smartBNB: A NEO<>BNC bridge based on XCLAIM

Albert Acebrón
What we built

- An XCLAIM-based bridge that allowed long-term porting of assets from Binance Chain to NEO
- Note that we are referring to the original cosmos-based Binance Chain, not the newer Ethereum fork
Bringing XCLAIM to prod

- XCLAIM only defines the basic building blocks for porting, multiple things are left as future work

- These include incentives, how to deal with over and under-collateralization, fungibility...
Incentives - Design

● Ported tokens carry a cost for collateral-providers, as they must lock capital and assume risk, for which they need to be rewarded.

● Users should be incentivized to eventually redeem the tokens as otherwise collateral-providers could have their capital locked forever.
Incentives - Spec

• Dilute token holders on a per-block basis by burning 0.0001% of their balances each block
• Burned tokens are awarded as fees to collateral providers
• Solves all problems but breaks an invariant (constant balance) that many contracts rely on
Under-collateralization

- Deposits <120% get liquidated
- Deposits >150% can have the extra collateral withdrawn
Challenges

- Binance Chain is a validator chain, so SPV proof verification requires checking their signatures.
- But NEO doesn’t support Ed25519 sig-checks nor the hash function used in them.
- We had to re-implement all of it on NEO’s VM.
Efficient signature verification

- Initial naive implementation had an execution cost of 55k$ in GAS
- We implemented a Truebit-like system to lower the cost through optimistic verification games (removing the forced failure part of truebit)
Unsolved problem

• There can be multiple redemptions with the same collateral provider happening at the same time
• But there’s a fixed amount of reward deposits (money awarded to challenger)
• Some challengers may end up not being rewarded
• Attacks based on this require a large money burn and redeemer is still incentivized to challenge, but it means that we can’t rely on challengers alone
End

crosschain@albert.sh