Apraemio Hacken Audit Addendum

Project’s notes

Introduction

Hacken did an audit of Apraemio’s smart contracts on 19/11/2024. The scope of the review was the currently deployed system on Binance Mainnet from the the repository: https://github.com/Apraemio/APRA/tree/audit at commit: 9e952d4
There were 4 findings of 2 severities:

- 2 medium
- 2 low

After considering the probability and possible impact of these findings the project decided to accept them for the current deployment and fix them for possible upcoming releases on different chains. This article describes the reasoning behind this decision case by case.

Potential Risk: Centralized Minting to a Single Address

This finding points to the method of the TGE, when all tokens were minted to the same address that could pose an elevated risk of being hijacked of a target of misuse. Since the project is already live, and the main supply of tokens were migrated to multisignature safe contracts (bnb:0x91D8Fe5EC8F2A0b72d2593a5C6AE9E85CeeF64dc, bnb:0x4E8F2c8485403F6A59f1Cc96266Ef9CCEb5f9a92) backed by several hardware wallets this risk is assumed to be mitigated.

Other Potential Risks

These findings refer to off chain functions that were out of scope of the audit. The project continuously communicates these methods and is open to discuss them with those who are interested.

F-2024-7128 - Double Fee Deduction and Inaccurate Token Amount Stored Due to Fee on Transfer - Medium

This finding refers to a possible calculation error if the vesting contract (TimeLock) is not excluded from the token fee mechanism. The contract was successfully excluded from fee calculation right after deployment, so we assume this risk is to be mitigated.

F-2024-7130 - Self-Transfer in TimeLock Contract Will Lead to Locked Tokens - Medium

There is a possibility in the vesting contract (TimeLock) to transfer the locked amount of tokens to another account in case the original one gets compromised. If this transfer is parametrized incorrectly and the origin account is set as the target account, then funds could get lost. We decided to accept this possible error, since the probability is low, and a UI function will be provided for initiating this transaction that will check for this possible misconfiguration.

F-2024-7129 - Missing Boundaries For The icoTimestamp Value Low

The finding refers to the possibility of setting a past time as the beginning of the vesting period, which could cause the premature withdrawal of locked funds. This was a design decision for two reasons:

1) Since several exchanges outright denies the publishing of the time of listing a token it was possible that setting the time into the past would be necessary (since if it was set in the contract, it would have been visible to all)
2) After setting the time of the listing it is also necessary to “lock” the contract to make funds retrievable. It works as another safety step to avoid such accidental mismanagements.

Since at the time of publishing these findings the date of the listing is known and public, and the correct timestamp is set and locked in the vesting contract, the risk is mitigated.

F-2024-7138 - Potential Overflow In The IncreaseAllowance() Function - Low

The problem described in the finding refers to a possible overflow in the APRA token’s contract where it is possible to increase the spending allowance given to a specific address. If an allowance is already present and the caller would like to increase it with an amount that is several magnitudes bigger than all the tokens minted, it could cause the allowance to become small instead of the expected unreasonably high value. The possibility of such a scenario is quite small and it also raises the likelihood of an algorithmic error on the caller’s side. The potential impact is also minimal, since querying the actual allowance is possible which makes recognizing this event possible. After rationalizing the possible gains of this fix versus the administrative costs we decided to accept it in the current deployment.

Gas Optimisation Observations

Different blockchains require different gas fees for operation. The current deployment of the project is live on the Binance Smart Chain where transaction fees are minimal and the proposed optimisations would make negligible difference. So we have decided that we would not redeploy the contract for the current chain but we do make these optimisations to the codebase to make further deployments as optimal as possible.