Great presentation about Casper FFG at EdCon. It seems many years of effort are finally paying off as the team prepares for the release of the most anticipated smart contracts ever created.

One interesting thing about CFFG is the phased approach the team is taking. Casper FFG is a finality layer on top of the PoW chain, adding higher security through economic finality to the chain, enough so that there have been discussions on reducing the block reward as the economic security of the network moves slowly from PoW to PoS.

Let’s assume PoS is released to the mainnet. It has been deployed for about a year, and it has worked very well up to this point, accuring 10M ETH in deposits by many 1000s of dedicated validators. Correct me if I’m wrong, but that would make the Casper FFG contract one of the largest honeypots by value ever created as a smart contract, with around 10% of all Ether locked into this account (and at much higher valuations than the DAO).

Now, the team is undergoing multiple verification approaches, which is the right process for building a smart contract which we all know is impossible to change once deployed. This verification will surely weed out easy exploits from the contract, ensuring that any potential exploit would have to be extremely creative. The fact remains, in software, there is ALWAYS something that can be improved, sometimes outright bugs do exist. No amount of verification activity can ever counteract that, only add higher degrees of confidence in the system.

The TL;DR of it all is this:

What process is the Casper FFG team following for monitoring and event response in case an exploit is found, either by a White Hat (who reports the exploit in secret) or a Black Hat (who uses the exploit to drain and/or lock funds in the Casper FFG contract)?

1. Is there a way to lock the operation of the contract (at least the methods that affect fund deposit/recovery) until a proper recovery procedure has been identified?

2. If the exploit traps or withdraws all of the funds in a short time period, how does the PoW reward respond to this event in order to increase the economic security of the network? How does that affect the economics of the network?

3. Is there a way to mitigate this large honeypot through fund storage in a much simpler contract, with defined entry/exit points and the ability to lock access in case of an attack?

4. What is the transition and communication procedure for handling this event, to ensure it occurs as smoothly as possible to those in the community who are rightly concerned?

Thank you to an engineer from a company that has undergone a similar exploit event. He reminded me the importance of deployment and response procedures for handling smart contract exploit events, something they have developed as a team in response to their incident and is now integral to their smart contract development process.