HACKEN

SMART CONTRACT PRE-AUDIT REPORT

Customer: Bonuz

Date: Nov 28, 2023

Overview

Name	Smart Contract Pre-Audit Report for Bonuz
Platform	EVM
Language	Solidity

Pre-Audit Results

Repository Management

It is advisable to maintain the source code and tests provided for the purpose of the Audit within a unified repository. Should this prove unfeasible, comprehensive instructions on collaborating with the repository group must be provided.

Current State: Repository management is sufficient.

Consequences: None

Action Points: None

Commit Management

It is preferable that there are no subsequent commits beyond the agreed-upon state when initiating the audit process, indicating the completion of development. In the event that such commits exist, it is the customer's responsibility to determine whether to proceed with the most recent commit or the commit that was available at the time of the agreement. **HACKEN bears no responsibility for identifying issues beyond the defined scope.**

Provided Commit: f8ef0de6b413bf411ae94ac21cfce38190afc35d.

Current State: The provided commit is not the latest one.

Consequences: The audit may be performed for the outdated code. This can lead to some missed issues, non-audited features, or longer remediation time.

Action Points: **Confirm the provided commit is the actual one or provide the latest commit : f978b897564bcf8724ed8431b7fc29903f876360.**

Contracts & Scope

The provided contracts that are in scope are :

File Path & SHA-3 Hash	LoC
File: contracts/BonuzSocialId.sol SHA3: d22a234e86a2c8e1aa92dadf5bbd27a24bcfe181e632326173ea307d63d9351d	127
File: contracts/BonuzTokens.sol SHA3: 09a7e32c4a9dc56fb464371ab8893ce05588f58cd2d1ac61c9f2510d0c232194	309

Current State: There are no missing dependencies

Consequences: None

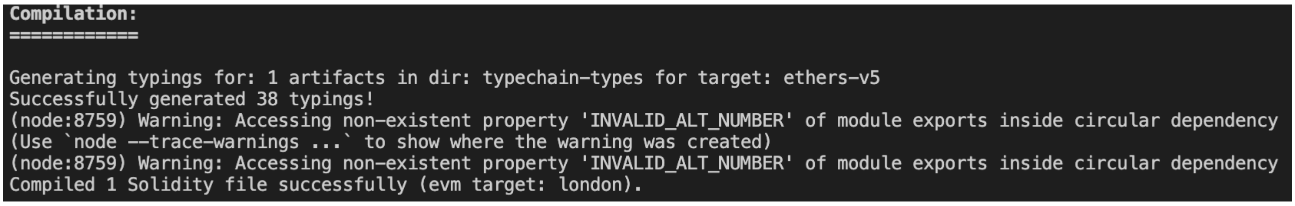
Action Points: None

Environment Configuration & Deployment Instructions

Deployment instructions are necessary for the auditor team not to use their limited time on edge cases and special configurations while running the code and to test the project in more detail.

The deployment instructions are not provided.

Compilation Results:



```
Compilation:
============

Generating typings for: 1 artifacts in dir: typechain-types for target: ethers-v5
Successfully generated 38 typings!
(node:8759) Warning: Accessing non-existent property 'INVALID_ALT_NUMBER' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
(node:8759) Warning: Accessing non-existent property 'INVALID_ALT_NUMBER' of module exports inside circular dependency
Compiled 1 Solidity file successfully (evm target: london).
```

Current State: The deployment instructions are not sufficient

Consequences: Missing deployment instructions will require longer time and effort as it will make it difficult for the auditors to understand and examine the project. This will be reflected in the audit fee and duration. In addition to these, a misconfigured environment will reduce the effectiveness of the audit.

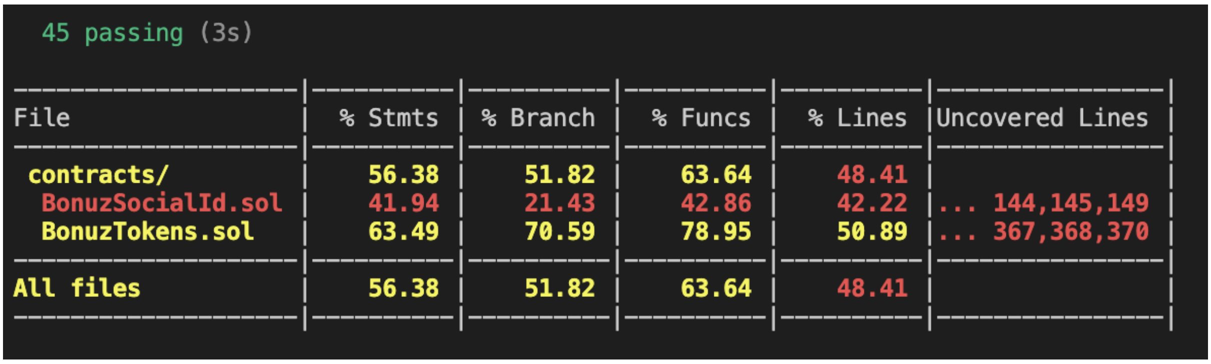
Action Points: Provide deployment instructions

Test Results

Testing is a powerful tool that allows verifying that implementation is compliant with requirements. Failures and boundary value cases should be checked for maximum performance.

Tests should be configured to run on the project environment without the necessity to start any third-party tools like local Ethereum node, etc. In case of uncommon repository configuration, instructions on how to run the test coverage measurement should be provided.

The current test coverage is **51.82%** (Branch).



```
  45 passing (3s)

---------------------|----------|----------|----------|----------|----------------|
File                 |  % Stmts | % Branch |  % Funcs |  % Lines |Uncovered Lines |
---------------------|----------|----------|----------|----------|----------------|
 contracts/          |    56.38 |    51.82 |    63.64 |    48.41 |                |
  BonuzSocialId.sol  |    41.94 |    21.43 |    42.86 |    42.22 |... 144,145,149 |
  BonuzTokens.sol    |    63.49 |    70.59 |    78.95 |    50.89 |... 367,368,370 |
---------------------|----------|----------|----------|----------|----------------|
All files            |    56.38 |    51.82 |    63.64 |    48.41 |                |
---------------------|----------|----------|----------|----------|----------------|
```

Current State: The coverage is average

Consequences: This will reduce the test score of the audit report (If no issues are found, the maximum possible score would be: 3.0/10.0).

Action Points: **Improve code coverage for the following files:**

- **BonuzSocialId.sol**
- **BonuzTokens.sol**

Tests Status: FAIL

Documentation

Functional Requirements

It is crucial for any project to have publicly available functional requirements that clarify the project's objectives and the means by which they are achieved. Detailed explanations of each smart contract entry point considerably streamline the process of examining the code.

Insufficient functional requirements can result in the discovery of numerous false-positive findings. Additionally, the absence of functional requirements makes it challenging for auditors to comprehend the project's business logic, assumptions, formulas, requirements, and features. Consequently, certain logic bugs may go unnoticed.

Provided documentation link: here here

Current State: The functional requirements are partially provided.

Consequences: Missing functional requirements reduce the documentation quality score of the audit report. **(If no issues are found, the maximum possible score would be: 9.3/10.0)**. A sample can be found here

Action Points: **The missing parts for the functional requirements are:**

- **Business logic is not provided.**
- **Use cases are not provided.**
- **Project's features information is not proved**

Technical Requirements

The absence of technical requirements poses challenges for auditors in comprehending the project's architectural logic, key functions, infrastructure, assumptions, and employed technologies. Consequently, it becomes arduous to identify certain logic bugs that may exist within the project.

To facilitate developers' and auditors' involvement in the project, it is imperative for any project to have well-documented technical requirements. Moreover, the documentation should encompass the mathematical formulas utilized within the project. **Insufficient technical requirements can result in longer estimation periods and higher charges.**

Provided documentation link: here here

Current State: The technical requirements are not provided.

Consequences: Missing technical requirements reduce the documentation quality score of the audit report. (If no issues are found, the maximum possible score would be: 9.7/10.0). A sample can be found here

Action Points: **The missing parts for the technical requirements are:**

- **Used technologies information is missed**
- **Roles and authorization information is missed**
- **Architectural design information is missed**
- **Contract, key-function, and state variable descriptions is missed**

Implementation Details

The items encountered by auditors during an audit necessitate additional efforts to ensure the comprehensive identification and mitigation of potential issues.

Metric	Status
The project can send/receive funds	**Y**
Assembly Usage	**N**
Low-Level Calls	**N**
Delegate Calls	**N**
Hash Functions	**N**
External Interactions	**Y**
Complex Formula for Fees/Interest/Rewards/Logic	**N**