The security of Mimblewimble

Georg Fuchsbauer

joint work with

Michele Orrù and Yannick Seurin

F, Orrù, Seurin: Aggregate cash systems: A cryptographic investigation of Mimblewimble. EUROCRYPT’19

F, Orrù: Non-interactive Mimblewimble transactions, revisited. (eprint 2022/265)
What is it?

- **Cryptocurrency scheme**
  - **Privacy** (all amounts hidden; input/output relation blurred)
  - **Scalability** (forget about spent tx’s)

- proposed by “Tom Elvis Jedusor” in 2016

- uses ideas from Gregory Maxwell

- further developed by Andrew Poelstra
## Applications

Implemented by several cryptocurrencies (...2021):

<table>
<thead>
<tr>
<th>Cryptocurrency</th>
<th>Rank</th>
<th>Category</th>
<th>Watchlists</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grin</td>
<td>851</td>
<td>Coin</td>
<td>13,841</td>
</tr>
<tr>
<td>Beam</td>
<td>635</td>
<td>Coin</td>
<td>30,698</td>
</tr>
<tr>
<td>MimbleWimbleCoin</td>
<td>565</td>
<td>Coin</td>
<td>4,210</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Market Cap</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>$6,705,865</td>
<td>7.42%</td>
</tr>
<tr>
<td>$15,762,711</td>
<td>3.72%</td>
</tr>
<tr>
<td>$21,444,531</td>
<td>7.26%</td>
</tr>
</tbody>
</table>
Non-interactive TXs

Main **drawback**: transactions are *interactive*

2020: David Burkett, Gary Yu: **Non-interactive** transactions

2021: Fixed by Burkett, F, Orrù
    
    Analyzed by F, Orrù

2022: Implemented in **Litecoin**
Non-interactive TXs

Main **drawback**: transactions are *interactive*

2020: David Burkett, Gary Yu: **Non-interactive**

2021: Fixed by Burkett, F, Orr` u
Analyzed by F, C

2022: Implemented in **Litecoin**
- **Transactions**

  - 2 BTC → 6 BTC
  - 2 BTC → 1 BTC
  - 3 BTC → 1 BTC

- **Blockchain**

  - 6 BTC → 2 BTC
  - 6 BTC → 4 BTC
Bitcoin

• Reference to previous output

- 2 BTC → 6 BTC
- 2 BTC → 1 BTC
- 3 BTC → 4 BTC
- 6 BTC → 2 BTC
Unspent transaction outputs (UTXO’s) = existing money in system

Bitcoin
• Coinbase transaction

Bitcoin

6.25 BTC

2 BTC → 6 BTC
2 BTC → 1 BTC
3 BTC →

6.25 BTC

2 BTC → 1 BTC
2 BTC → 6 BTC

6.25 BTC

In
Out

In
Out

In
Out

In
Out

6 BTC

0.25 BTC
A Bitcoin transaction is depicted with two transactions: the first with a signature \( \sigma \) under public key \( pk \) on transaction \( tx \). The second transaction sends 6 BTC to another transaction that receives 2 BTC with a signature under \( pk' \) and 4 BTC under \( pk'' \).
Security

- signatures ⇒ no theft
- balancedness of tx’s checkable ⇒ no illegal creation

σ is signature under pk on tx
**Drawbacks**

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

Blockchain size: 
> 400 GB

Size of UTXO set: 
< 5 GB
Scalability

Diagram showing the scalability of a transaction system with labels for keys and transactions.
“cut-through”

not possible in Bitcoin:

$\sigma'$ is needed to verify validity

$\Rightarrow$ Mimblewimble
Anonymity

Transaction

1 BTC → In → Out → 3 BTC
3 BTC → In → Out → 3 BTC
2 BTC → In → Out → 3 BTC

?
Anonymity

• **CoinJoin** [Maxwell’13]
  – no *link* between inputs and outputs
  – join many transactions?
  – in Bitcoin: *only interactively*, since all inputs must sign tx
Anonymity

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

(by default in MONERO)
Anonymity

How can we get

- Confidential transactions (check balancedness)
- Coin-join (non-interactively)
- Cut-through (post-confirmation)

while maintaining verifiability?
Anonymity

- Confide:
  - hide tľ
  - not co
  - balanc

Mimblewimble
Discrete logarithms

- Finite group (of prime order) \((G, +)\)
  - generator \(G\)
  - \(xG := G + \ldots + G\) \(x\) times

- **Discrete logarithm** problem:
  - given \(G, H \in G\)
  - 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”

- **hiding:** commitment hides $v$

- **binding:** Alice can open commitment only to one value
**Pedersen commitment**

**Commitment**
- “digital envelope”

\[
\text{Commit: } v
\]

\[
\text{Open: } vH + rG
\]

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

\[
G, H \in \mathbb{G}
\]

pick random \( r \)

reveal \( v \) and \( r \)
Commitment

- “digital envelope”

Pedersen commitment

\[ G, H \in \mathbb{G} \]

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 \( v \):

\[ r = \log_G C - v \cdot \log_G H \]
Commitment

- “digital envelope”

Pedersen commitment

\[ G, H \in \mathbb{G} \]

pick random \( r \)

\[ C := vH + rG \]

reveal \( v \) and \( r \)

- **binding:** assume Alice finds \( v \neq v', r, r' \) with

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

\( \Rightarrow \) Alice solved discrete log problem!
Pedersen commitment

## Commitment

- “digital envelope”

![Commitment Diagram]

- commitments are **homomorphic**:

\[
\text{Com}(v_1; r_1) + \text{Com}(v_2; r_2) = (v_1H + r_1G) + (v_2H + r_2G) \\
= (v_1 + v_2)H + (r_1 + r_2)G \\
= \text{Com}(v_1 + v_2; r_1 + r_2)
\]

*e.g.*: \(\text{Com}(1; 5) + \text{Com}(1; 10) - \text{Com}(2, 15) = 0\)
Confidential Transactions

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

\[ C = vH + rG \]

\[ \sum \text{Out} - \sum \text{In} = 0 \]
Confidential Transactions

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

\[ C = vH + rG \]

\[ \sum \text{Out} - \sum \text{In} = 0 \]

Range proofs
- add proofs that committed values are in \( \in [0, 2^{64}] \)
Confidential Transactions

Confidential transaction

\[ C = vH + rG, \quad \pi \]

\[ \sum \text{Out} - \sum \text{In} = 0 \]

Signatures ⇒
- no non-interactive CoinJoin
- no Cut-Through
Mimblewimble

\[ C = vH + rG, \quad \pi \]

\[ \sum \text{Out} - \sum \text{In} = 0 \]

But: sender knows sum of output \( r \)'s

[Jedusor '16]
Mimblewimble

In1 \rightarrow \text{Transaction} \rightarrow \text{Out1} \rightarrow \text{Out2} \rightarrow \text{In2} \rightarrow \text{In3}

\[ C = vH + rG, \quad \pi \]

\[ \sum \text{Out} - \sum \text{In} = 0H + xG \]

\[
\sum C_{i}^{\text{out}} - \sum C_{i}^{\text{in}} \\
= \sum (v_{i}^{\text{out}} H + r_{i}^{\text{out}} G) - \sum (v_{i}^{\text{in}} H + r_{i}^{\text{in}} G) \\
= (\sum v_{i}^{\text{out}} - \sum v_{i}^{\text{in}}) H + (\sum r_{i}^{\text{out}} - \sum r_{i}^{\text{in}}) G \\
\equiv 0
\]

no more signatures!

secret key!
Mimblewimble

\[
C = vH + rG, \quad \pi
\]

\[
\sum \text{Out} - \sum \text{In} = 0H + xG
\]

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

secret key!
Mimblewimble

\[ \sum \text{Out}_1 - \sum \text{In}_1 = X_1 \]
\[ \sigma_1 \text{ valid for } X_1 \]

\[ \sum \text{Out}_2 - \sum \text{In}_2 = X_2 \]
\[ \sigma_2 \text{ valid for } X_2 \]
Mimblewimble

Non-interactive CoinJoin

\[ \sum \text{Out}_1 - \sum \text{In}_1 = X_1 \]
- \( \sigma_1 \) valid for \( X_1 \)

\[ \sum \text{Out}_2 - \sum \text{In}_2 = X_2 \]
- \( \sigma_2 \) valid for \( X_2 \)

\[ \sum \text{Out} - \sum \text{In} = X_1 + X_2 \]
- \( \sigma_1 \) valid for \( X_1 \)
- \( \sigma_2 \) valid for \( X_2 \)
Mimblewimble

\[ \sum \text{Out} - \sum \text{In} = X_1 + X_2 \]

• \( \sigma_1 \) valid for \( X_1 \)
• \( \sigma_2 \) valid for \( X_2 \)
Mimblewimble

Post-confirmation Cut-Through

• \( \sum \text{Out} - \sum \text{In} = X_1 + X_2 \)
• \( \sigma_1 \) valid for \( X_1 \)
• \( \sigma_2 \) valid for \( X_2 \)
Mimblewimble

**Post-confirmation Cut-Through**

\[ \sum \text{Out} - \sum \text{In} = X_1 + X_2 \]

- \( \sigma_1 \) valid for \( X_1 \)
- \( \sigma_2 \) valid for \( X_2 \)
Scalability

“cut-through”
Scalability

“cut-through”
Cut through all transactions in blockchain

\[ \sum \text{Out} - \sum \text{In} = \sum X_i \]

\[ \forall i : \sigma_i \text{ valid for } X_i \]

Only coinbase transactions

UTXO set
Mimblewimble

How are transactions actually created?

\[ \sum \text{Out} - \sum \text{In} = X \]
\[ \sigma \text{ valid for } X \]

signature under key \( \sigma \)

Use interactive protocol for signature under \( X_1 + X_2 \)
Mimblewimble

[FOS19]

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

- **Abstraction of Mimblewimble** from:
  - homomorphic commitments
  - compatible signatures
  - simulation-extractable NIZK range proofs

- **Proof** that abstraction satisfies model

- **Instantiations:** proof that
  - Pedersen + Schnorr
  - Pedersen + (aggregate) BLS

... satisfy joint security
Non-interactive TXs

Mimblewimble: receiver needs to interact with sender

Bitcoin: knowing receiver’s address, anyone can send money

Privacy? Bitcoin:
  • use every address only once $\rightarrow$ unlinkability
  • send address privately $\rightarrow$ requires interaction

Stealth addresses:
  • publish one address
  • receive unlinkable payments non-interactively

(by default in Monero)
Stealth addresses:

• publish one address
Stealth addresses:

- publish **one** address
- receive **unlinkable** payments
Stealth addresses

Bitcoin

$P = xG$

address: $H(P)$
Stealth addresses

- pick random $r$

$$P = \mathcal{H}(rA)G + B = (\mathcal{H}(aR) + b)G$$

$$= raG = aR$$

Diffie-Hellman shared key between $A$ and $R$

$$(A, B) = (aG, bG)$$
Stealth addresses

- pick random $r$

$R = rG$

$(A, B) = (aG, bG)$

$(a, b) = \mathcal{H}(aR) + b)G$

Stealth addresses
Non-interactive TXs

MW with non-interactive TXs

\[ \sum \text{Out} - \sum \text{In} = X \]

- \( \sigma \) valid for \( X \)
- rangeproofs valid

stealth addresses for outputs

\((A_1, B_1)\)
\((A_2, B_2)\)
Non-interactive TXs

MW with non-interactive TXs

\[ \sum \text{Out} - \sum \text{In} = X \]
\[ \sigma \text{ valid for } X \]
\[ \text{rangeproofs valid} \]

one-time addresses
“transfer keys”

(A₁, B₁)
(A₂, B₂)
### Non-interactive TXs

#### MW with non-interactive TXs

<table>
<thead>
<tr>
<th>$T_x$</th>
<th>$\text{In}$</th>
<th>$\text{In}$</th>
<th>$\text{In}$</th>
<th>$\text{Out}$</th>
<th>$\text{Out}$</th>
<th>$\sigma$</th>
<th>$X$</th>
</tr>
</thead>
</table>

- $\sum \text{Out} - \sum \text{In} = X$
- $\sigma$ valid for $X$
- Rangeproofs valid

**Derive $sk$ for $P_1$**

**Derive opening for $C$**

$(A_1, B_1)$

$(A_2, B_2)$
Non-interactive TXs

MW with non-interactive TXs

\[ \sum \text{Out} - \sum \text{In} = X \]

\[ \sigma \text{ valid for } X \]

\[ \text{rangeproofs valid} \]

But: \( \sigma \) cannot sign \( T_X \)

\( \leftarrow \text{CoinJoin, anonymity} \)
Non-interactive TXs

MW with non-interactive TXs

\[ \sum \text{Out} - \sum \text{In} = X \]

\[ \sigma \text{ valid for } X \]

\[ \text{rangeproofs valid} \]

\[ (A_1, B_1) \]

\[ (A_2, B_2) \]

But: no “authentication” of outputs
Non-interactive TXs

MW with non-interactive TXs

<table>
<thead>
<tr>
<th>$P'_1$</th>
<th>$P'_2$</th>
<th>$P'_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>$\sigma_2$</td>
<td>$\sigma_3$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_1$</td>
<td>$P_1$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_2$</td>
<td>$P_2$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sigma$</td>
<td>$X$</td>
<td>$\sigma_Y$</td>
</tr>
</tbody>
</table>

sig under one-time key $P'_3$ on input

- $\sum \text{Out} - \sum \text{In} = X$
- $\sigma$ valid for $X$
- rangeproofs valid
- verify $\sigma_i$'s
- $\sum R_i - \sum P'_i = Y$
- $\sigma_Y$ valid for $Y$

But: miner could just change $P$
Non-interactive TXs

MW with non-interactive TXs

\[ \sum \text{Out} - \sum \text{In} = X \]
\[ \sigma \text{ valid for } X \]
\[ \text{rangeproofs valid} \]
\[ \sum R_i - \sum P_i' = Y \]
\[ \sigma_Y \text{ valid for } Y \]
\[ \text{verify } \hat{\sigma}_i \text{'s} \]

But: still subtle attacks

sig under one-time key $P_3'$ on input

sig under $R_2$ on $P_2$
Non-interactive TXs

[FO22]

- **Fixing** scheme with Burkett
- **Prove** properties
  - inflation-resistance
  - coin-theft-resistance
  - transaction-binding
  - transaction-privacy

**assuming**
- hardness of computing discrete logarithms
  (and DDH for privacy)
- range proofs are extractable (and zero-knowledge)
- Schnorr is *simulation-sound* proof of knowledge of $sk$