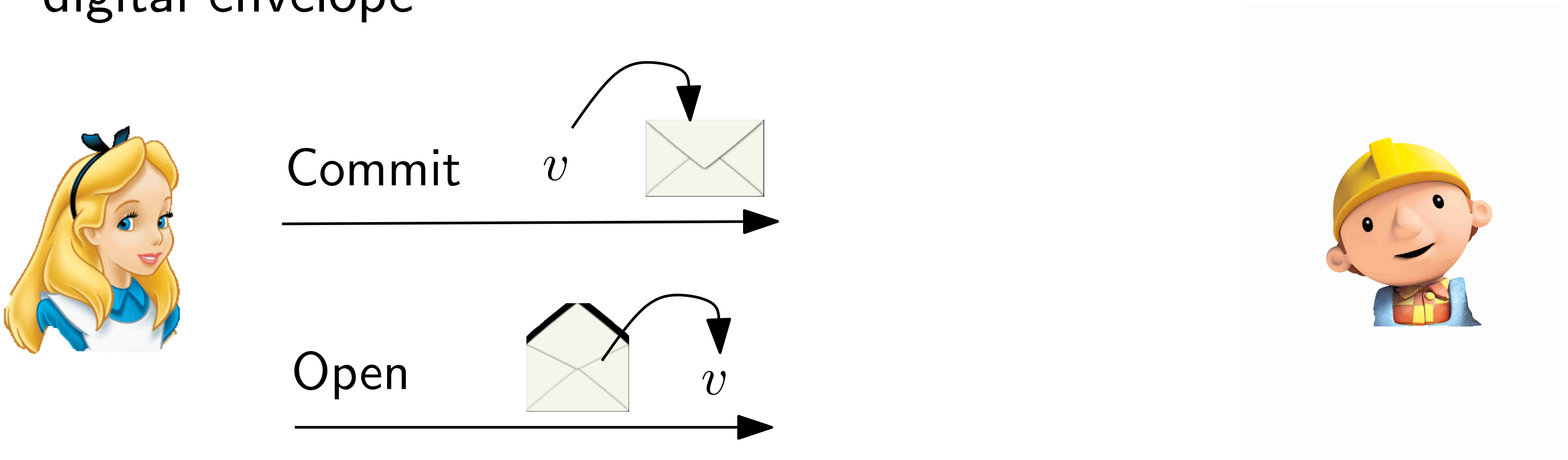
Commit v

Open v

Pedersen commitment

Commitment

- "digital envelope"



Commit v

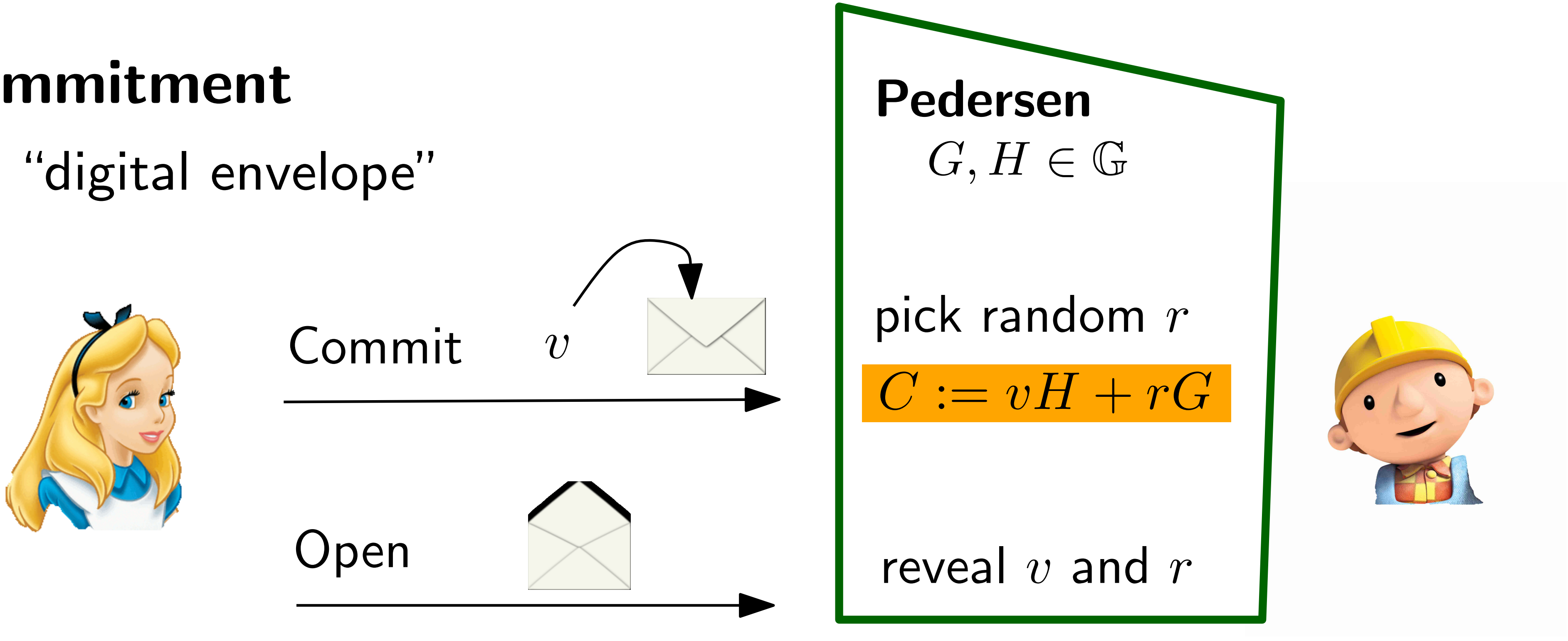
Open v

- **hiding:** commitment hides v
- **binding:** Alice can open commitment only to one value

Pedersen commitment

Commitment

- "digital envelope"



Commit v

Open

Pedersen

$G, H \in \mathbb{G}$

pick random r

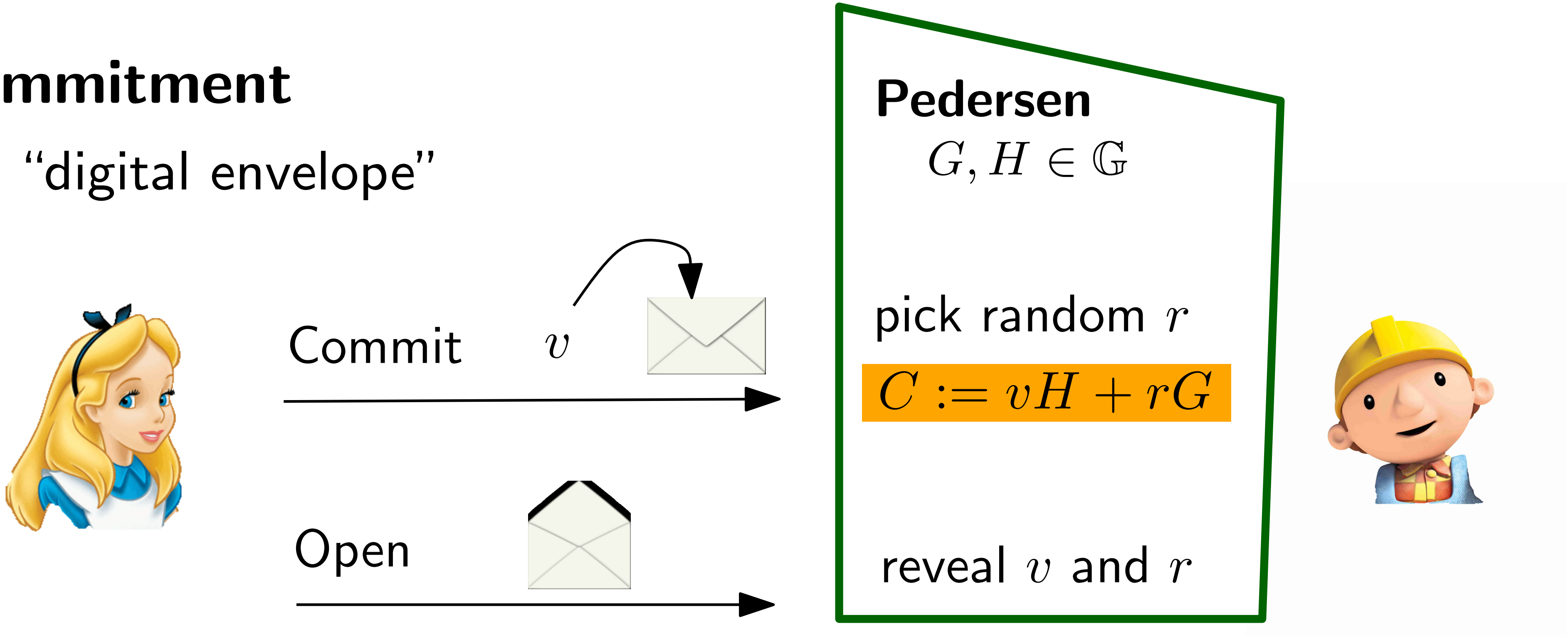
$C := vH + rG$

reveal v and r

Pedersen commitment

Commitment

- "digital envelope"



Commit v

Open

Pedersen

$G, H \in \mathbb{G}$

pick random r

$C := vH + rG$

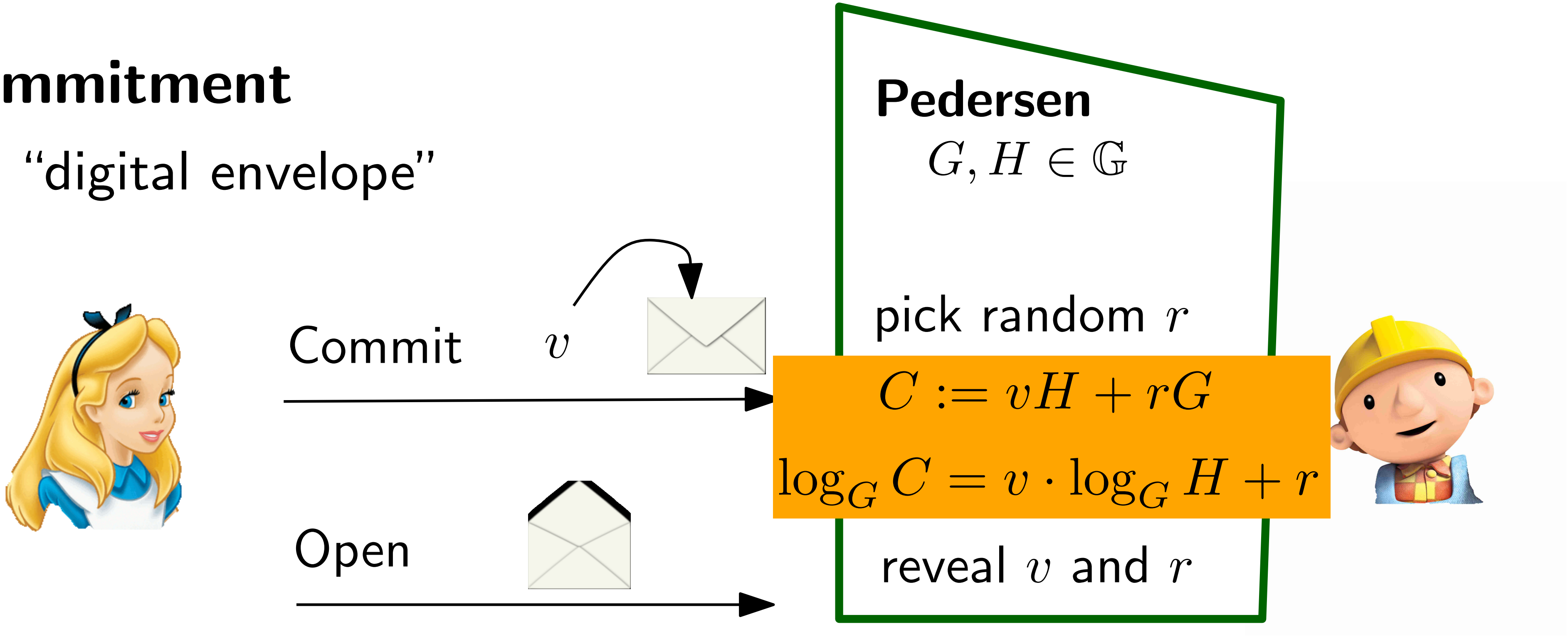
reveal v and r

- **hiding:** for any v exists r so that C commits v

Pedersen commitment

Commitment

- “digital envelope”



Commit v

Open

Pedersen

$G, H \in \mathbb{G}$

pick random r

$$C := vH + rG$$

$$\log_G C = v \cdot \log_G H + r$$

reveal v and r

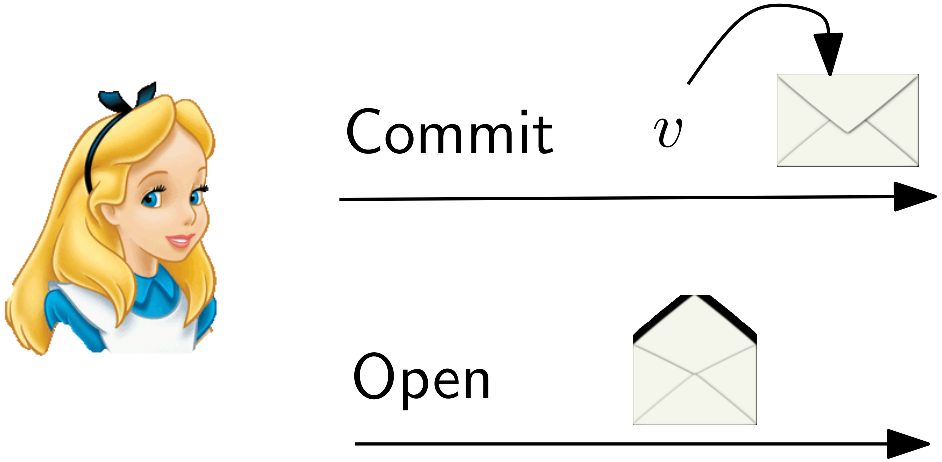
- **hiding:** for any v exists r so that C commits v:

$$(r = \log_G C - v \cdot \log_G H)$$

Pedersen commitment

Commitment

- "digital envelope"



Commit v

Open

Pedersen

$G, H \in \mathbb{G}$

pick random r

$C := vH + rG$

reveal v and r

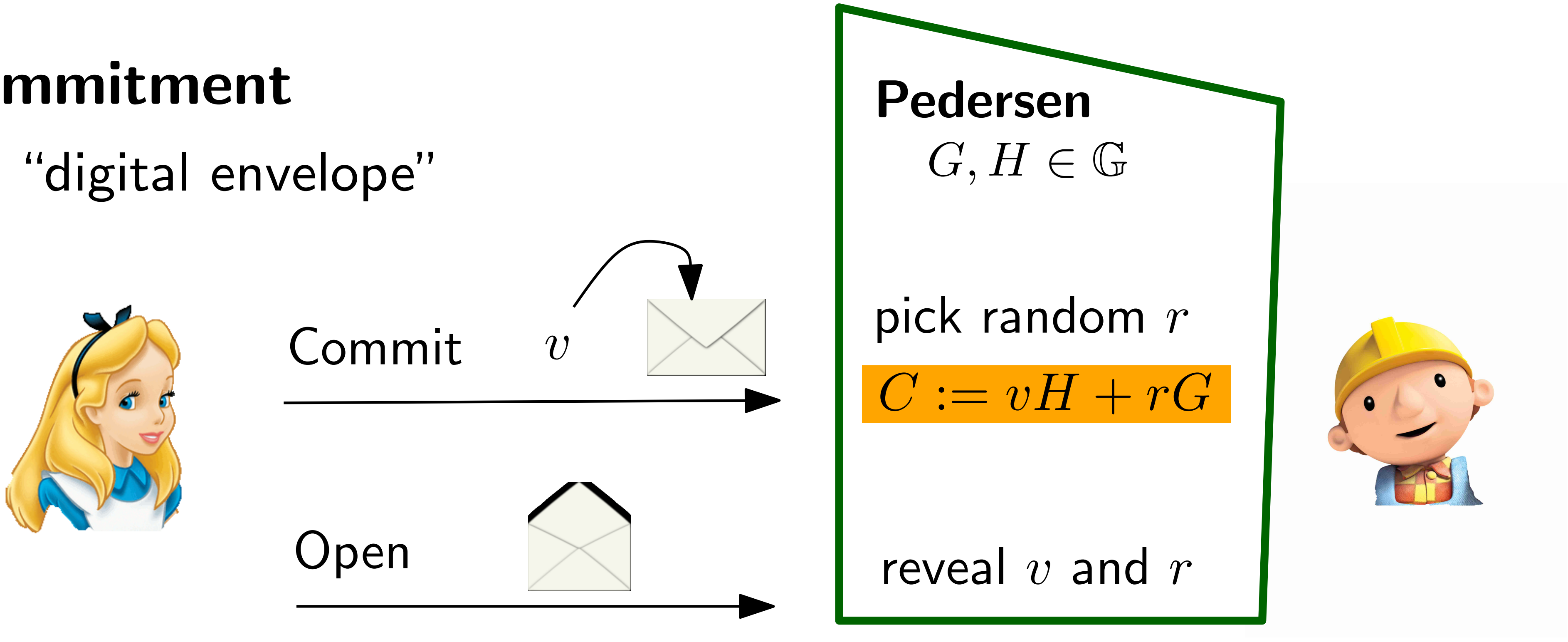
- **binding:** assume Alice finds v, r, v', r' with

$$vH + rG = C = v'H + r'G$$

Pedersen commitment

Commitment

- “digital envelope”



Commit v

Open

Pedersen

$G, H \in \mathbb{G}$

pick random r

$C := vH + rG$

reveal v and r

- **binding:** assume Alice finds v, r, v', r' with

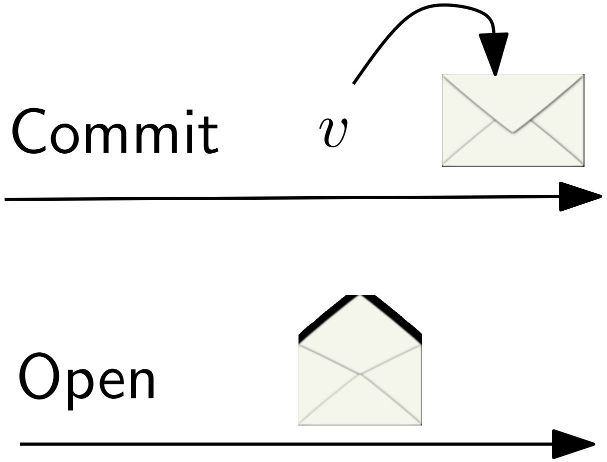
$$vH + rG = C = v'H + r'G, \quad \text{then } \frac{r'-r}{v-v'}G = H$$

$\Rightarrow$ Alice solved discrete log problem!

Pedersen commitment

Commitment

- "digital envelope"



Pedersen
$G, H \in \mathbb{G}$

pick random r

Com$(v; r) := vH + rG$

reveal v and r

- commitments are **homomorphic**:

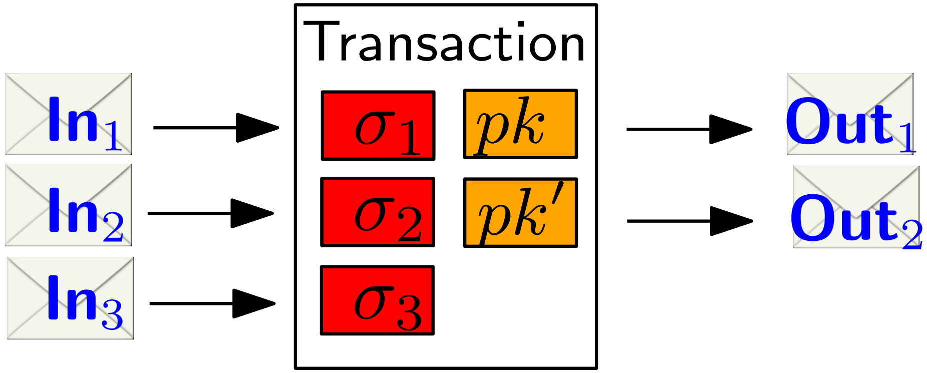
$$
\begin{aligned}
\textbf{Com}(v_1; r_1) + \textbf{Com}(v_2; r_2) &= (v_1 H + r_1 G) \; + \; (v_2 H + r_2 G) \\
&= (v_1 + v_2) H + (r_1 + r_2) G \\
&= \textbf{Com}(v_1 + v_2; r_1 + r_2)
\end{aligned}
$$

e.g.: $\textbf{Com}(1; 5) + \textbf{Com}(1; 10) - \textbf{Com}(2, 15) = \mathbf{0}$

Confidential Transactions

[Back,Maxwell '13–'15]

- use *commitments* to amounts

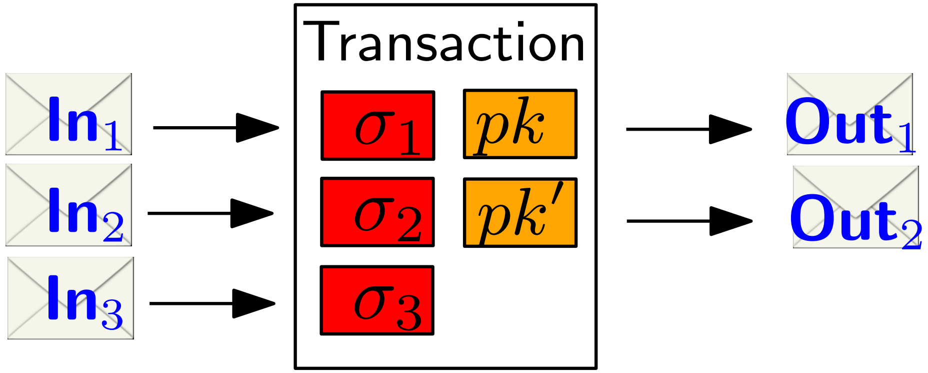


$$C = vH + rG$$

Confidential Transactions

[Back,Maxwell '13–'15]

- use *commitments* to amounts
- ensure that transactions do not create money?



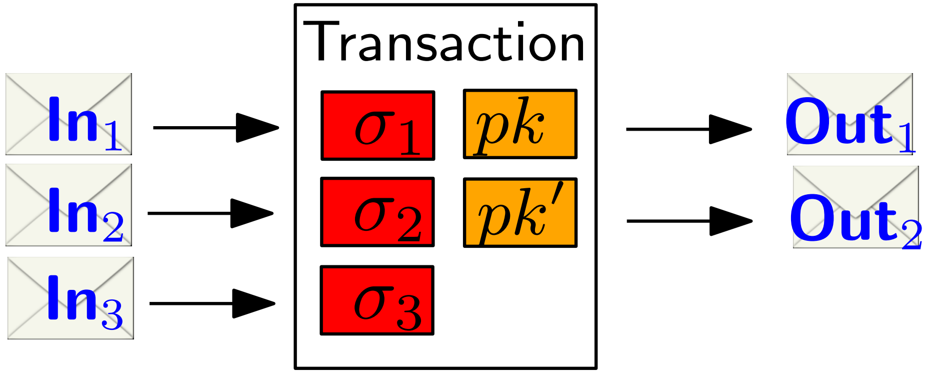
$$C = vH + rG$$

$$\boxed{\begin{aligned}&\mathbf{Out}_1 + \ldots + \mathbf{Out}_n\\ &\quad - \mathbf{In}_1 - \ldots - \mathbf{In}_\ell = \mathbf{0}\end{aligned}}$$

Confidential Transactions

[Back,Maxwell '13–'15]

- use *commitments* to amounts
- ensure that transactions do not create money?



$$C = vH + rG$$

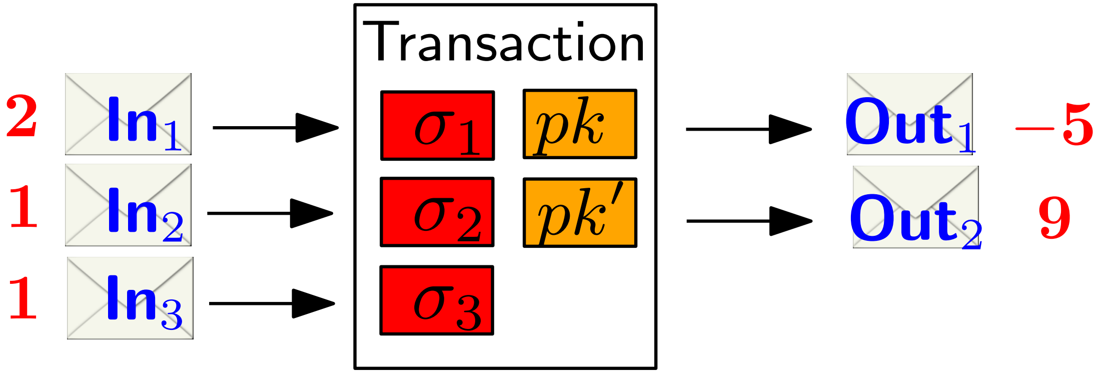
$$\boxed{\sum \textbf{Out} - \sum \textbf{In} = 0}$$

$$\begin{aligned}
&\sum C_i^{\text{out}} - \sum C_i^{\text{in}} \\
&= \sum(v_i^{\text{out}} H + r_i^{\text{out}} G) - \sum(v_i^{\text{in}} H + r_i^{\text{in}} G) \\
&= \underbrace{(\sum v_i^{\text{out}} - \sum v_i^{\text{in}})}_{\overset{!}{=}0} H + \underbrace{(\sum r_i^{\text{out}} - \sum r_i^{\text{in}})}_{\overset{!}{=}0} G
\end{aligned}$$

Confidential Transactions

[Back,Maxwell '13–'15]

- use *commitments* to amounts
- ensure that transactions do not create money?



$$C = vH + rG$$

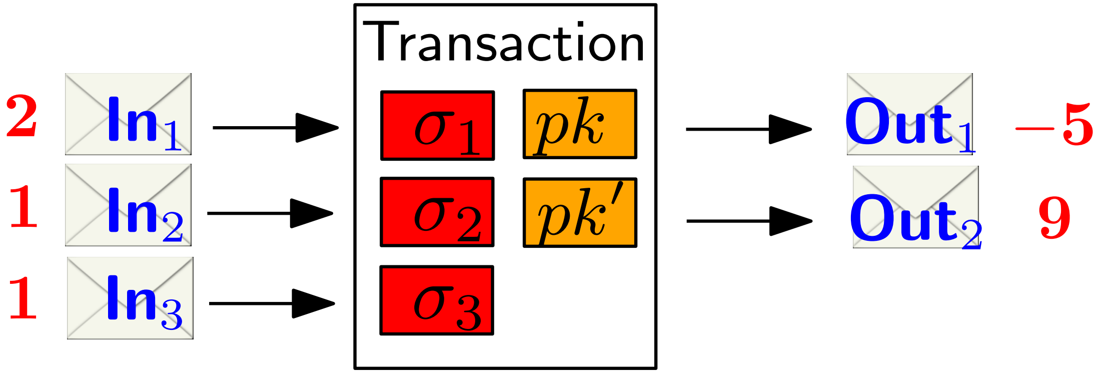
$$\boxed{\sum \mathbf{Out} - \sum \mathbf{In} = 0}$$

- negative amounts!

Confidential Transactions

[Back,Maxwell '13–'15]

- use *commitments* to amounts
- ensure that transactions do not create money?



$C = vH + rG,\ \pi$

$\sum \mathbf{Out} - \sum \mathbf{In} = 0$

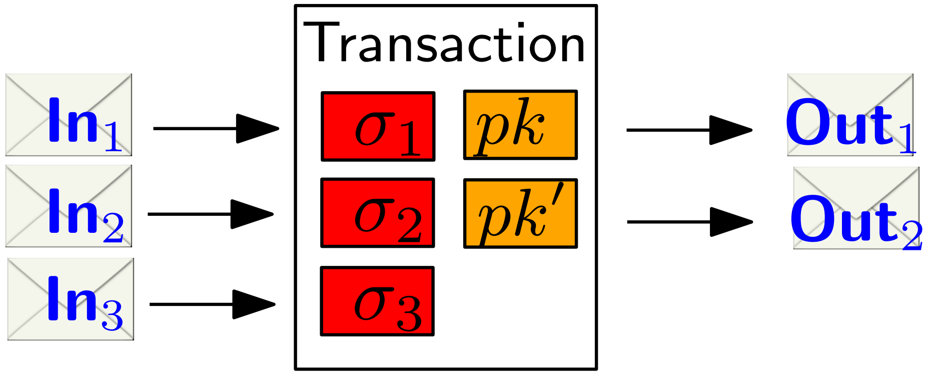
- negative amounts!

Range proofs

– add proofs that committed values are in $\in [0, 2^{64}]$

Confidential Transactions

Confidential transaction



$$C = vH + rG, \quad \pi$$

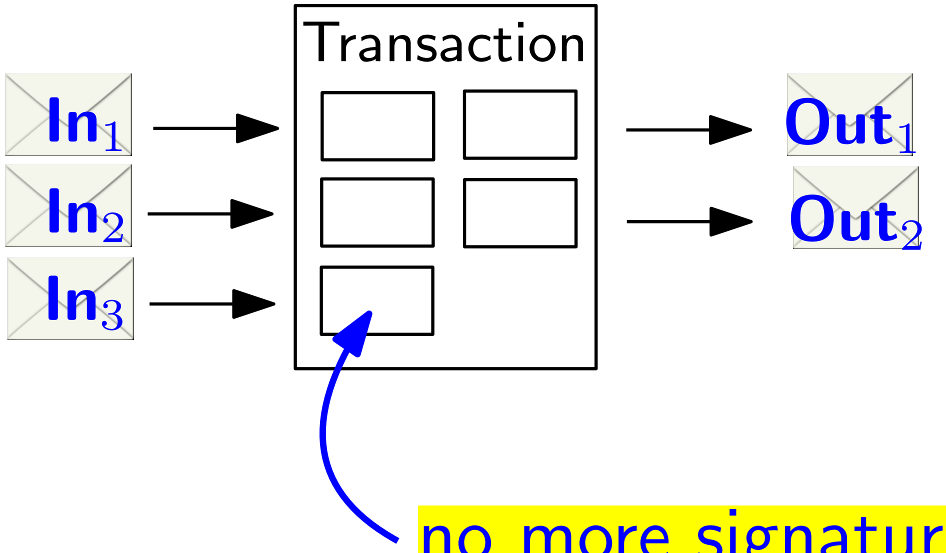
$$\sum \mathbf{Out} - \sum \mathbf{In} = \mathbf{0}$$

Signatures ⇒

- no non-interactive CoinJoin
- no Cut-Through

Mimblewimble

[Jedusor '16]



no more signatures!

secret key!

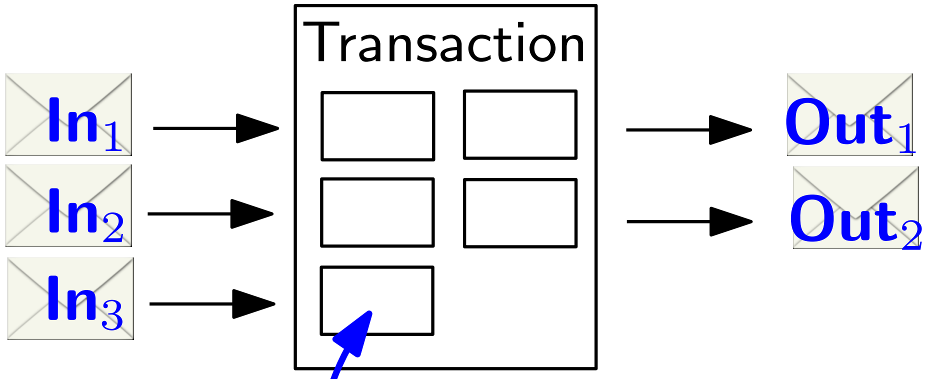
$$C = vH + rG, \quad \pi$$

$$\boxed{\sum \mathbf{Out} - \sum \mathbf{In} = 0}$$

Mimblewimble

[Jedusor '16]

secret key!



$$C = vH + rG, \quad \pi$$

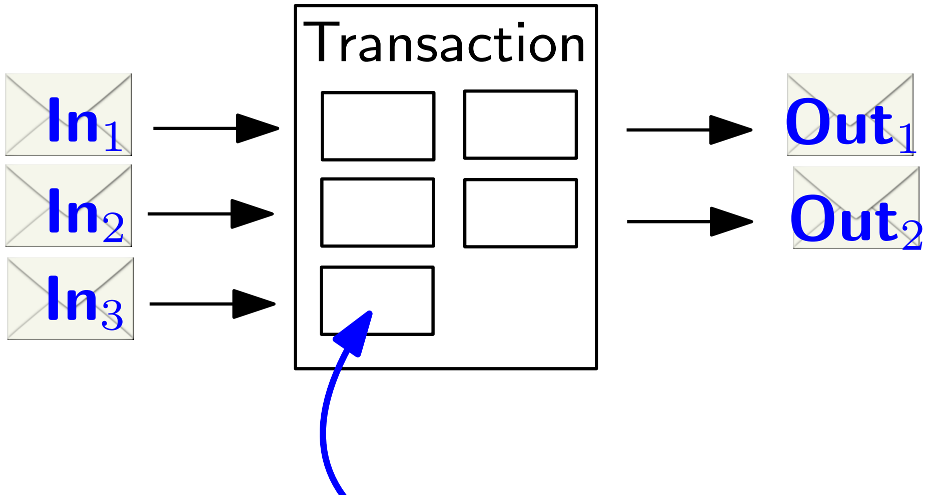
$$\sum \mathbf{Out} - \sum \mathbf{In} = 0$$

no more signatures!

But: sender knows sum of output r's

Mimblewimble

[Jedusor '16]



no more signatures!

secret key!

$$C = vH + rG, \quad \pi$$

$$\sum \textbf{Out} - \sum \textbf{In} = \mathbf{0}$$

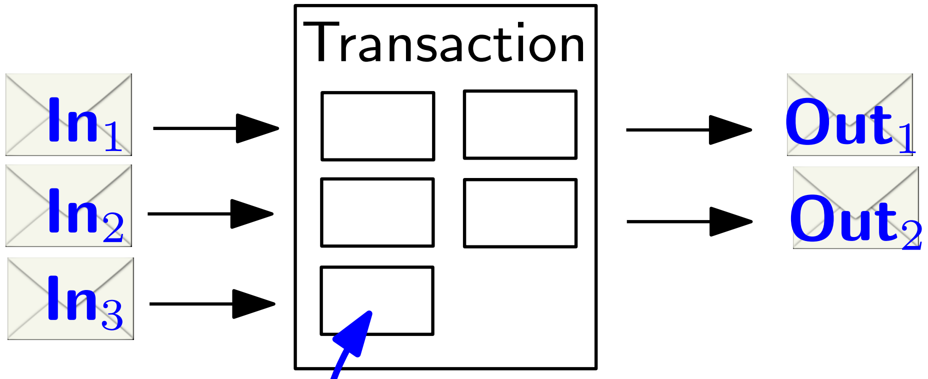
$$\begin{aligned}
&\sum C_i^{\text{out}} - \sum C_i^{\text{in}} \\
&= \sum (v_i^{\text{out}} H + r_i^{\text{out}} G) - \sum (v_i^{\text{in}} H + r_i^{\text{in}} G) \\
&= \underbrace{(\sum v_i^{\text{out}} - \sum v_i^{\text{in}})}_{\overset{!}{=} 0} H + \underbrace{(\sum r_i^{\text{out}} - \sum r_i^{\text{in}})}_{=: x} G
\end{aligned}$$

Mimblewimble

[Jedusor '16]

secret key!

$$C = vH + rG, \quad \pi$$



In$_1$

In$_2$

In$_3$

Transaction

Out$_1$

Out$_2$

no more signatures!

$$\sum \textbf{Out} - \sum \textbf{In}$$

$$= 0H + xG$$

$$\sum C_i^{\text{out}} - \sum C_i^{\text{in}}$$

$$= \sum (v_i^{\text{out}} H + r_i^{\text{out}} G) - \sum (v_i^{\text{in}} H + r_i^{\text{in}} G)$$

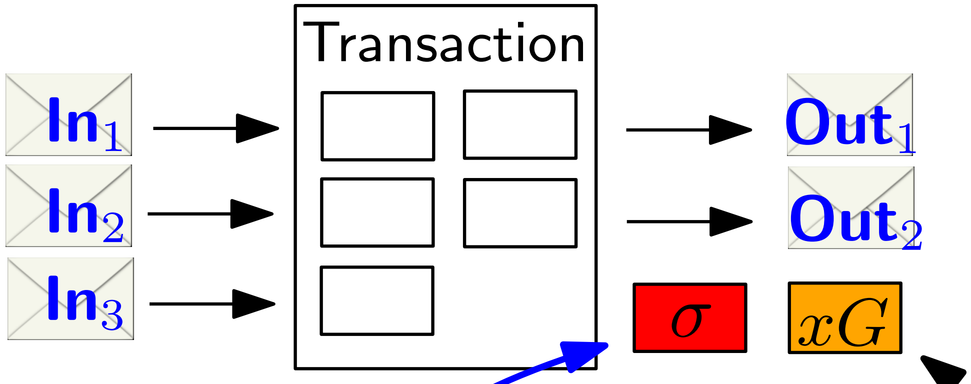
$$= \underbrace{(\sum v_i^{\text{out}} - \sum v_i^{\text{in}})}_{\overset{!}{=} 0} H + \underbrace{(\sum r_i^{\text{out}} - \sum r_i^{\text{in}})}_{=: x} G$$

Mimblewimble

[Jedusor '16]

secret key!

$$C = vH + rG, \quad \pi$$



$\mathbf{In}_1$

$\mathbf{In}_2$

$\mathbf{In}_3$

Transaction

$\mathbf{Out}_1$

$\mathbf{Out}_2$

σ

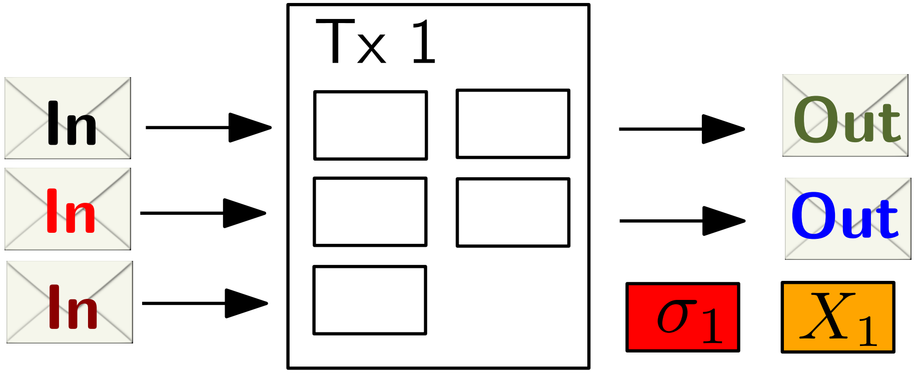
xG

$$\sum \mathbf{Out} - \sum \mathbf{In} = 0H + xG$$

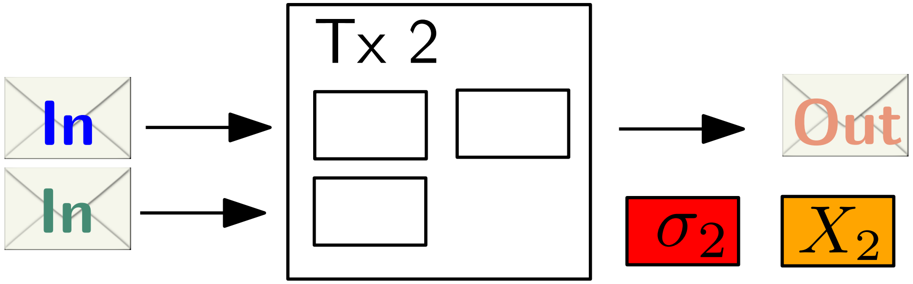
one signature

“proves” that $\overbrace{\sum \text{Out} - \sum \text{In}}$ is commitment to 0

Mimblewimble



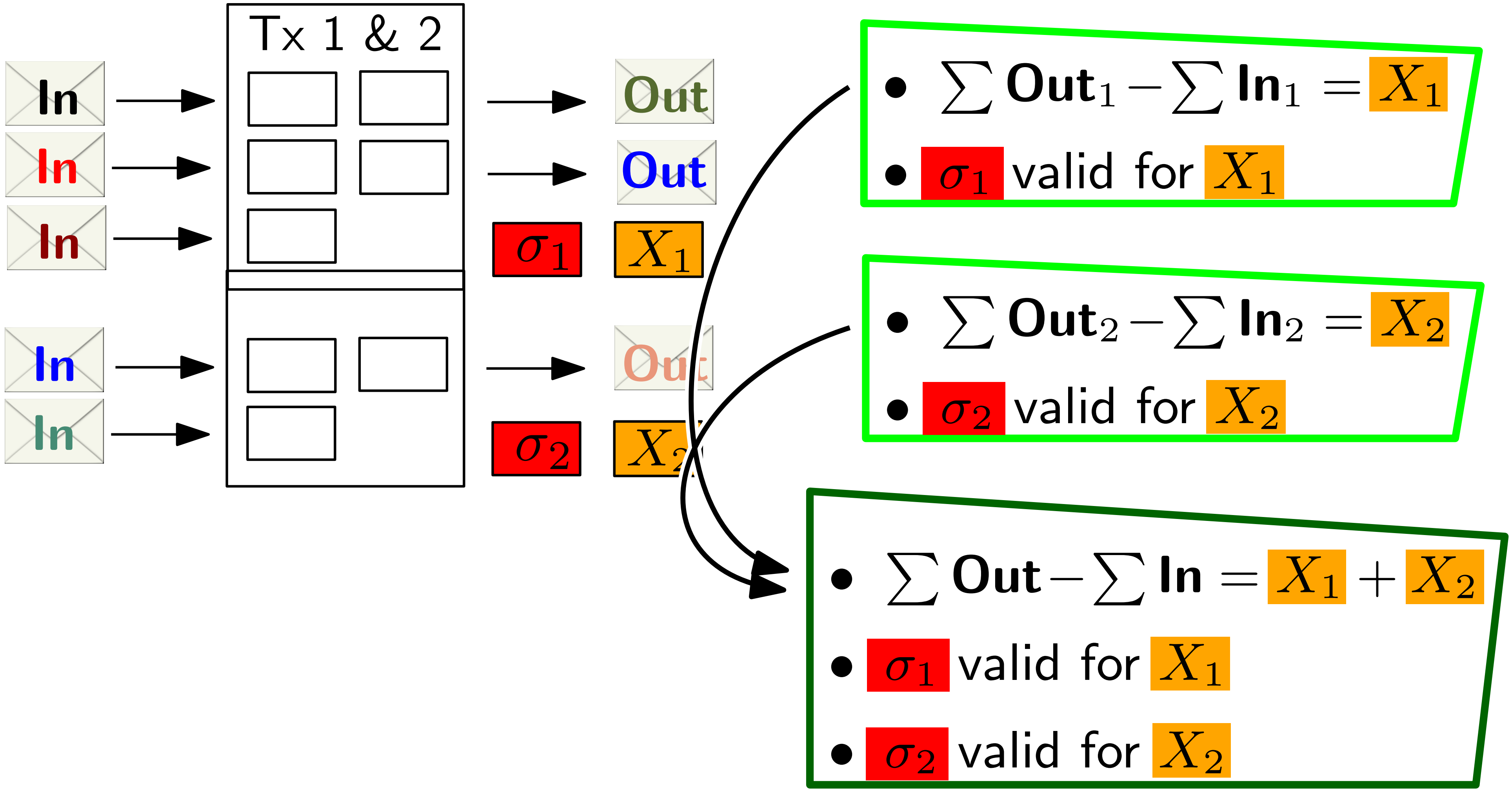
- $\sum \textbf{Out}_1 - \sum \textbf{In}_1 = X_1$
- σ_1 valid for X_1



- $\sum \textbf{Out}_2 - \sum \textbf{In}_2 = X_2$
- σ_2 valid for X_2

Mimblewimble

Non-interactive CoinJoin



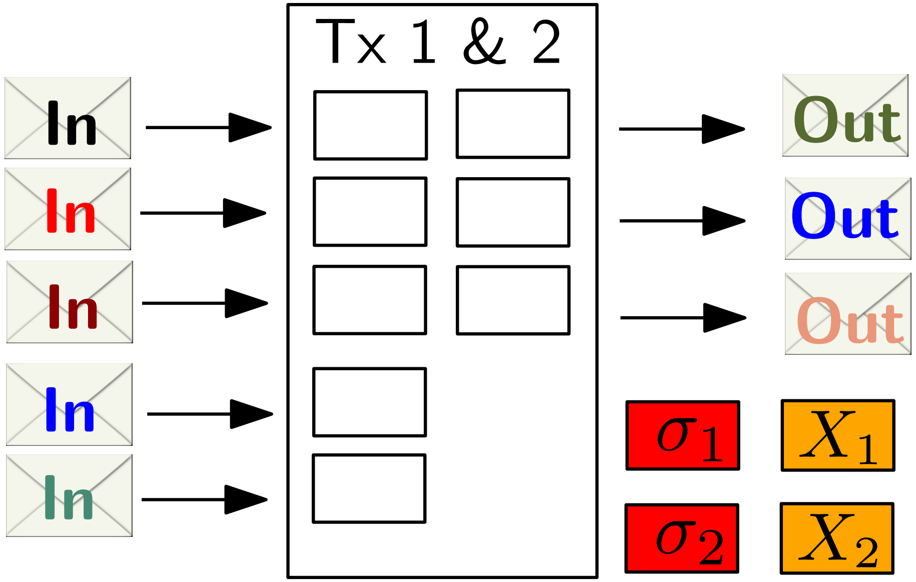
- $\sum$ **Out**$_1 - \sum$ **In**$_1 = X_1$
- σ_1 valid for X_1

- $\sum$ **Out**$_2 - \sum$ **In**$_2 = X_2$
- σ_2 valid for X_2

- $\sum$ **Out** $- \sum$ **In** $= X_1 + X_2$
- σ_1 valid for X_1
- σ_2 valid for X_2

Mimblewimble

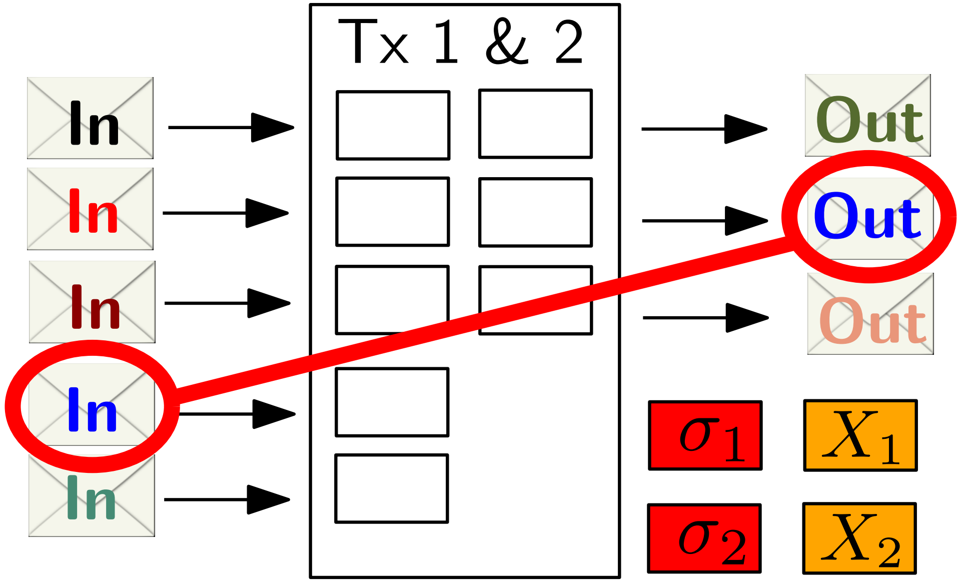
Non-interactive CoinJoin



- $\sum$ **Out** $-\sum$ **In** $= X_1 + X_2$
- σ_1 valid for X_1
- σ_2 valid for X_2

Mimblewimble

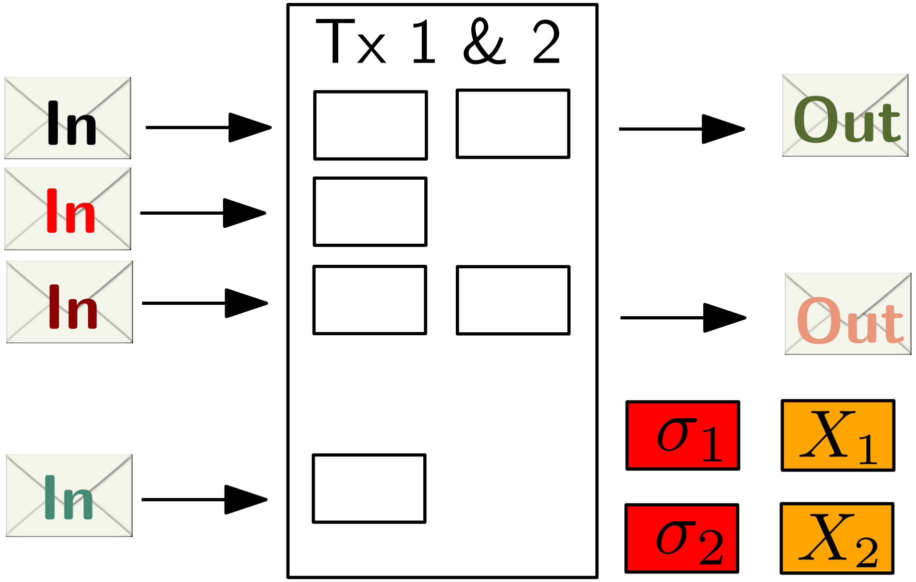
Cut-Through!



- $\sum$ **Out** $-\sum$ **In** $= X_1 + X_2$
- σ_1 valid for X_1
- σ_2 valid for X_2

Mimblewimble

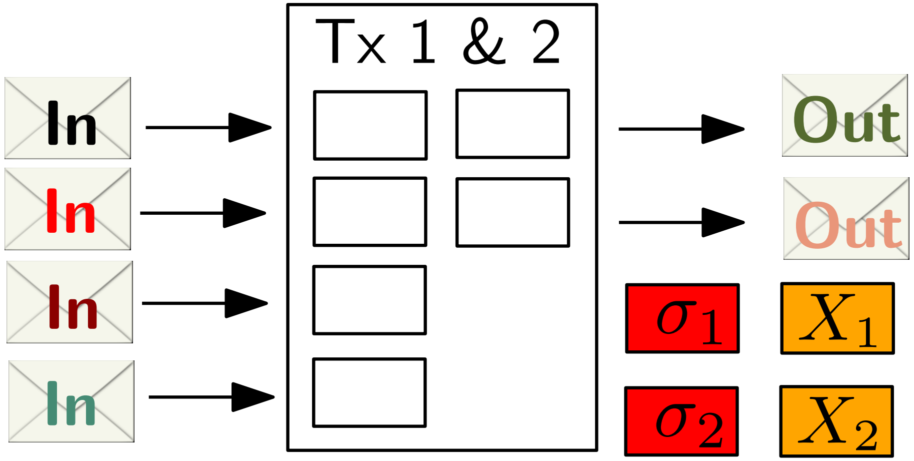
Cut-Through!



- $\sum$ **Out** $-\sum$ **In** $= X_1 + X_2$
- σ_1 valid for X_1
- σ_2 valid for X_2

Mimblewimble

Cut-Through!



- $\sum$ **Out** $-\sum$ **In** $= X_1 + X_2$
- σ_1 valid for X_1
- σ_2 valid for X_2

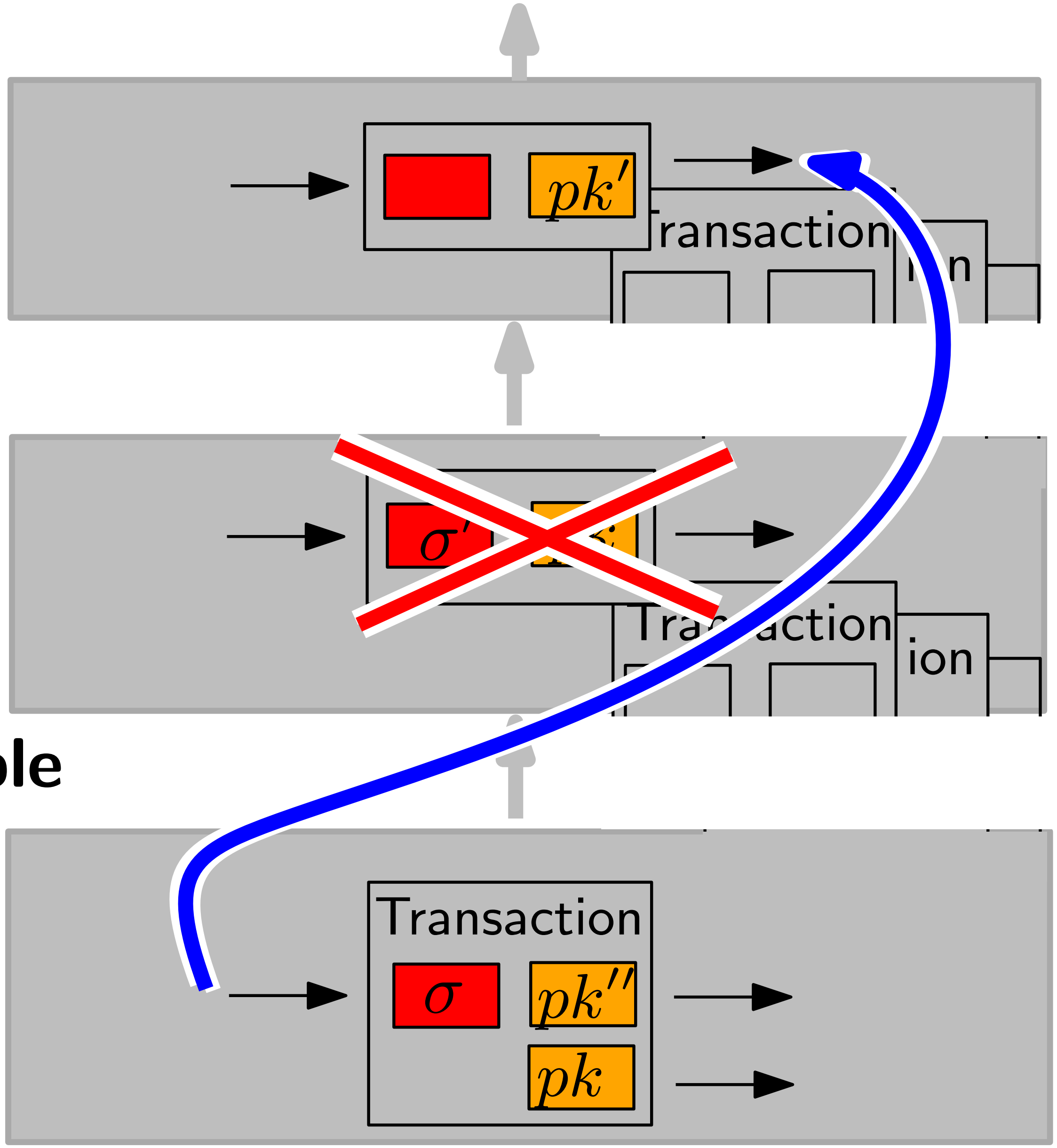
Scalability

"cut-through"

not possible
in Bitcoin:

σ' is needed
to verify validity

⇒ **Mimblewimble**



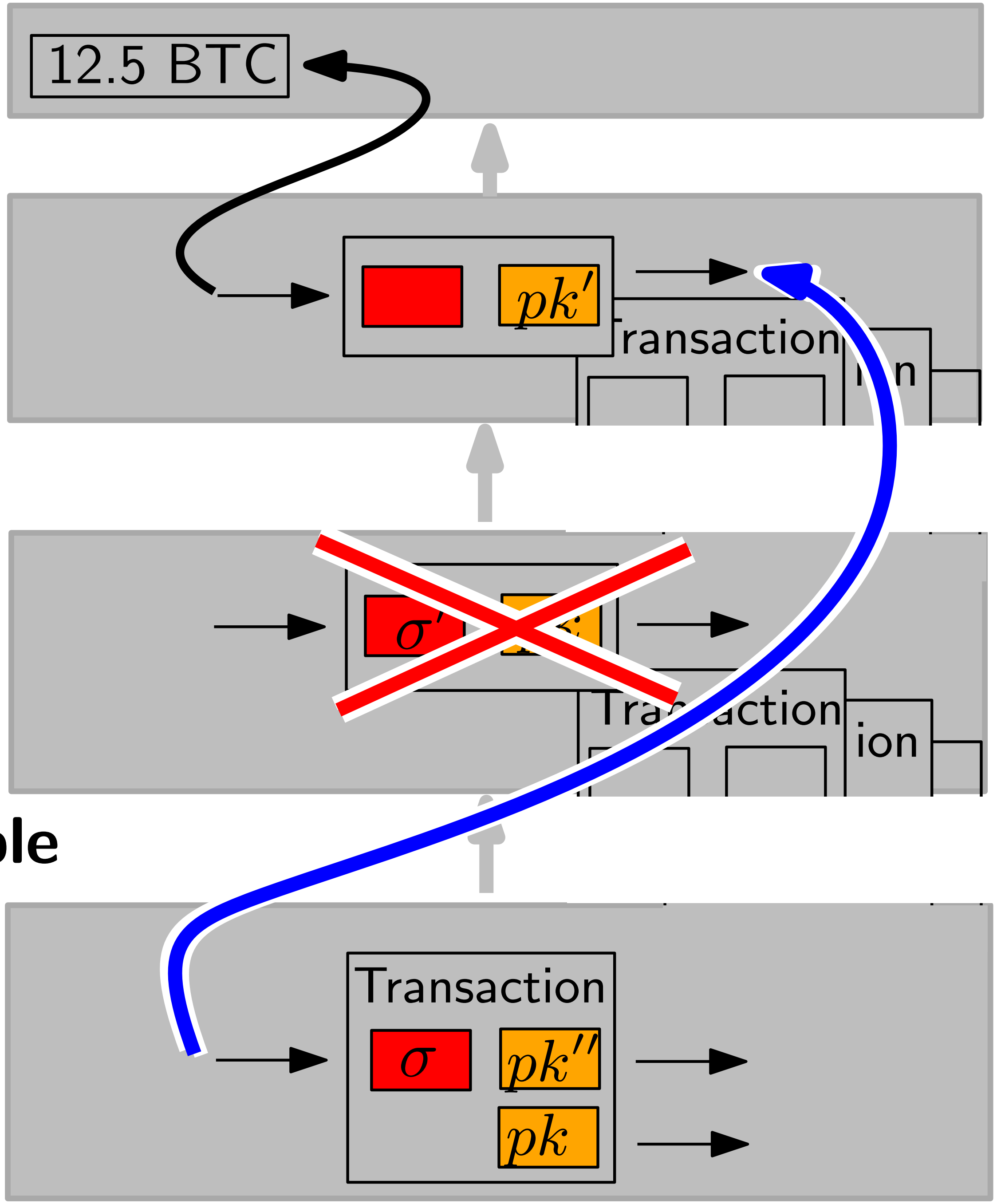
Scalability

“cut-through”

not possible in Bitcoin:

σ' is needed to verify validity

$\Rightarrow$ **Mimblewimble**



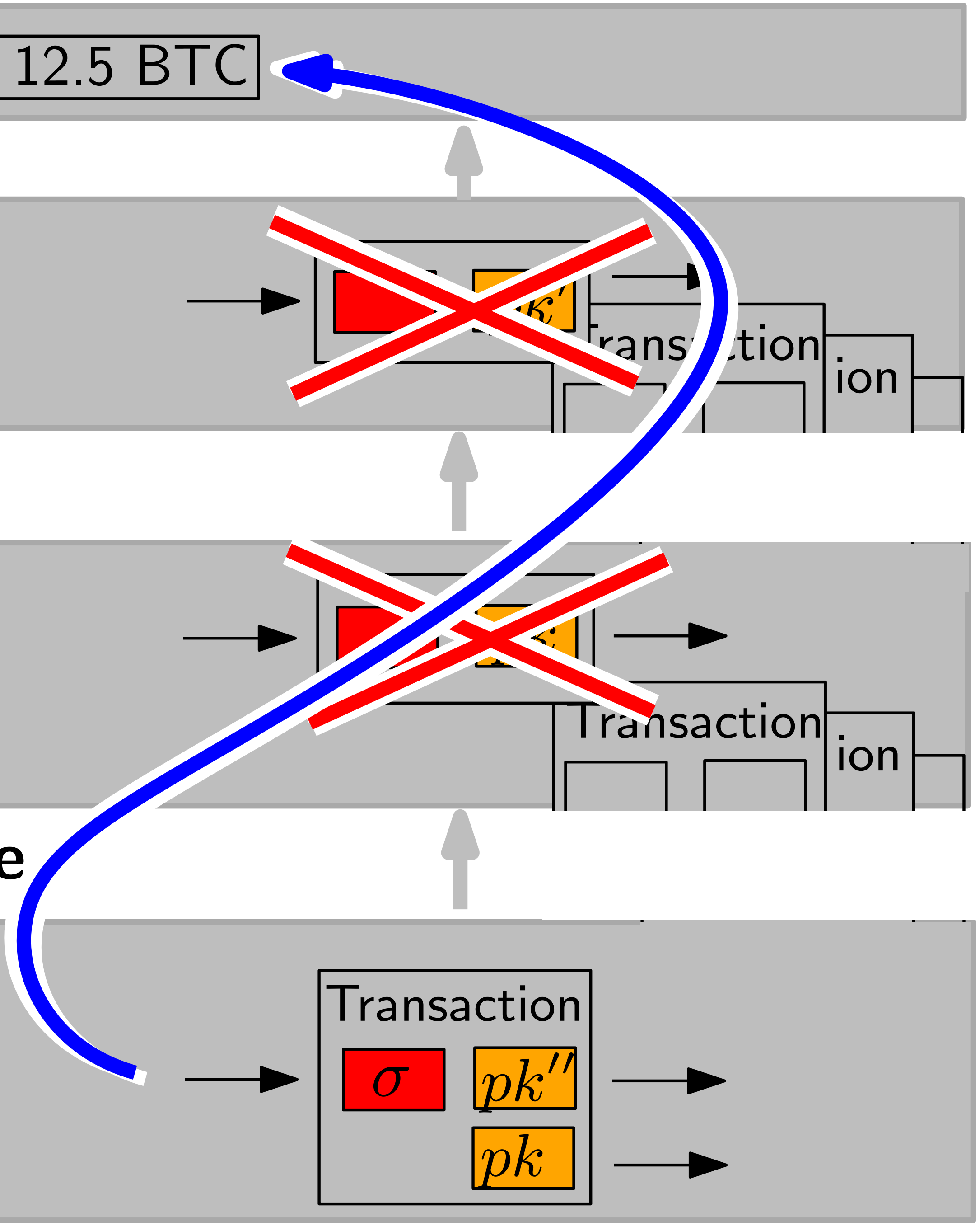
Scalability

“cut-through”

not possible
in Bitcoin:

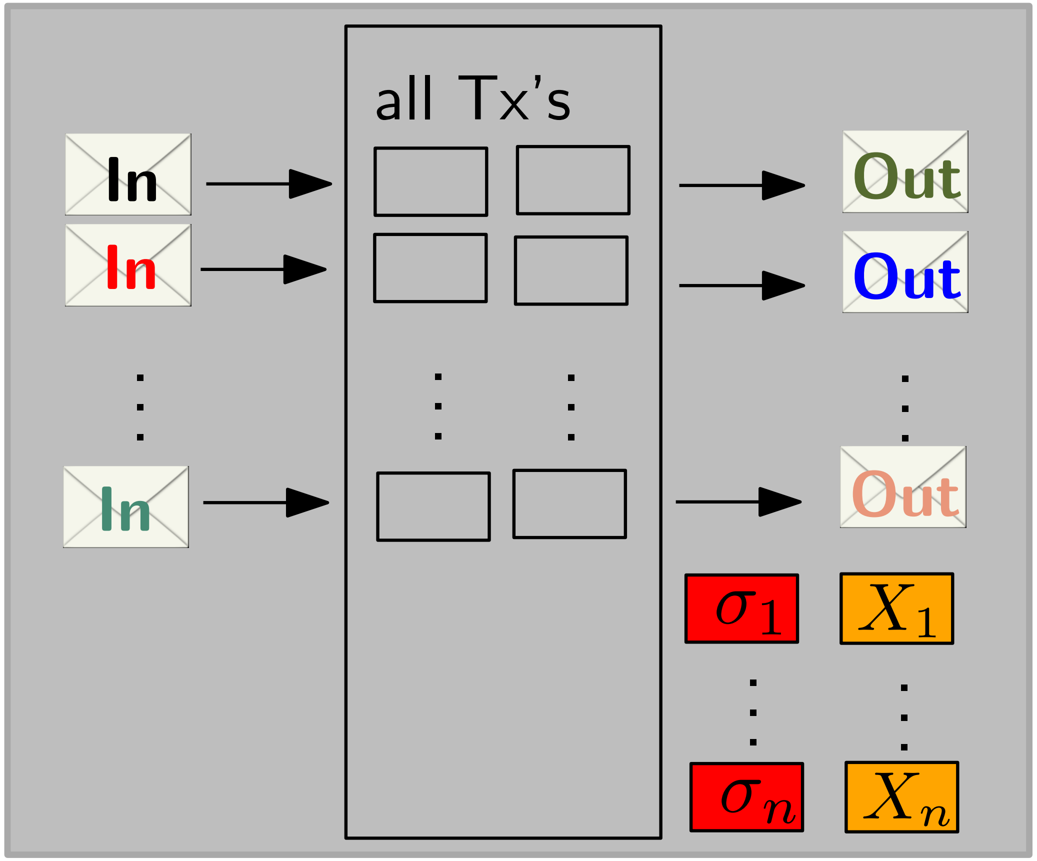
σ' is needed
to verify validity

⇒ **Mimblewimble**



Mimblewimble

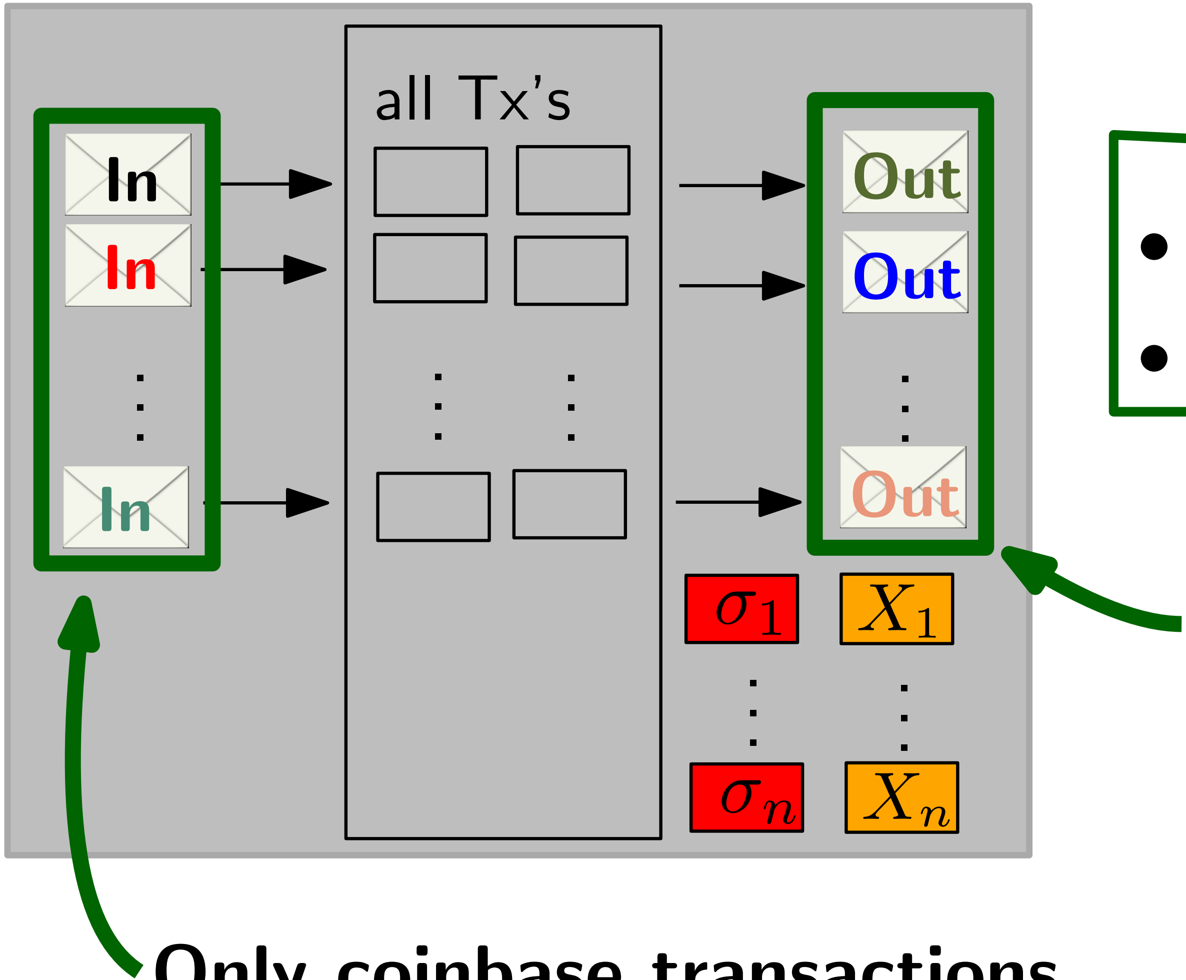
Cut through all transactions in blockchain



- $\sum$ **Out** $-\sum$ **In** $=\sum X_i$
- $\forall i$: σ_i valid for X_i

Mimblewimble

Cut through all transactions in blockchain



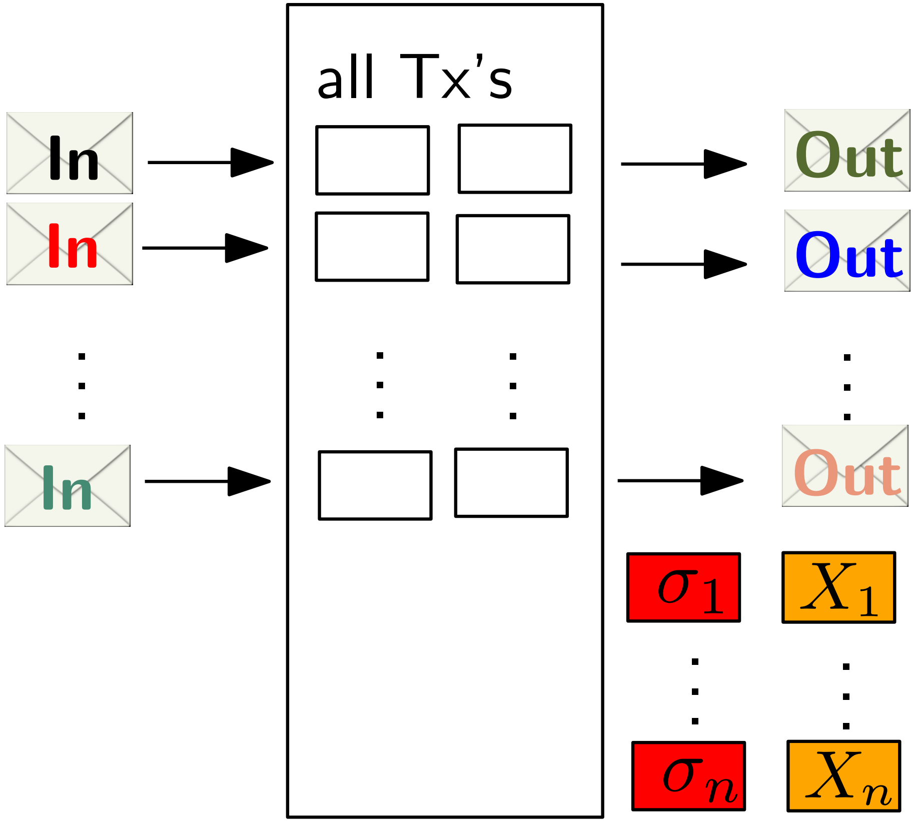
- $\sum$ **Out** $-\sum$ **In** $=\sum X_i$
- $\forall i : \sigma_i$ valid for X_i

UTXO set

Only coinbase transactions

Mimblewimble

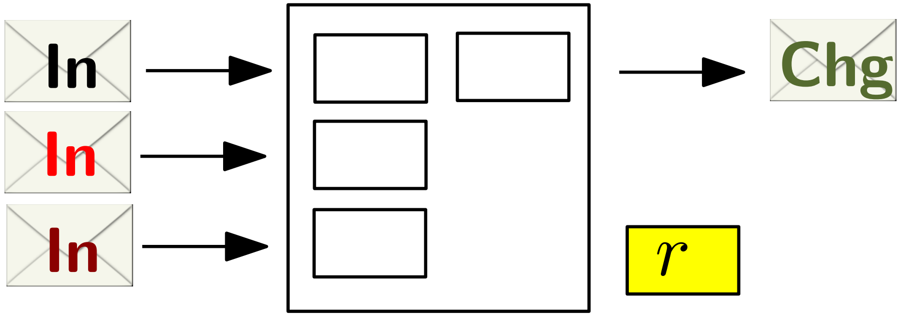
How to we actually make payments?



- $\sum$ **Out** $-\sum$ **In** $= \sum X_i$
- $\forall i : \sigma_i$ valid for X_i

Mimblewimble

How to we actually make payments?

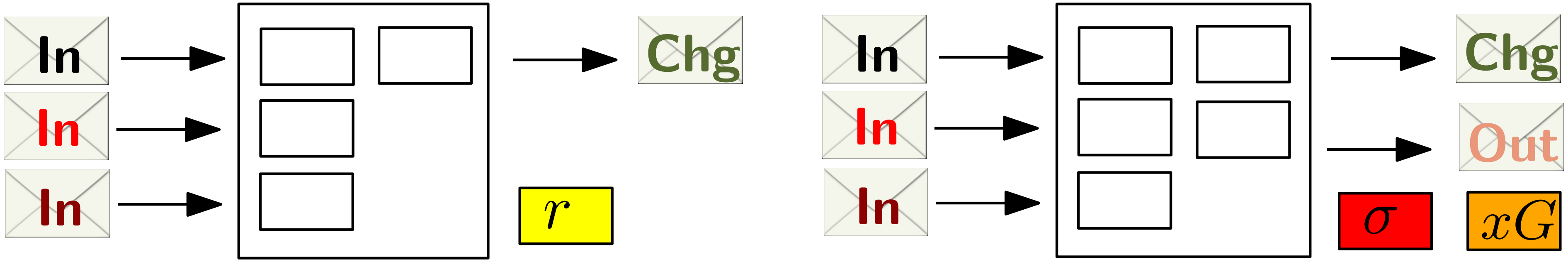


Original proposal. To pay p: [Jedusor '16]

- Sender
  - choose input coins worth $\sum v_i^{\text{in}} \geq p$
  - create change coins C_i^{chg} worth $\sum v_i^{\text{chg}} = \sum v_i^{\text{in}} - p$
  - send $r = \sum r_i^{\text{chg}} - \sum r_i^{\text{in}}$

Mimblewimble

How to we actually make payments?

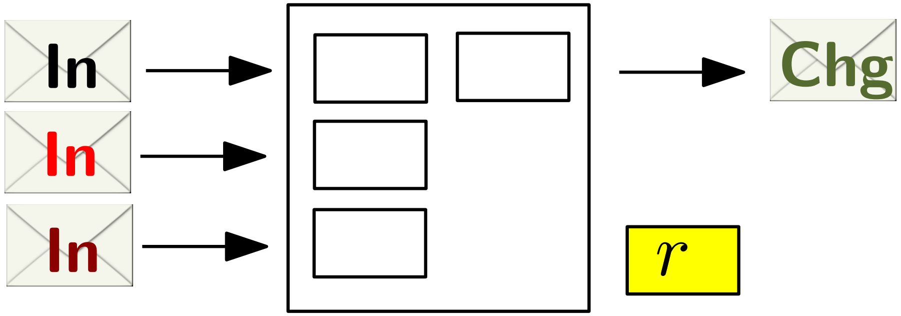


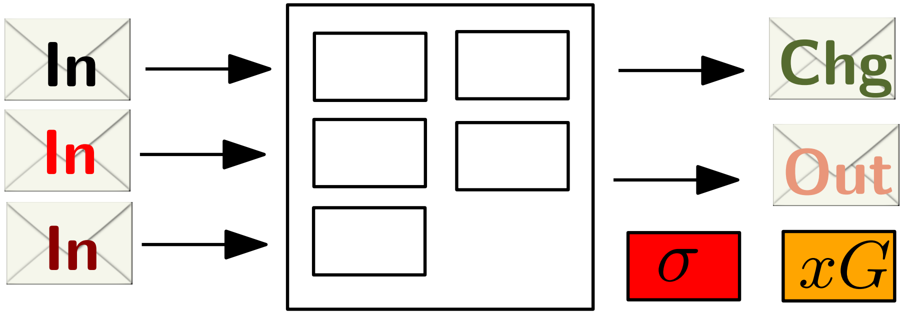
Original proposal. To pay p:

- Sender
  - choose input coins worth $\sum v_i^{\text{in}} \geq p$
  - create change coins C_i^{chg} worth $\sum v_i^{\text{chg}} = \sum v_i^{\text{in}} - p$
  - send $r = \sum r_i^{\text{chg}} - \sum r_i^{\text{in}}$
- Receiver
  - creates output coins C_i^{out} worth p
  - signs using $x = r + \sum r_i^{\text{out}}$

Mimblewimble

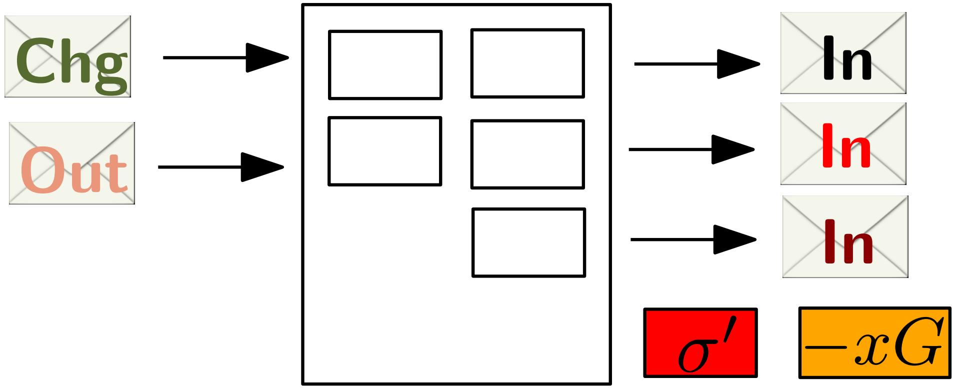
How to we actually make payments?





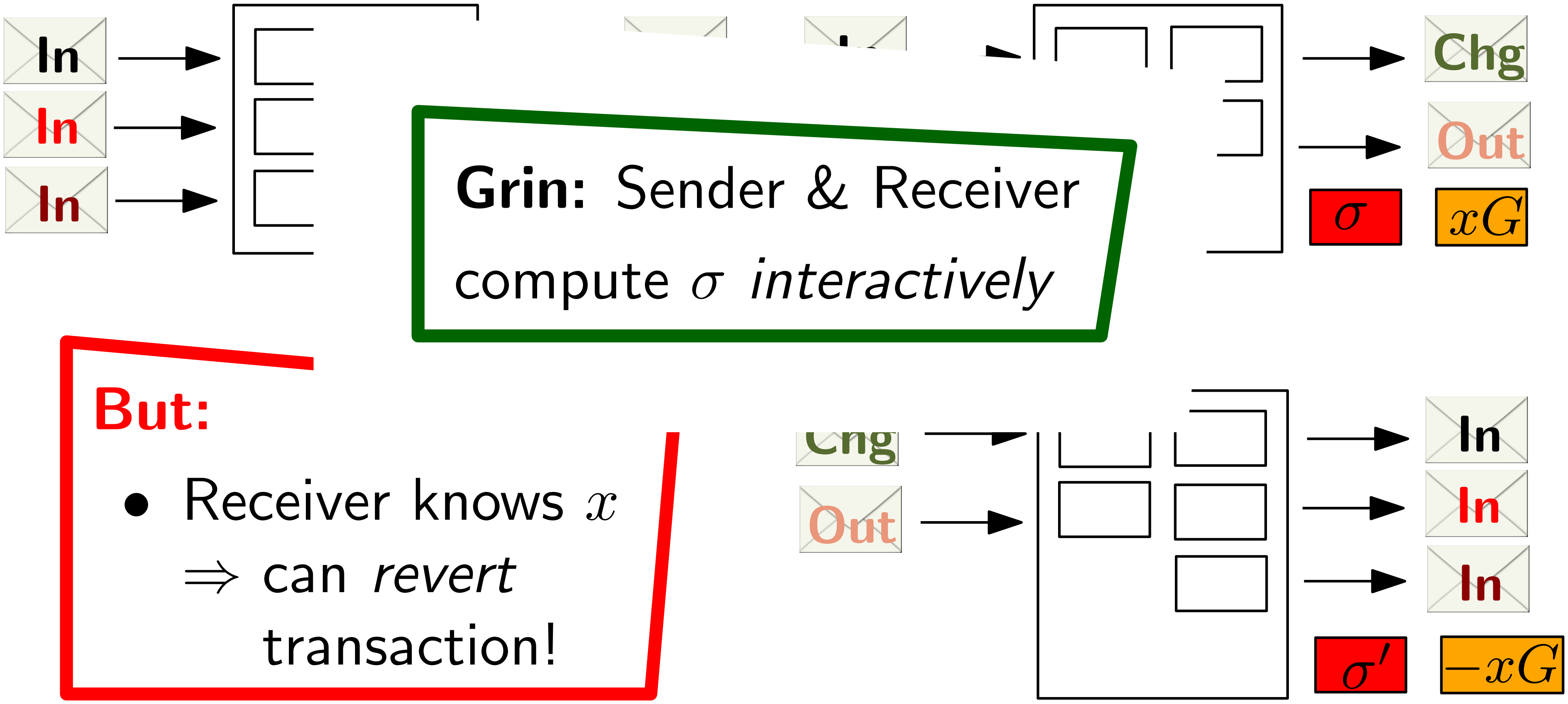
But:

- Receiver knows x
  $\Rightarrow$ can *revert* transaction!



Mimblewimble

How to we actually make payments?



Grin: Sender & Receiver compute σ *interactively*

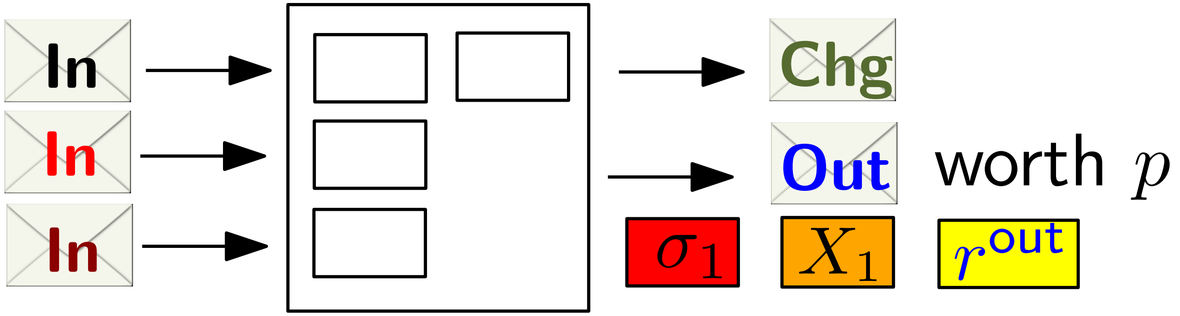
But:

- Receiver knows x
  $\Rightarrow$ can *revert* transaction!

Mimblewimble

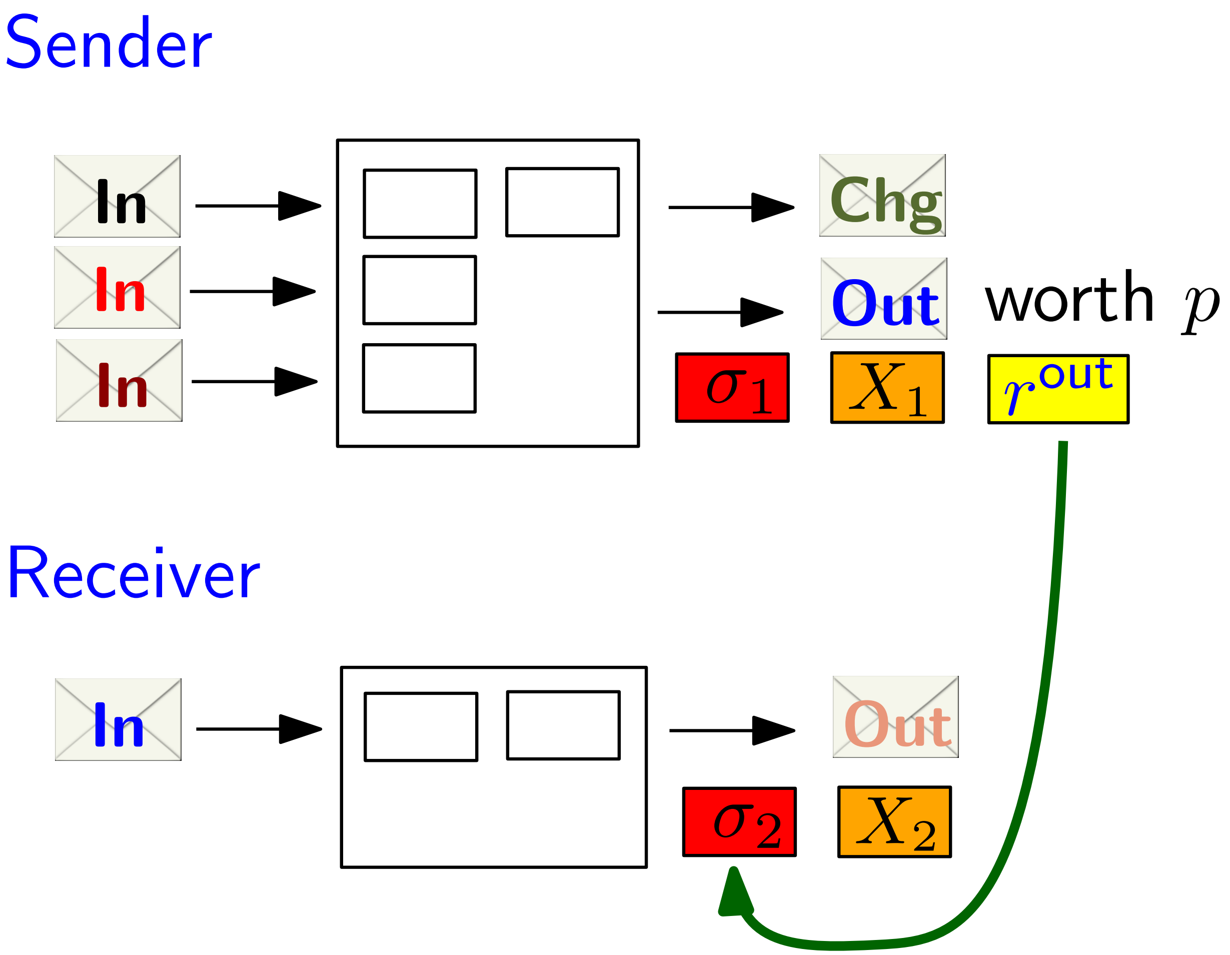
Our proposal: non-interactive!

Sender, to pay p, send:



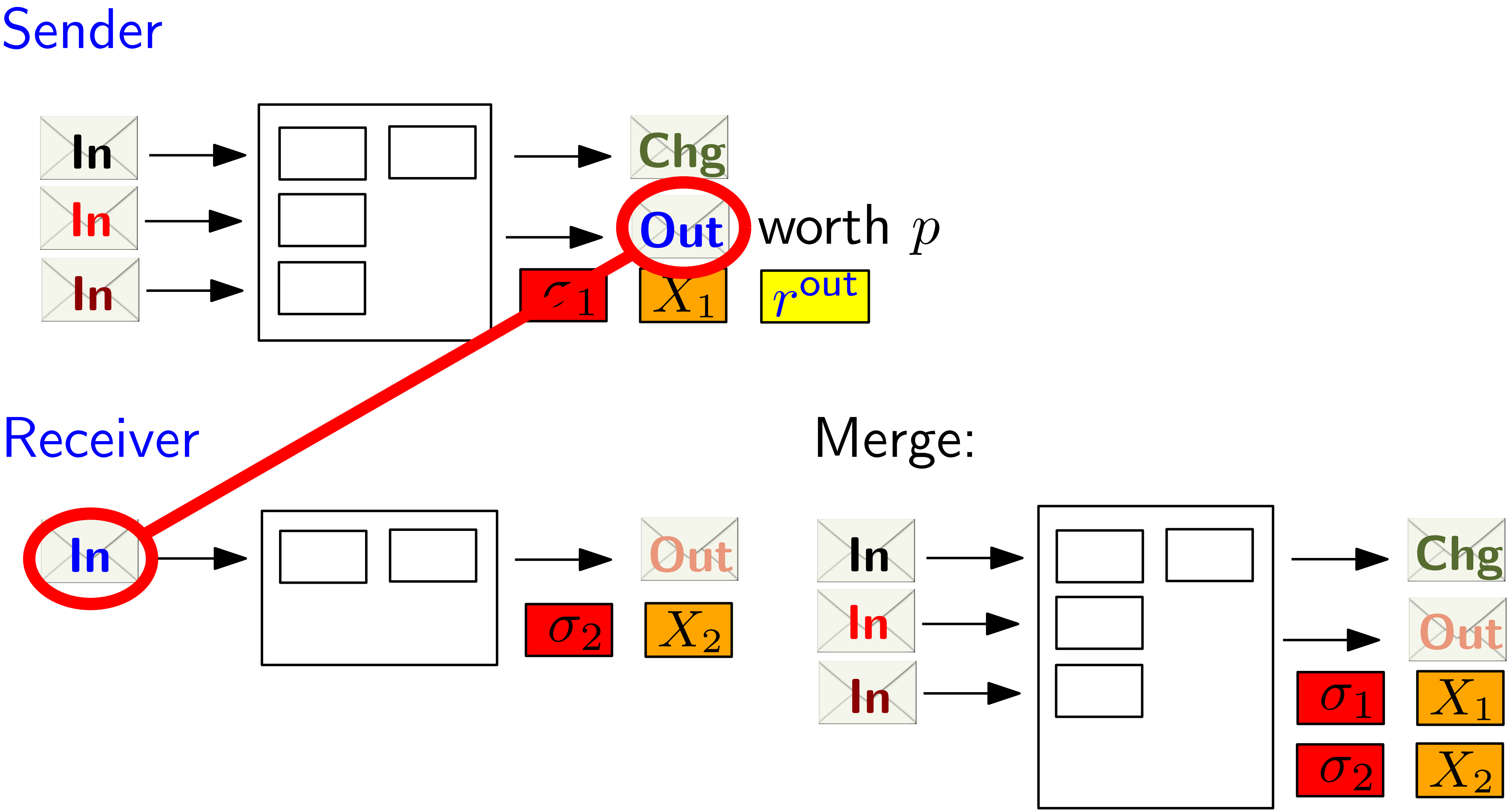
Mimblewimble

Our proposal: non-interactive!



Mimblewimble

Our proposal: non-interactive!



Mimblewimble

Our contributions: to appear at EUROCRYPT'19

- **Formal security models:**
  - inflation-resistance
  - coin-theft-resistance
  - confidential amounts

Mimblewimble

Our contributions: to appear at EUROCRYPT'19

- **Formal security models:**
  - inflation-resistance
  - coin-theft-resistance
  - confidential amounts
- **Abstraction of Mimblewimble** from:
  - homomorphic commitments
  - compatible signatures
  - simulation-extractable NIZK range proofs

Mimblewimble

Our contributions: to appear at EUROCRYPT'19

- **Formal security models:**
  - inflation-resistance
  - coin-theft-resistance
  - confidential amounts
- **Abstraction of Mimblewimble** from:
  - homomorphic commitments
  - compatible signatures
  - simulation-extractable NIZK range proofs

...satisfying joint security

Mimblewimble

Our contributions: to appear at EUROCRYPT'19

- **Formal security models:**
  - inflation-resistance
  - coin-theft-resistance
  - confidential amounts
- **Abstraction of Mimblewimble** from:
  - homomorphic commitments
  - compatible signatures

  ] . . . satisfying joint security
  - simulation-extractable NIZK range proofs
- **Proof** that abstraction satisfies model

Mimblewimble

Our contributions: to appear at EUROCRYPT'19

- **Formal security models:**
  - inflation-resistance
  - coin-theft-resistance
  - confidential amounts
- **Abstraction of Mimblewimble** from:
  - homomorphic commitments
  - compatible signatures

  ] . . . satisfying joint security
  - simulation-extractable NIZK range proofs
- **Proof** that abstraction satisfies model
- **Instantiations:** proof that
  - Pedersen + Schnorr
  - Pedersen + (aggregate) BLS

  ] . . . satisfy joint security