Concurrency and Privacy with Payment Channel Networks

Giulio Malavolta†, Pedro Moreno-Sanchez‡, Aniket Kate‡, Matteo Maffei*, and Srivatsan Ravi§

†Friedrich-Alexander-University
‡Purdue University
*TU Vienna
Bitcoin Scalability Issues

- < 10 transactions per second
- > 135 GB of memory required
- No micropayment (high fees)
Payment Channels

- Enable multiple payments between two users without committing every single payment to the blockchain

### Blockchain Transactions

- **Alice**: 5
- **Bob**: 3
- **Alice**: 2

### Off-chain Payments

- **Alice**: 5
- **Bob**: 3
- **Alice**: 2

**OPEN CHANNEL**

**CLOSE CHANNEL**
Payment Channel Networks (PCN)

- Each payment channel requires to deposit bitcoins
- Impractical to open a channel with every other user

![Diagram showing Alice, Bob, and Cat with transactions of 1 BTC to Bob and 1 BTC to Cat]
Hash Time-Lock Contracts

- Hash-Time Lock Contract (HTLC) enables conditional payments between two users

\[
\text{HTLC}(\text{Alice, Bob, 1, } y, 30): \\
\text{Pay Bob 1 BTC iff Bob shows some x such that } H(x) = y, \text{ before 30 days}
\]

Alice \rightarrow Bob

Blockchain Transactions

\[H(x) = y\]
The Lightning Network

- Multiple “chained” HTLC enable multi-hop payments in the presence of untrusted intermediaries
Multiple “chained” HTLC enable multi-hop payments in the presence of untrusted intermediaries
The Lightning Network

- Multiple “chained” HTLC enable multi-hop payments in the presence of untrusted intermediaries

```
HTLC(Alice, Bob, 1, y, 30)  HTLC(Bob, Cat, 1, y, 29)
```

```
x : H(x) = y
```
Multiples “chained” HTLC enable multi-hop payments in the presence of untrusted intermediaries
Multiple “chained” HTLC enables multi-hop payments in the presence of untrusted intermediaries

Bob does not gain or lose coins
Contributions

- Definition of security and privacy properties for PCNs
- Privacy analysis of PCNs and solution (Fulgor)
- Concurrency analysis of PCNs and solution (Rayo)
- Prototype implementation
Security Properties

- Our model highlights two main security properties:
  - Balance Security:
    \[
    \Pr[\text{after payment}] < \text{negl}
    \]
    \[\Delta = 10\]
  - Serializability:
Privacy Properties

• Our model highlights two privacy properties

• (Off-path) value privacy:

• (On-path) relationship anonymity:
Privacy in PCNs: Challenge?

- Off-chain payments  =>  Privacy-preserving payments

Blockchain Transactions

HTLC(Alice, Bob, 1, y, 30)  
HTLC(Bob, Cat, 1, y, 29)

Alice  
Bob  
Cat
A payment channel enables several Bitcoin payments between two users. Current proposals allow payment channel balance updates without the need for a blockchain transaction, but this requires that both users contribute funds. Technically, a bidirectional channel might require that both users contribute funds to the deposit in the opening transaction. However, current proposals allow payment channel balance updates without the need for a blockchain transaction, but this requires that both users contribute funds. Technically, a bidirectional channel might require that both users contribute funds to the deposit in the opening transaction. In the running example, assume that the current payment channels operate in essence as the unidirectional version. Bob.

In the illustrative example depicted in Figure 1, Alice opens a payment channel with Bob, then uses it to pay Bob some bitcoins opening a payment channel. Dashed arrows denote temporal sequence. Alice, independently of the bitcoins that he has received from Alice, makes sure that the payment from Alice to Bob is successful. At the end of a successful payment, every edge has associated a non-negative number that denotes the amount of remaining bitcoins that users can pay to each other edge. The weight on a directed link must have a capacity available along a path connecting the two users and the initial payment value minus the fees charged by intermediate users in such path. Assume that the capacity of the channels before (after) the initial payment is 2 (48) bitcoins. This allows Alice to pay Bob 2 bitcoins (58 bitcoins if Alice pays them directly to Bob), and the capacity of the other channels is 56 (30) bitcoins. For ease of explanation, in the rest of the paper we explain the problem by assuming that the capacity of the channels is infinite. Nevertheless, our simplification greatly ease the understanding of the rest of the paper and proposed algorithms can be easily extended to support bidirectional channels.

Privacy in PCNs: Challenge?

* Off-chain payments ≠ Privacy-preserving payments

Blockchain Transactions

HTLC(Alice, Bob, 1, y, 30)  HTLC(Bob, Cat, 1, y, 29)
Privacy in PCNs: Our Solution

- Our setting: P2P Network

- Our goal:
  - On-chain operations: HTLC as in the Lightning Network
  - Rest of cryptographic operations must be off-chain
  - Full compatibility with the current Bitcoin script

- Our solution:
  - Fulgor: Based on Multi-hop HTLC
Multi-hop HTLC

- Building block: Non-interactive zero knowledge (ZKBoo [GMO16])

\[
x_0 : H(x_0) = y_0 \\
x_1 : H(x_0 \oplus x_1) = y_1
\]
Multi-hop HTLC

- Building block: Non-interactive zero knowledge (ZKBoo [GMO16])

\[
x_0 : H(x_0) = y_0 \\
x_1 : H(x_0 \oplus x_1) = y_1 \\
(x_0, y_0)
\]
**Multi-hop HTLC**

Building block: Non-interactive zero knowledge (ZKBoo [GMO16])

\[
\begin{align*}
\forall x_0 &: H(x_0) = y_0 \\
\forall x_1 &: H(x_0 \oplus x_1) = y_1 \\
\pi &: \exists x_0 \text{ s.t. } H(x_0 \oplus x_1) = y_1 \land H(x_0) = y_0
\end{align*}
\]
Multi-hop HTLC

* Building block: Non-interactive zero knowledge (ZKBoo [GMO16])

\[ \pi : \exists x_0 \text{ s.t. } H(x_0 \oplus x_1) = y_1 \land H(x_0) = y_0 \]
Multi-hop HTLC

- Building block: Non-interactive zero knowledge (ZKBoo [GMO16])

\[ x_0 : H(x_0) = y_0 \]
\[ x_1 : H(x_0 \oplus x_1) = y_1 \]

\[ s := (x_1, y_1, y_0, \pi) \]

\[ \pi : \exists x_0 \text{ s.t. } H(x_0 \oplus x_1) = y_1 \land H(x_0) = y_0 \]
Multi-hop HTLC

$\text{HTLC}(\text{Alice, Bob, 1, } r_1, 30) \approx \text{HTLC}(\text{Bob, Cat, 1, } r_0, 29)$

$\text{HTLC}(\text{Alice, Bob, 1, } y_1, 30) \approx \text{HTLC}(\text{Bob, Cat, 1, } y_0, 29)$

$x_0 : H(x_0) = y_0$

$x_1 : H(x_0 \oplus x_1) = y_1$

$s := (x_1, y_1, y_0, \pi)$

$\pi : \exists x_0 \text{ s.t. } H(x_0 \oplus x_1) = y_1 \land H(x_0) = y_0$
Multi-hop HTLC

\[ x_0 : H(x_0) = y_0 \]
\[ x_1 : H(x_0 \oplus x_1) = y_1 \]

\[ s := (x_1, y_1, y_0, \pi) \]

\[ \pi : \exists x_0 \text{ s.t. } H(x_0 \oplus x_1) = y_1 \land H(x_0) = y_0 \]
Multi-hop HTLC

- Soundness of NIZK => Bob does not lose coins
- Zero-knowledge of NIZK => Bob does not steal coins
Concurrent on-chain payments can be easily ordered by miners

No user has a complete view of off-chain concurrent payments in a P2P network

A blocking solution can lead to deadlocks
Concurrency in PCNs

- Concurrent on-chain payments can be easily ordered by miners
- No user has a complete view of off-chain concurrent payments in a P2P network
- A blocking solution can lead to deadlocks
Concurrency in PCNs

- Concurrent on-chain payments can be easily ordered by miners.
- No user has a complete view of off-chain concurrent payments in a P2P network.
- A blocking solution can lead to deadlocks.
Concurrency in PCNs

- Concurrent on-chain payments can be easily ordered by miners.
- No user has a complete view of off-chain concurrent payments in a P2P network.
- A blocking solution can lead to deadlocks.
Concurrence in PCNs: Our Solution

- A non-blocking solution (Rayo): at least one payment finishes
- Main idea: Use global transaction identifiers

![Diagram showing concurrency and transaction flow]
Global identifiers leak transaction ID to intermediate users

Non-blocking solutions cannot achieve strong privacy
Running time of our solution largely dominated by NIZK

- Creating a proof requires 309 ms. Proof verification requires 130 ms
- Proof size: 1.65MB

5-hop payment:

- **Non-private (LN):** 609 ms
- **Private:** 1929 ms and ~ 5 MB (*Proofs are not included in the blockchain*)
Conclusions

- Define the security and privacy properties of interest in PCN
- Inherent tradeoff between concurrency and privacy
- Fulgor and Rayo: two approaches for concurrency and privacy
- Our solutions are efficient, compatible with Bitcoin script and without storage overhead in the blockchain
Thank you for your attention!

Giulio Malavolta, Pedro Moreno-Sanchez, @pedrorechez
Aniket Kate, Matteo Maffei, and Srivatsan Ravi
@aniketpkate @matteo_maffei