I wrote these notes a few weeks ago, publishing here for posterity.

Note:

Any reference to “on-chain” and “off-chain” here relates to the Plasma chain and not the root chain.

Background

The original Plasma paper describes a mechanism of “confirmation signatures” used to ensure that both parties to a transaction have data availability:

1. Alice and Bob observes the transaction and signs an acknowledgement that he has seen the transaction and block. This acknowledgement gets signed and included in another Plasma block.

The authors also hint at a griefing vector enabled by Plasma block withholding:

If blocks are being withheld after step 2 but before step 3, then it is presumed that Bob has sufficient information to withdraw funds, but since neither Alice nor Bob have fully committed to the payment, then it is not treated as complete, depending on information availability either party may theoretically be able to claim the funds. If both parties sign off on Step 3, then it is presumed that it is truly finalized.

Basically, if Bob broadcasts a confirmation signature, then Alice knows that Bob has enough information to claim the funds. This means that transactions will only be valid if Bob (receiving party) has data availability and is able to access the funds. However, if blocks are withheld and Bob is unable to publish the signature on the Plasma chain, then Alice will not know if Bob is able to claim the funds.

A griefing vector becomes apparent when Alice does not know if Bob can access the bonds. If Bob is dishonest, then Alice might attempt to exit from her old input and be caused to lose a bond when Bob challenges with Alice’s spend.

We originally attempted to mitigate this issue in the block withholding case by requiring the confirmation signature to be sent off-chain, directly between Alice and Bob. Our specific construction had a confirmation signature sent from Alice to Bob, and then a signature from Bob. Unfortunately, this enables the same griefing vector because it assumes cooperation between the two parties. Otherwise, there exists a time where Bob has enough information to exit (Alice’s signature and his own), but Alice does not.

Note:

I belive that the above statement is provably true. Concisely, if two parties with secret information must collaborate to generate some value, then there will be some point in time where one party has enough information to generate the value and the other does not.

Quick sketch: There will be a final message in the protocol (let’s say from Alice to Bob). After this message, both parties must know the value. However, no further information can flow from Bob to Alice, so Alice must’ve had enough information to access the value before the message was sent. Therefore, there was a point in time (before the final message) during which Alice knew the value and Bob did not.

What’s the point?

On-chain confirmations are (relatively) safe

Generally, if we’re assuming that Alice and Bob will cooperate in most cases, then it’s fine to get rid of two-party confirmation signatures and just have Bob sign off. This seems to be what the original paper suggests anyway. In the average case (Plasma chain functional), then Bob will be unable to grief as long as we require

that all confirmation signatures be published on-chain. Griefing would only be potentially possible if blocks are being withheld.

However, on-chain confirmations would take up a lot of block space (520 bits per signature for each output). In a 4in4out virtual account model, we’d basically be using half of the block space just for confirmation signatures. This was part of the original motivation for off-chain confirmations.

Off-chain confirmations might be safe enough in some cases

Off-chain confirmations work if we assume that Alice and Bob will cooperate. This is probably a strong assumption, but it’s not entirely unreasonable when parties explicitly trust each other or have a reason to behave. This is particularly true if the recipient is a merchant who wouldn’t be well-served to grief customers.

I think it might be feasible to specify a transaction flag, such that users can specify whether or not they “trust” the recipient and require on-chain confirmations. Think “friends & family” vs “business” purchases in PayPal.

We probably need something better

I think better constructions than confirmation signatures exist. Currently, I'm workingMore Viable Plasma, which removes the need for confirmation signatures. Confirmations follow the logic "implicitly invalid, until explicitly valid", which is generally worse UX than "implicitly valid, until explicitly invalid". However, the construction is non-trivial.