Protocol for Recursive Virtual Channels

Aggelos Kiayias[1,2] and Orfeas Stefanos Thyfronitis Litos[1]

[1] University of Edinburgh
[2] IOHK
akiayias@inf.ed.ac.uk, o.thyfronitis@ed.ac.uk

Abstract. Protocol overview for Recursive Virtual Lightning-like payment channels on Bitcoin

Consider a sequence of parties $A_1, \ldots, A_n$. We say that i is left of $i+1$ and $i+1$ is right of i. $\forall i \in \{2, \ldots, n-1\}$, party A_i has a channel with A_{i-1} of total value $x_{i-1,i}$ and a channel with A_{i+1} of total value $x_{i,i+1}$. A_1 only has a channel with A_2 (of value $x_{1,2}$), likewise A_n only has a channel with A_{n-1} (of value $x_{n-1,n}$).

After following a specific protocol that does not involve any new on-chain transactions, each party holds off-chain a number of transactions and signatures that imply the existence of a new channel between A_1 and A_n with value x', funded by A_1. At a high level, these transactions are as follows:

- Each edge party has a transaction that consumes the funding output of its only channel and produces two outputs: one for the preexisting channel, where the left party has x' coins less and one that carries the x' coins for the virtual channel (read: the left party pays for the virtual channel). Call the latter "virtual output".
- Each intermediate party A_i has three types of transactions:
  - A "first-mover" transaction, which consumes both its channel outputs and produces four: one for the left channel where the left party A_{i-1} has x' less coins, one for the right channel where A_i has x' less coins, one that pays A_i directly x' coins and one virtual output with x' coins.
  - Several "second-mover" transactions which may be used if exactly one of the two adjacent parties has consumed the funding output of the shared channel. Wlog, assume that the party to the left has consumed the funding output $A_{i-1}A_i$ whereas the party to the right has not consumed A_iA_{i+1}. A_i's suitable second-mover tx consumes A_iA_{i+1} and the virtual output produced by A_{i-1}'s transaction. In turn it produces one A_iA_{i+1} funding output where A_i has x' less coins, one output with x' coins for A_i and a new virtual output with x' coins.
  - Several third-mover transactions which can be used if both adjacent parties have consumed their respective funding output. The suitable "third mover" tx consumes both virtual outputs from left and right and produces a new virtual output with x' coins and an output that pays A_i directly x' coins.

Q&A

- *Why are there many second- and third-mover transactions?*
- A virtual output produced by a tx of A_i specifies exactly the interval of parties around A_i that have already made their move. A_i can only spend a virtual output of which the interval ends just before or just after i and the single newly produced virtual output has an interval that is the union of the intervals of the consumed virtual outputs with i added. Therefore A_i has multiple second- and third-mover transactions because each one corresponds to different previous interval(s).

  As a result, each intermediate party can only publish exactly one transaction. This transaction always generates exactly one new virtual output. If it is a first-mover tx, it does not consume a virtual output. If it is a second-mover, it consumes one and if it is a third-mover it consumes two. A third-mover tx can be published only if the publishing party is surrounded (directly or indirectly) by two first-movers, therefore eventually only one virtual output will remain, as intended.
- *What if a malicious intermediary creates a new virtual output and consumes it together with an honest virtual output using its third-mover transaction?*
- As the third-mover tx has a virtual output with a wider interval, the same party cannot repeat the same trick. Since the interval is always widening, even if only one edge party is honest, the attack cannot carry for ever, therefore eventually the edge party will be able to consume the virtual output as intended. Similar reasoning applies to second-mover malicious transactions, where the malicious party fabricates the funding output. Regarding the case where a malicious party fabricates a virtual output and then publishes a second-mover transaction that consumes this fabricated output and a valid funding output, we observe that the valid intervals of the aforementioned virtual output may include only parties that are not towards the direction of the honest counterparty. This means that the counterparty has the same view as if the malicious party was indeed a second-mover, which causes it no financial loss. This fact does not change if more parties are malicious: the only possible difference for any honest party is the ability to spend more than one (second- or third-mover) transactions and therefore gain more coins than if everyone were honest. Intuitively, any malicious party that fabricates a malicious output in order to spend an honest one just introduces more coins to the protocol in a way that does not allow it to gain value.
- *What if a malicious party publishes an old commitment transaction (i.e. consumes a funding output without using any of the first-, second- or third-mover txs)?*
- Its counterparty A_i won't be able to close honestly its other adjacent channel, but it will be able to punish the malicious party with the revocation transaction, thus confiscating all its funds. Therefore, to ensure no monetary loss is possible, A_i must always enforce that $x_{i-1,i,\text{right}} \leq x_{i,i+1,\text{right}}$ and $x_{i,i+1,\text{left}} \leq x_{i-1,i,\text{left}}$ (where $x_{i,j,\text{left/right}}$ is the coins owned by the left-

/right party of channel A_iA_j respectively). This balance check is performed on every payment and new virtual channel.

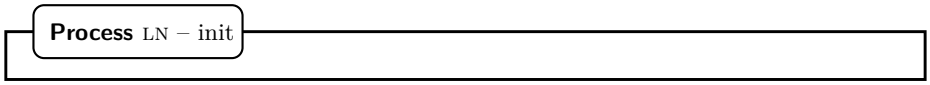

Fig. 1.