Plasma implementations generally rely on exits and challenges as mechanisms to protect users, i.e. their funds, from the actions of malicious operators.

We can notice and unavoidable dilemma that precedes any such exit and/or challenge. Let's say a potentially malicious action is noticed (e.g. block root/header is submitted to the main chain but the block is not available). At this moment, every Plasma user (or their client software) has to choose one of the following options/strategies:

1. "Rabbit" strategy - The user doesn't want to take any chances and immediately submits an exit (named like this because rabbits run away at the slightest sign of danger).

2. "Child" strategy - The user believes that the action is not necessarily malicious (e.g. it could be just a temporarily crash of the operator's server), and decides to wait for some time (named like this because children are innocent and trustful).

There are tradeoffs for both. By choosing the "rabbit" strategy, the user can easily waste their time and money "running away" from (and then "coming back" to) an honest operator who really just had a temporarily technical issue. Also, if the majority of users in the Plasma ecosystem prefer this strategy, that can be cause issues for the main chain (too many exits -> clogged main chain). On the other hand, by choosing the "child" strategy, the user can completely lose the chance to submit an exit (or challenge malicious operator's exit) at a later point in time (e.g. the main chain can get clogged in the meantime), and consequently lose all of their funds (this especially relates to Plasma MVP).

Also, it's not clear who should be making the decision - the user themselves or the client software?

To make this decision somehow easier, it might be good to have some sort of scoring for Plasma chains/operators in the future. Having that, we can even try to design a simple hybrid model for such situations:

1. A potentially malicious action is detected by the client software

2. UI notifies the user about the issue, provides the scoring to support the decision, and waits for the user's input (e.g. "We were unable to download block #345678

from the OmiseGO chain. Their score is Great (9.8/10). Do you want to submit an exit (0.2ETH available)? [Yes]/[No, update me in 1 hour]"

1. The users provides an input and the software acts accordingly.

Thought and comments are welcome.Thanks!