TLDR

About

- During the recent MixBytes

audit contest over a feature/shapella-upgrade branch, I was lucky enough to look into the complex architecture behind implementing EIP-4895. A huge shoutout to the team for providing such a robust solution. The protocols like EigenLayer

and Asymmetry Finance

have already inherited a lot of ideas. Anyway, due to the limited time and the contest payout distribution formula, it was a little bit hard to think in a free way, meaning that you're always trying to search for low-hanging bugs, especially if the payout distribution formula is not sybil resistant.

Results

- Despite the lack of time, I managed to receive ~33% of the total shares by dropping: 1 high and 2 medium severity issues, winning the whole MixBytes

contest and making a tiny contribution to the Lido's ecosystem!

- Around that time, I got the top 5 out of ~250 pretty competitive participants in the code4rena contest run by Asymmetry Finance

which aims to decentralize the LSDs market!

Background

- Due to the fact that a lot of crucial operations are running asynchronously, it's definitely hard to foresee everything. Therefore, it opens up the doors towards poking around. From my experience, a bunch of critical bugs appear at the junction of multiple integration modules. For example, Lido's earliest front-running issue appeared in between the details related to the CL and the code on the EL side. It was a simple and clean exploit scenario that could end up with catastrophic consequences for the Lido community. Fortunately, everything was fixed smoothly.

Scope of Work

- When it comes to security, it's pretty hard to define exact deliverables! Therefore,

the scope for this live audit is not exactly limited, meaning that the goal behind this proposal is not just about reviewing a certain piece of code but rather thinking about the system as a whole and trying to break it. I think feature/shapella-upgrade branch would be a perfect start point!

Bugs Disclosure

- Any critical or high-severity issues with existential threats should be disclosed via Lido's bug bounty program on Immunefi, applying a certain bounty discount mentioned below if the vulnerable asset is within the scope. Anything apart from it could be contributed directly to the Lido via Github!

Grant Request

Timeline

- The absence of exact deliverables doesn't fully allow everything to break down smoothly. However, considering the fact that 3 months would be perfect to represent the first steps, it's possible to share the results with the Lido community at the end of each month!
- Also, I'll share a notion document for the community to fully track the progress in live!

Discount Factor and Payment

- The bounty for major vulnerabilities submitted via Immunefi according to the impact listed on Lido's program page

should include the 25% discount!

- The initial grant request is: $44,444 worth of DAI with the following structure for execution:
- 25% upon agreement
- 25% upon M1 report delivery
- 25% upon M2 report delivery
- 25% upon M3 report delivery + the bonus of $55,555 worth of DAI, if the Lido community decides to share based on a provided quality!
- 25% upon agreement
- 25% upon M1 report delivery
- 25% upon M2 report delivery
- 25% upon M3 report delivery + the bonus of $55,555 worth of DAI, if the Lido community decides to share based on a provided quality!

Summary

- Overall, I think it should be a great experience for Lido, not only because of this proposal but for future engagements as well! There might be a demand for reviewing the fresh code, and having someone who fully understands the codebase makes it a lot easier to pre-review the code before going for an audit! Would love to hear the feedback from the Lido community, especially from: @ujenjt, @TheDZhon, @kadmil, @GrStepanov.

Author

@m_Rassska | 26.05.2023

With a decent CS background, additional experience in competitive programming, and 2 years in the Web3 security space trying to make it more secure!