Problem:

State channels, truebit, etc have deadlines for receiving final updates and fraud proofs. An attacker can participate in many of these state channels, commit mass fraud, and then congest the main chain (e.g. by attacking lots of state channels at the same time, or just spamming transactions). Fees could become very high, and attackees could miss the deadline.

Possible solution:

Allow a deadline to be extended by a transaction claiming that there are many valid fraud proofs waiting to be processed. The claim carries a security deposit as well as the root of a merkle tree of the supposed transactions containing fraud proofs. The contract managing the deadline could sample from the list of claimed fraud proofs, and then interactively request & verify the actual fraud proofs. If the claimer fails to provide an actual fraud-proof-containing transaction, then the original fraud deadline is maintained and the claim's security deposit is destroyed.

This "proof of proofs" process doesn't require too many transactions, so they could include high fees that cut through congestion. Sufficiently-verified claims could both extend the deadline and deliver a reward to make up for the work, transaction fees, and time-value of the security deposit. The reward could be funded by state channel tx fees.

Could an attacker endlessly extend the deadline by committing fraud against themselves in order to generate valid proof-of-proofs? We could make it expensive:

1. After the deadline has been extended, the protocol could continue to demand submission of all the fraud proofs that the claimer committed to. If the claimer fails to submit any of them, their security deposit could be revoked.

2. It could be the case that each processed fraud proof results in the destruction of an associated security deposit (separate from the deposit in the claim). The claim could include a deposit size for each fraud proof it attests to, and selections for verification could be weighted by the size of the security deposit. Prior to actually extending the deadline, interactive verification could continue until some threshold amount of coin has been provably destroyed.

(This approach wouldn't help against e.g. a network DDoS that takes down a sufficient number of nodes)