SPADE SOLIDITY AUDITS

FHM Token Audit

November 26, 2021

For :
FHM Team

Website:
www.fantohm.com

Telegram:
t.me/fantohm

Twitter:
@SpadeAudits

Telegram:
t.me/spadeaudits

Disclaimer

Spade Solidity reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Spade to perform a security review.

Spade Solidity Reports do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

Spade Solidity Reports should not be used in any way to make decisions around investment or involvement with any particular project.These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

Spade Solidity Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Spade Solidity’s position is that each company and individual are responsible for their own due diligence and continuous security. Spade Solidity’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

What is a Spade Solidity report?

- A document describing in detail an in depth analysis of a particular piece(s) of source code provided to Spade Solidity by a Client.

- An organized collection of testing results, analysis and inferences made about the structure, implementation and overall best practices of a particular piece of source code.

- Representation that a Client of Spade Solidity has indeed completed a round of auditing with the intention to increase the quality of the company/ product’s IT infrastructure and or source code.

OVERVIEW

Project Summary

Project Name	FantOHM Smartcontracts Audit
Description	DeFi
Platform	Fantom Network

Audit Summary

Delivery Date	November 26, 2021
Method of Audit	Static Analysis, Manual Review
Timeline	Story Points - 64

Vulnerability Summary

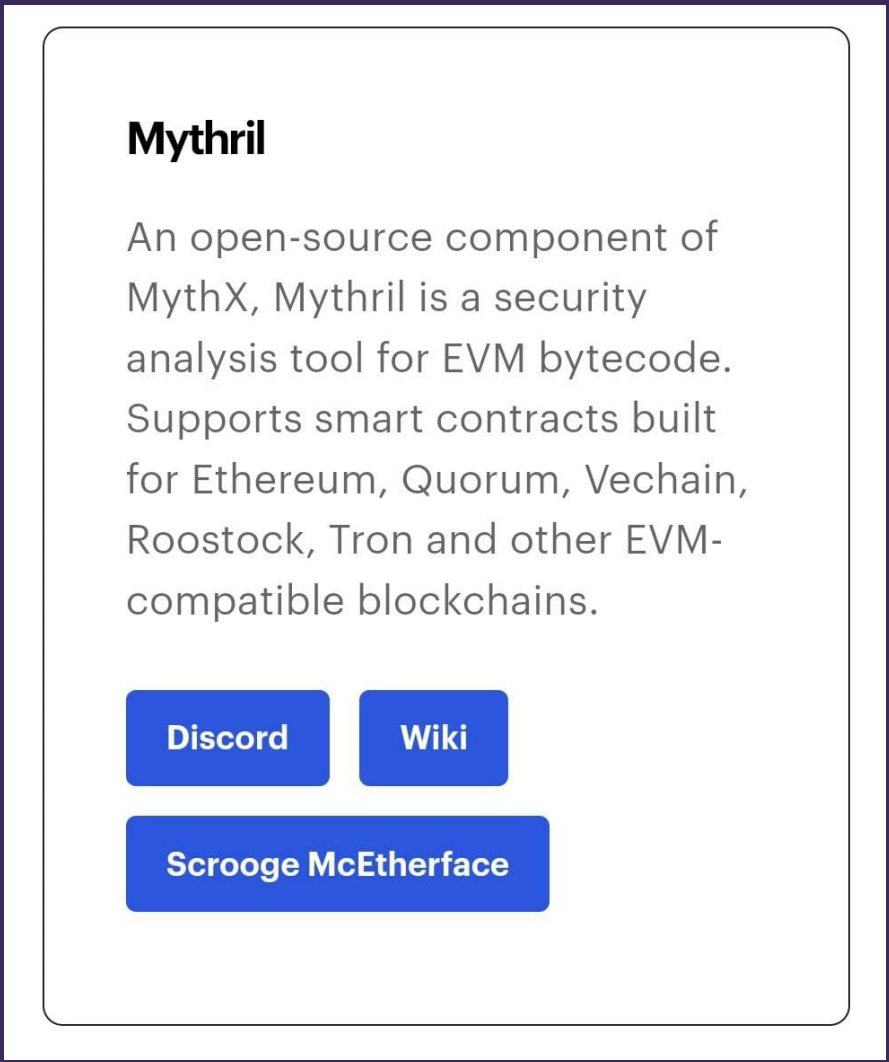
Total Issues	0
Total Critical	0
Total High	0
Total Medium	0
Total Low	0
Total Informational	0

Executive Summary

Our detailed audit methodology was as follows:

Step 1
A manual line-by-line code review to ensure the logic behind each function is sound and safe from common attack vectors.
Step 2
Simulation of hundreds of thousands of Smart Contract Interactions on a test blockchain using a combination of automated test tools and manual testing to determine if any security vulnerabilities exist.
Step 3
Consultation with the project team on the audit report pre-publication to implement recommendations and resolve any outstanding issues.

Auditing Tools :



https://consensys.net/diligence/tools/

Grading

The following grading structure was used to assess the level of vulnerability found within all Smart Contracts:

Threat Level	Definition
Critical	Severe vulnerabilities which compromise the entire protocol and could result in immediate data manipulation or asset loss.
High	Significant vulnerabilities which compromise the functioning of the smart contracts leading to possible data manipulation or asset loss.
Medium	Vulnerabilities which if not fixed within in a set timescale could compromise the functioning of the smart contracts leading to possible data manipulation or asset loss.
Low	Low level vulnerabilities which may or may not have an impact on the optimal performance of the Smart contract.
Informational	Issues related to coding best practice which do not have any impact on the functionality of the Smart Contracts.

About

Fantohm is building a community-owned decentralized financial infrastructure to bring more stability and transparency for the world. FantOHM (FHM) is a market making token backed by other cryptocurrency assets. With mechanisms to ensure the price is always greater than or equal to 1 Magical Internet Money (MIM), the protocol provides heavy incentives for passive and active investors through staking and bonding respectively. As bonds are created the liquidity backing of FHM is distributed into newly minted FHM tokens with the goal being 1 FHM >/= 1 MIM. These new tokens are largely distributed to stakers while investors who have purchased bonds may access a discounted price point for FHM after their vesting period.

FantOHM (FHM) Token Contract

https://ftmscan.com/token/0xfa1fbb8ef55a4855e5688c0ee13ac3f202486286

Scope of Audit

The following Smartcontracts are to be audited by Spade Audits Team.

FantohmTreasury

https://ftmscan.com/address/0xA3b52d5A6d2f8932a5cD921e09DA840092349D71

MIM bond: FantohmBondDepository

https://ftmscan.com/address/0xd4b8a4e823923ac6f57e457615a57f41e09b5613

DAI bond: FantohmBondDepository

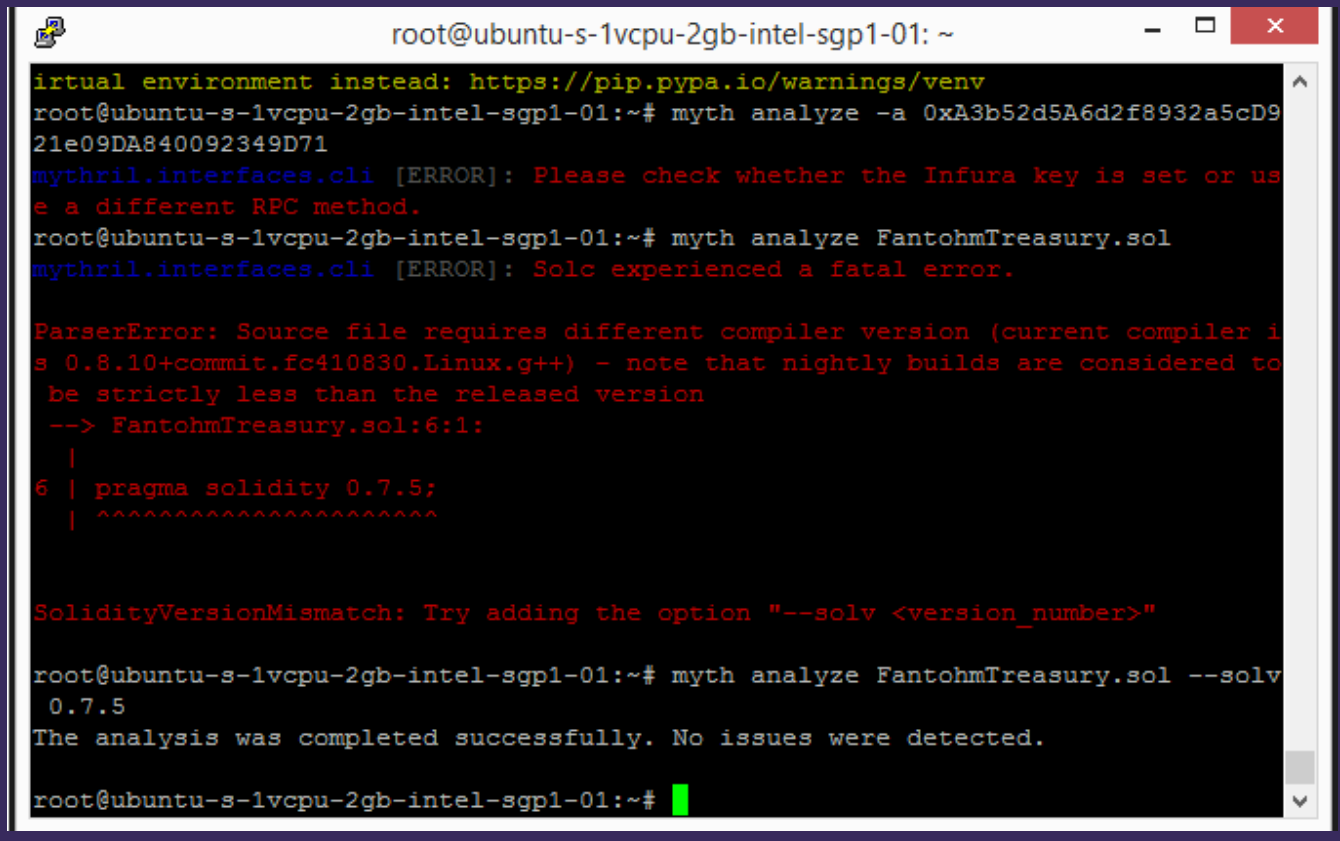
https://ftmscan.com/address/0x462eec9f8a067f13b5f8f7356d807ff7f0e28c68

LP bond: FantohmBondDepository

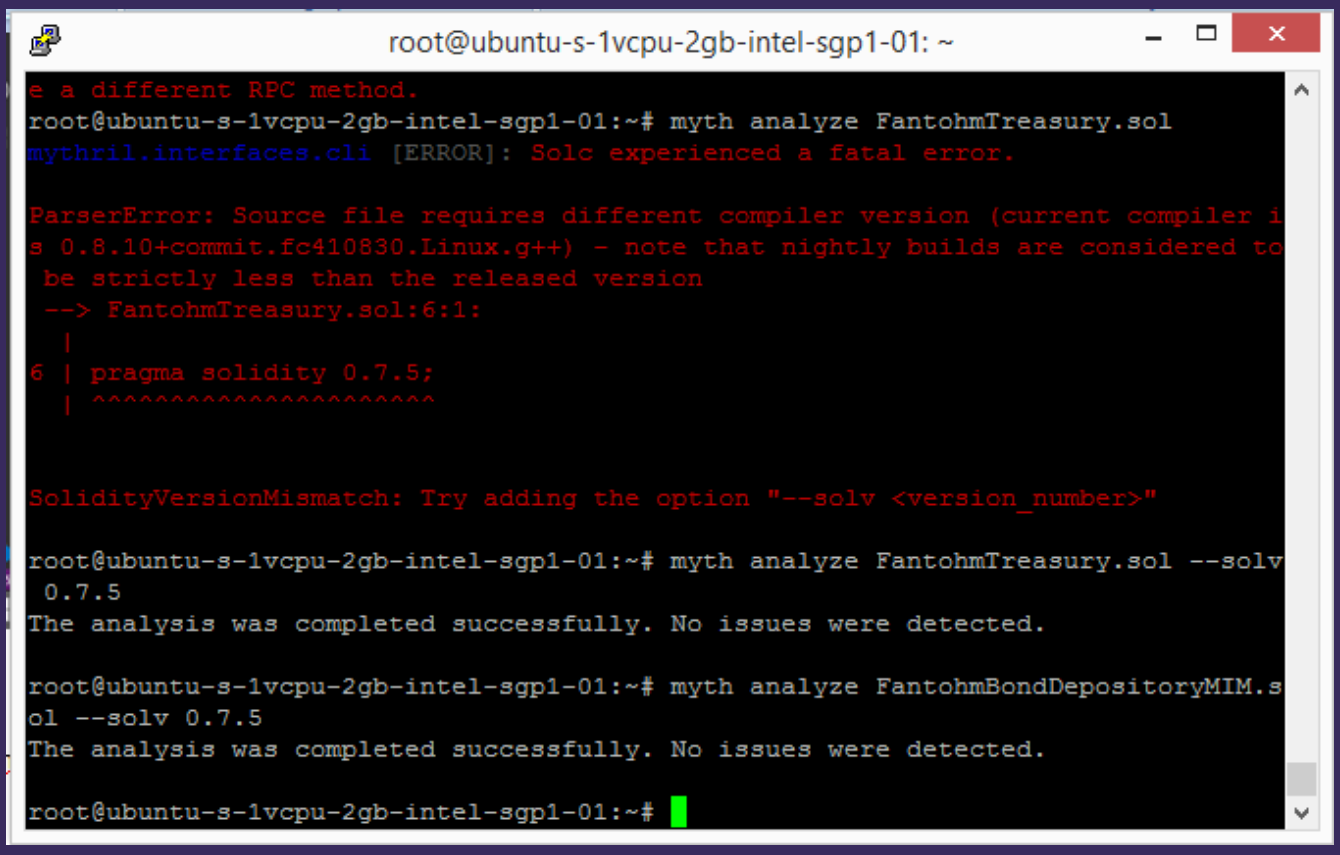
https://ftmscan.com/address/0x71976906ad5520a1cb23fd40b40437c1a2640bcd

Mythril Tool Test Report

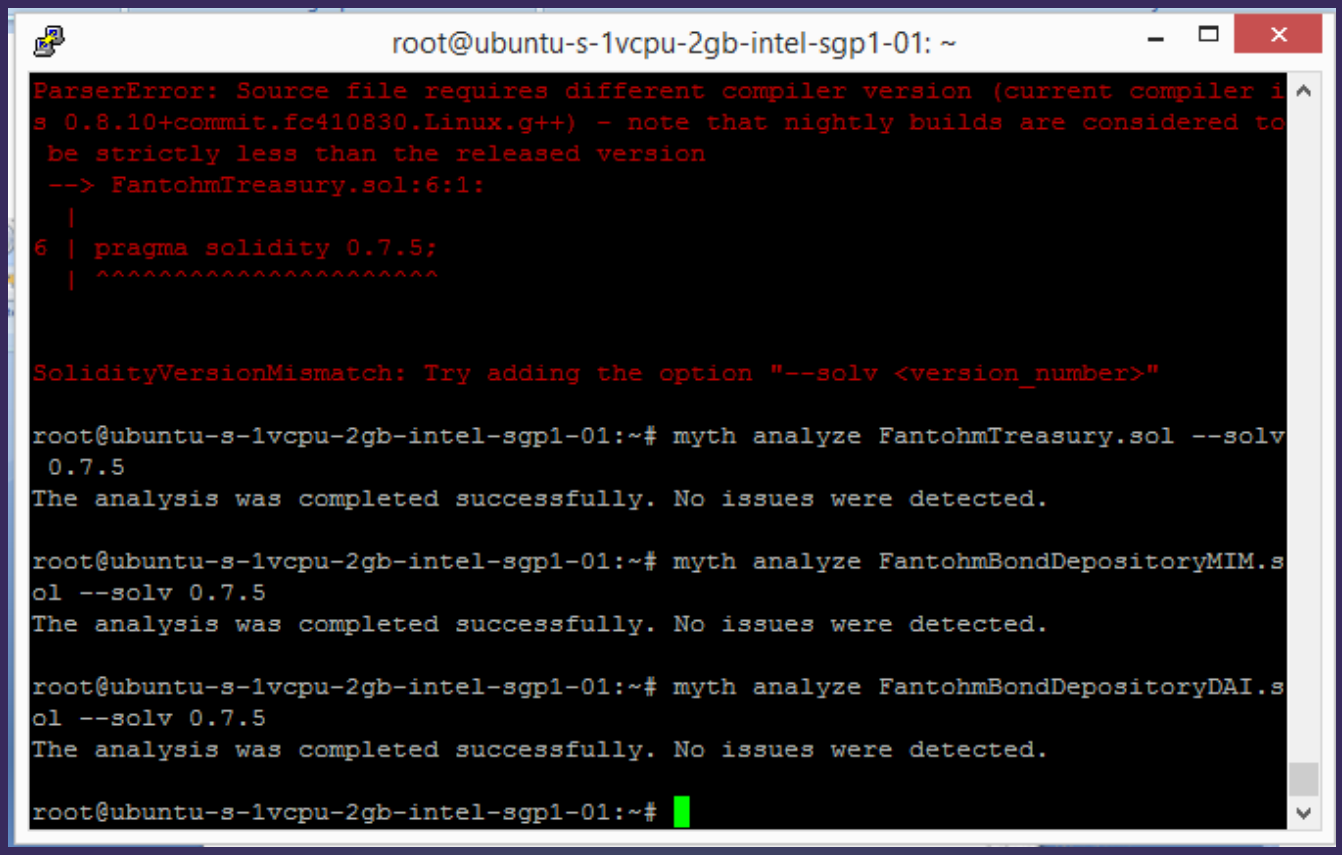
FantohmTreasury.sol



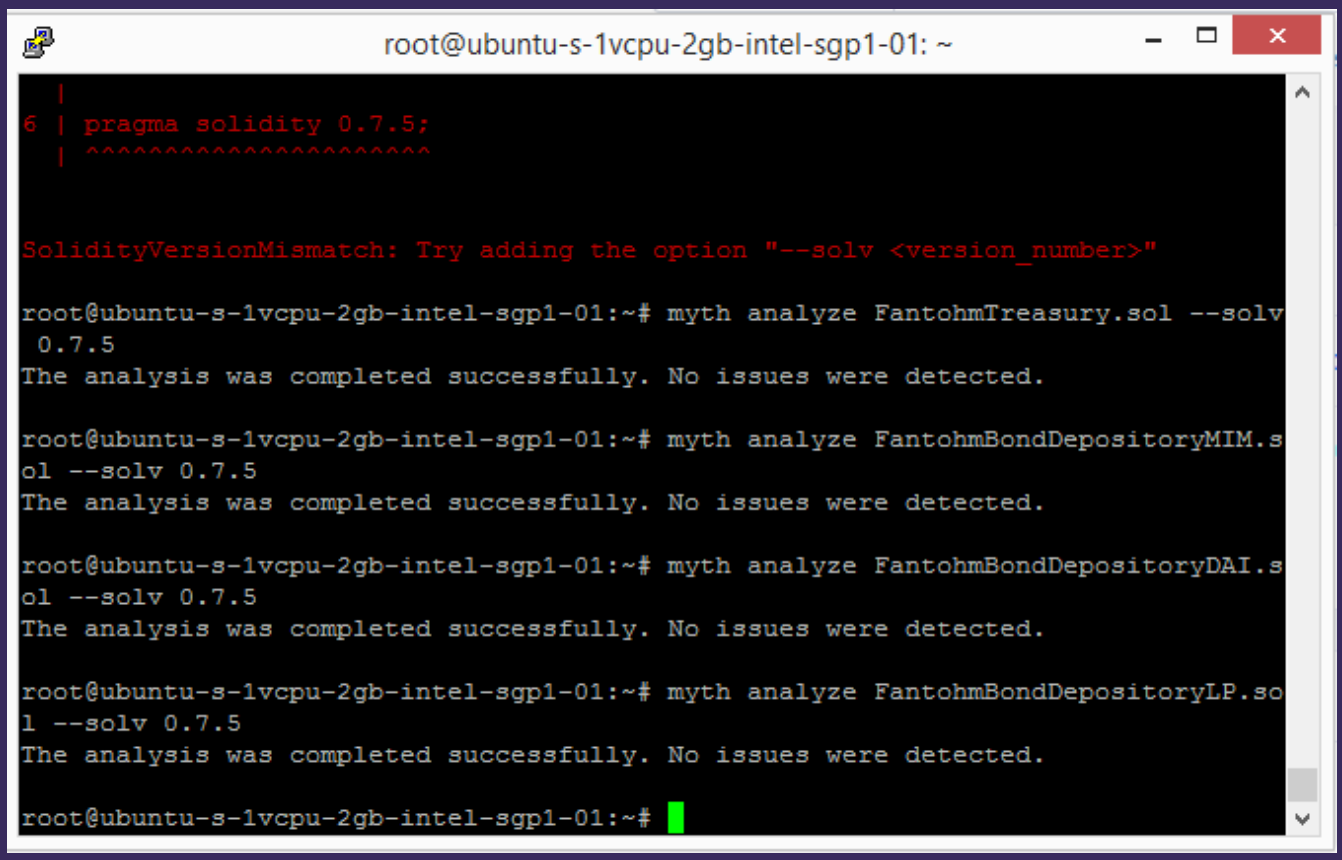
FantohmBondDepositoryMIM.sol



FantohmBondDepositoryDAI.sol



FantohmBondDepositoryLP.sol



FantohmBondDepositoryMIM.sol, FantohmBondDepositoryDAI.sol & FantohmBondDepositoryLP.sol all have the same code.

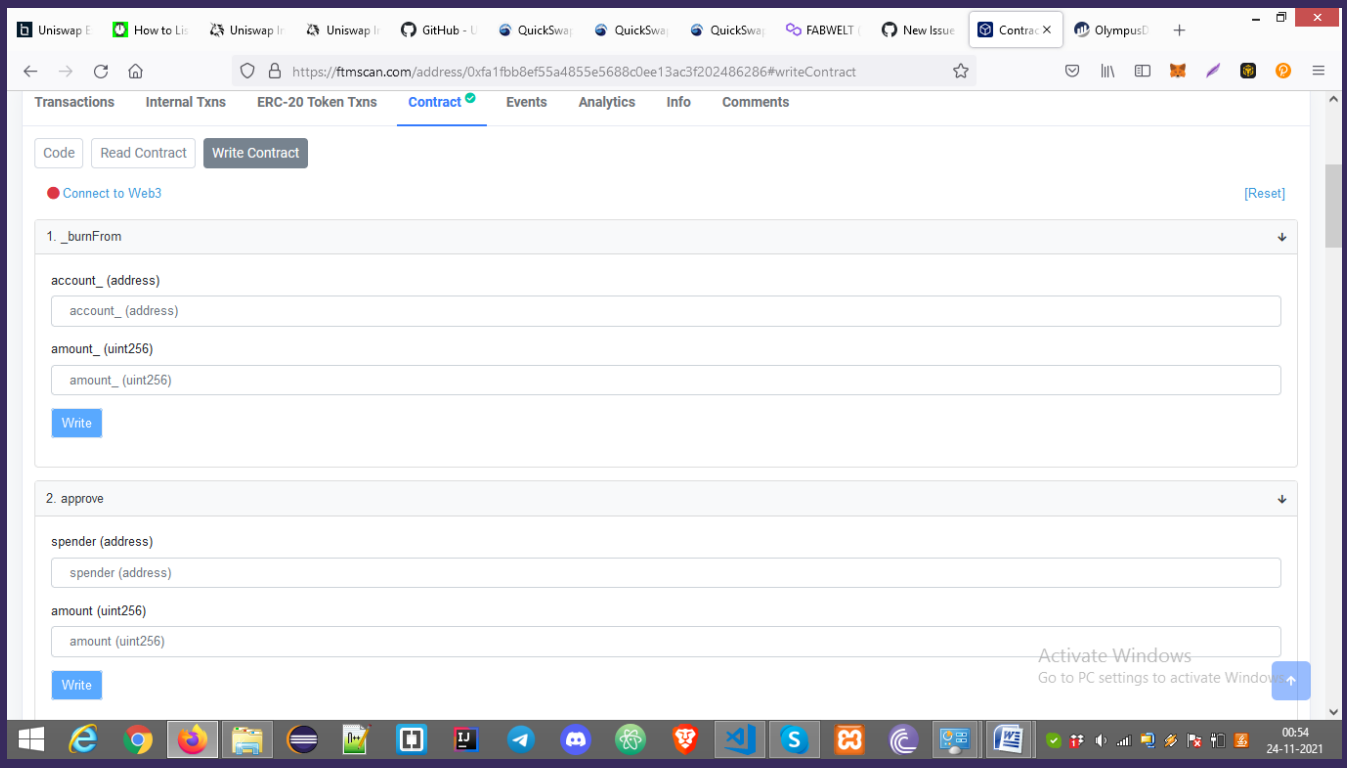
- Constructor Mismatch
- Ownership Takeover
- Redundant Fallback Function
- Overflows & Underflows
- Reentrancy
- Money-Giving Bug
- Blackhole
- Unauthorized Self-Destruct
- Revert DoS
- Unchecked External Call
- Send Instead Of Transfer
- (Unsafe) Use Of Untrusted Libraries
- (Unsafe) Use Of Predictable Variables
- Deprecated Uses
- Semantic Consistency Check
- Business Logics Review
- Functionality Checks
- Authentication Management
- Access Control & Authorization
- Oracle Security

- Digital Asset Escrow
- Operation Trails & Event Generation
- ERC20 Idiosyncrasies Handling
- Avoiding Use of Variadic Byte Array
- Using Fixed Compiler Version
- Making Visibility Level Explicit
- Making Type Inference Explicit
- Adhering To Function Declaration Strictly

The above mentioned checks were all performed on the smartcontracts and the outcome was satisfactory.

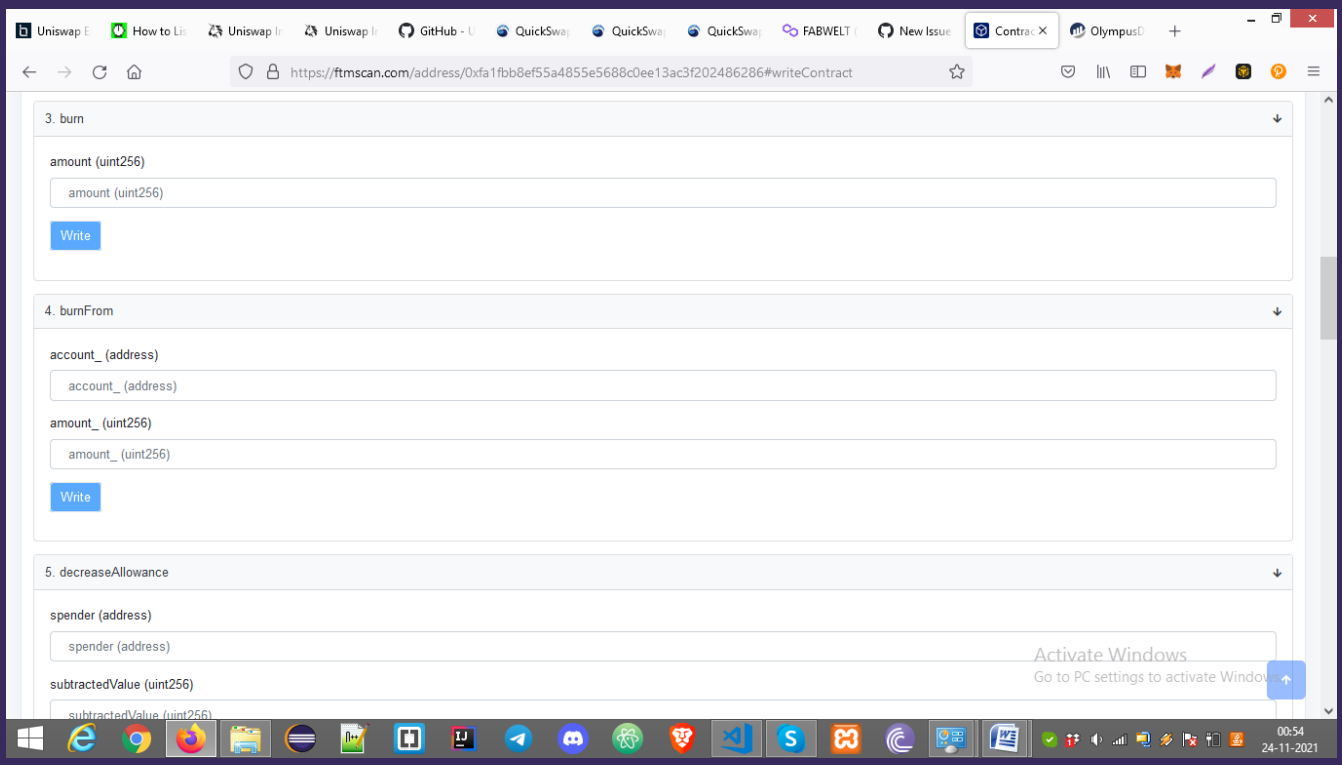
FantohmERC20Token.sol

Fantohm Token contract has 13 write functions. All these write functions have been copied from OHM Token contract and no changes have been done on the code. The 13 write functions have been shown below...



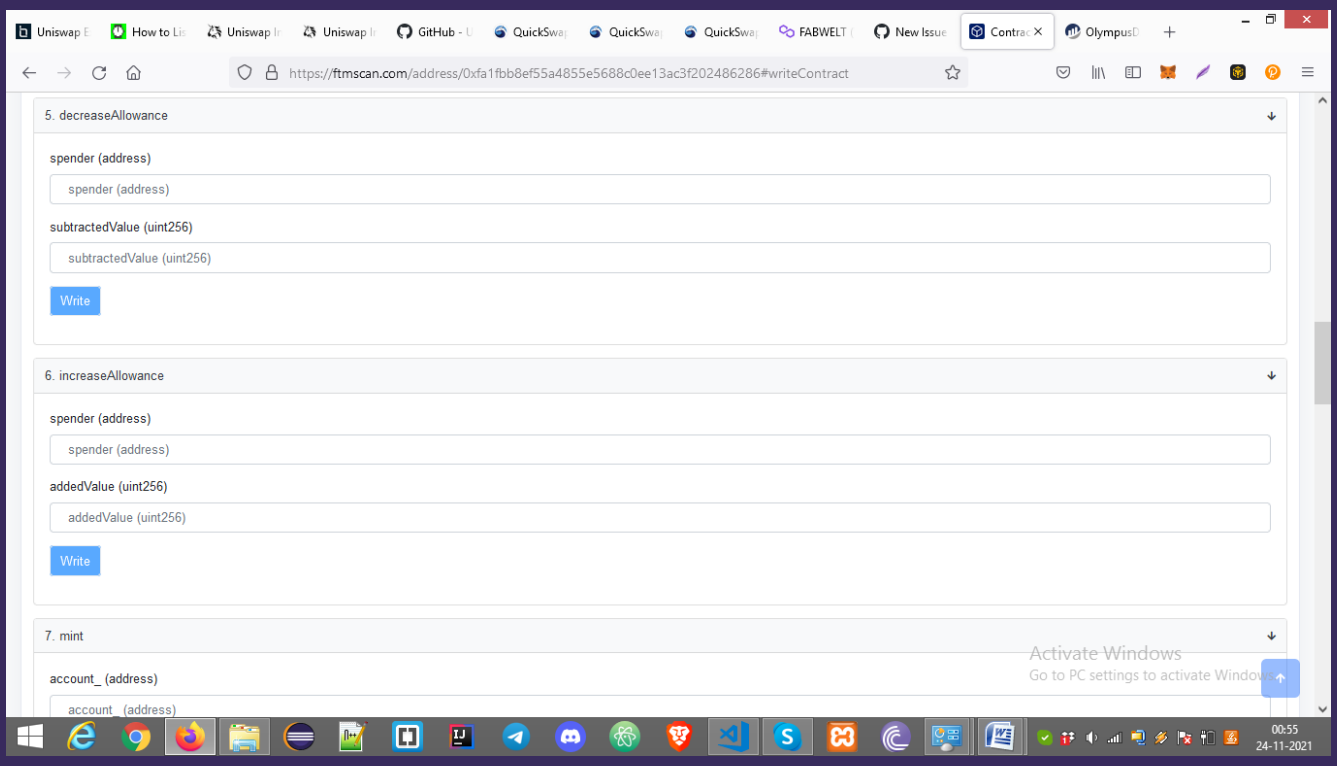
1. burnFrom

2. approve



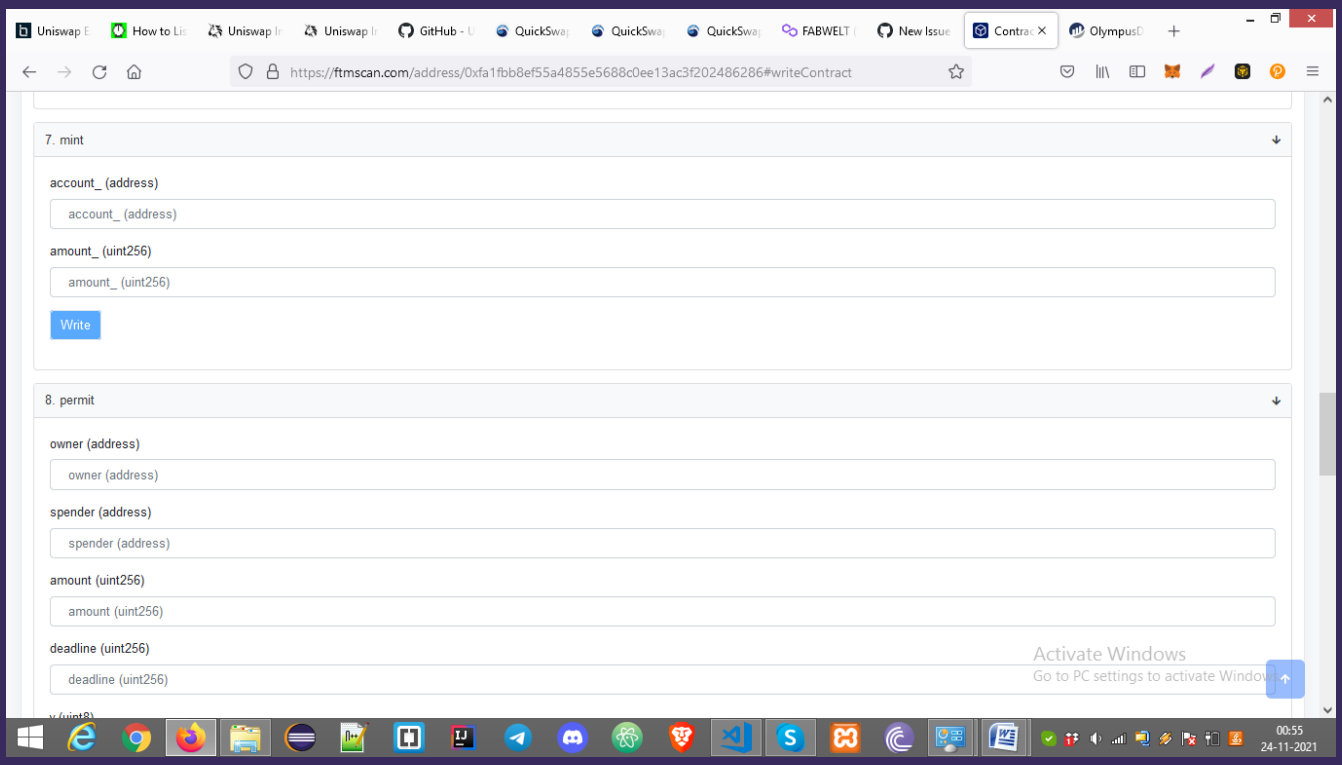
3.burn

4.burnFrom

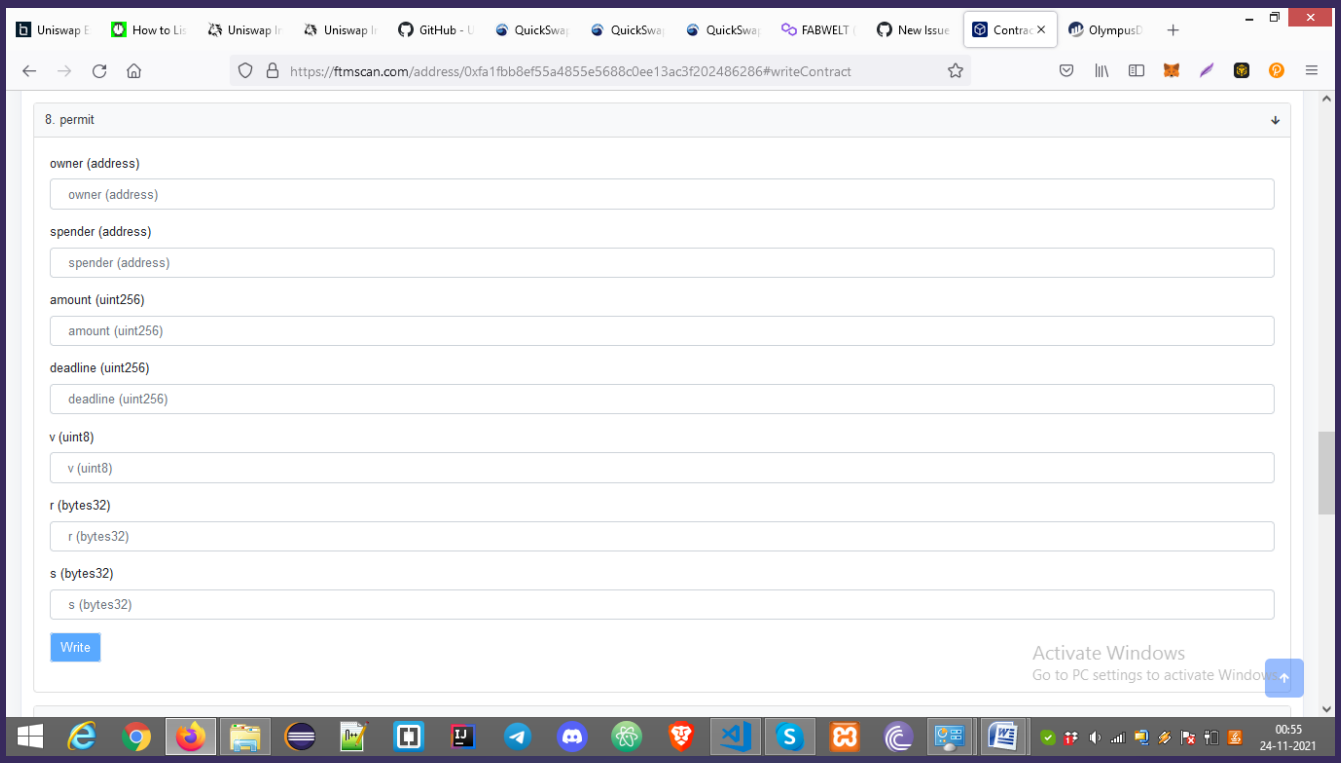


5.decreaseAllowance

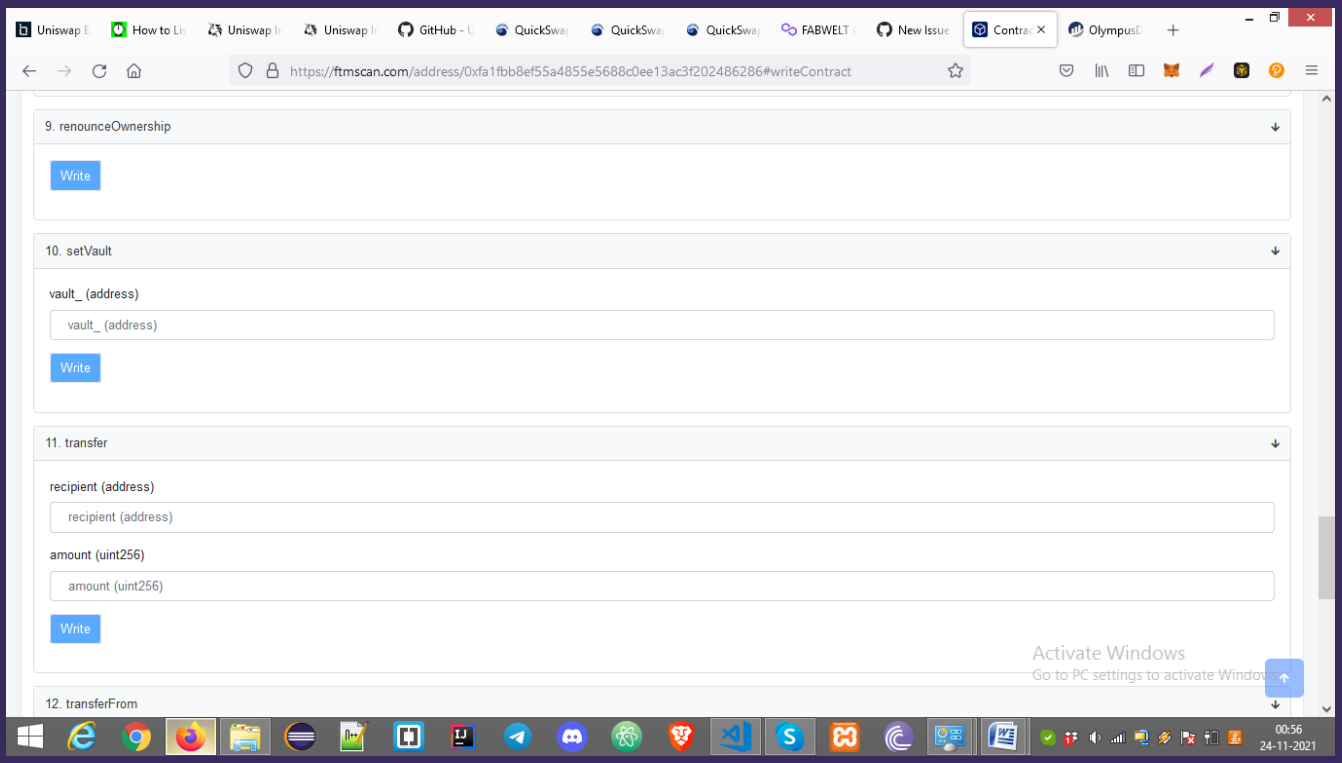
6.increaseAllowance



7.mint



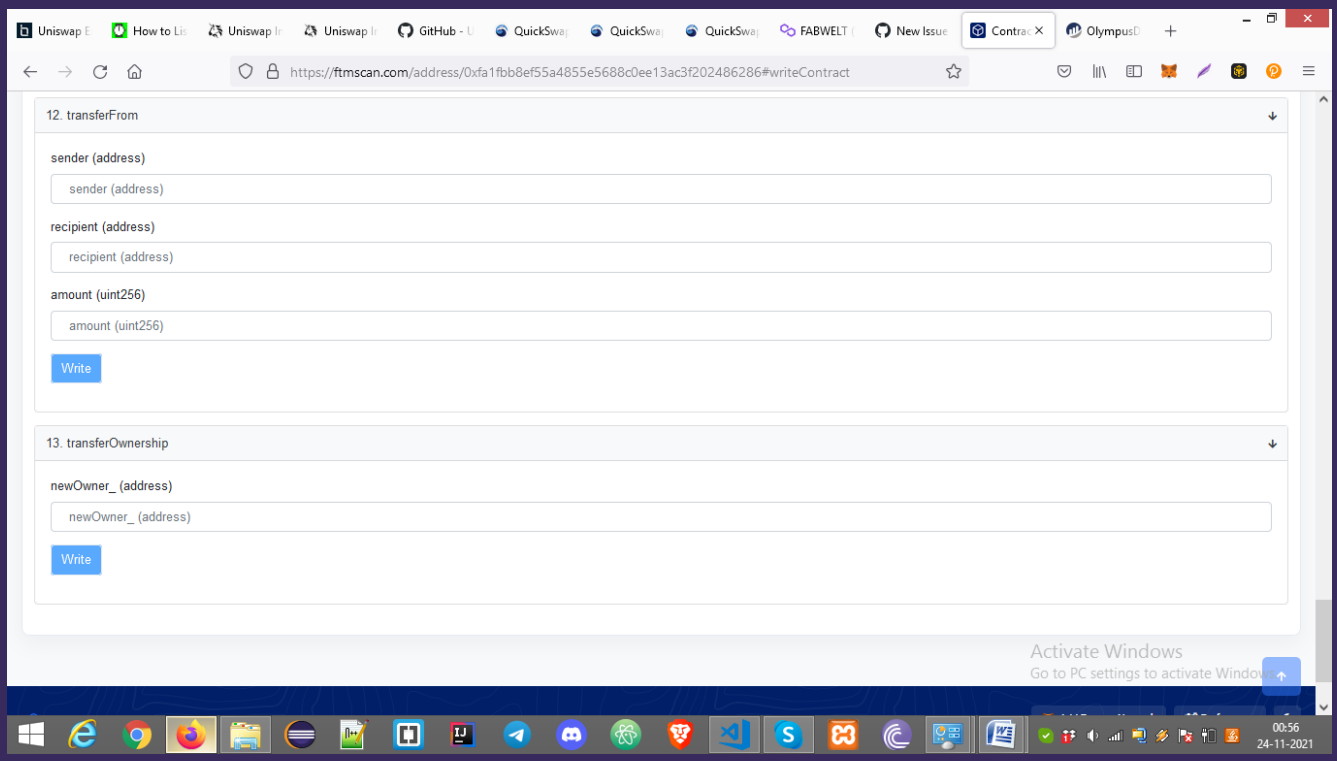
8.permit



9.renounceOwenership

10.setVault

11.transfer



12.transferFrom

13.transferOwnership

The above write functions seem to be standard functions copied from the OHM Olympus token contract. No changes have been made to any of the functions and they are all safe.

Conclusion

The Fantohm Project has been forked from OHM Olympus Project and only 4 contracts have been changed by the Fantohm team. The 4 contracts have been carefully studied for any vulnerabilities and the team found the code to be satisfactory.

Appendix

Finding Categories

Gas Optimization
Gas Optimization findings refer to exhibits that do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations
Mathematical Operation exhibits entail findings that relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue
Logical Issue findings are exhibits that detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Control Flow
Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code
Volatile Code findings refer to segments of code that behave unexpectely on certain edge cases that may result in avulnerability.

Data Flow
Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a structassignment operation affecting an in-memory struct rather than an instorage one.

Language Specific
Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete .

Coding Style
Coding Style findings usually do not affect the generated byte-code and comment on how to make the codebase more legible and as a result easily maintainable.

Inconsistency
Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers
Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error
Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

Dead Code
Code that otherwise does not affect the functionality of the codebase and can be safely omitted.