DEFENCERT
BLOCKCHAIN SECURITY

# Meta Hamster

14 Mar 2022

# Smart Contract Audit Report

www.defencert.com

# Table of Contents

## Contents

# Disclaimer

Defencert Blockchain Security (“Defencert”) has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance, or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies. Further, this audit report shall not be disclosed nor transmitted to any persons or parties on any objective, goal, or justification without due written assent, acquiescence, or approval by Defencert.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report. Information is provided ‘as is’, and Defencert is under no covenant to the completeness, accuracy, or solidity of the contracts audited. In no event will Defencert or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

Cryptocurrencies and any technologies by extension directly or indirectly related to cryptocurrencies are highly volatile and speculative by nature. All reasonable due diligence and safeguards may yet be insufficient, and users should exercise considerable caution when participating in any shape or form in this nascent industry.

The audit report has made all reasonable attempts to provide clear and articulate recommendations to the Project team with respect to the rectification, amendment, and/or revision of any highlighted issues, vulnerabilities, or exploits within the contracts provided. It is the sole responsibility of the Project team to sufficiently test and performs checks, ensuring that the contracts are functioning as intended, specifically that the functions therein contained within said contracts have the desired intended effects, functionalities, and outcomes of the Project team.

# 1. Overview

This report has been prepared for Meta Hamster on the Binance Smart Chain network. Defencert provides a user-centered examination of the smart contracts to look for vulnerabilities, logic errors, or other issues from both an internal and external perspective.

## 1.1 Summary

| | |
|---|---|
| **Project Name** | Meta Hamster |
| **URL** | https://metahamster.io/ |
| **Platform** | Binance Smart Chain |
| **Language** | Solidity |

## Contracts Assessed

| Name | Contract | Live Code Match |
|---|---|---|
| Metahamster | 0x9428f4cd18896eda03633429c3f52e5244504d14 | Yes |

## 1.2 Findings Summary

| Severity | Found |
|---|---|
| High | 0 |
| Medium | 0 |
| Low | 1 |
| Informational | 0 |
| Total | 1 |

| Severity | Description |
|---|---|
| High | Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency. |
| Medium | Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Issues under this classification are recommended to be fixed as soon as possible. |
| Low | Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless. |
| Informational | Consistency, syntax or style best practices. Generally, pose a negligible level of risk, if any. |

## 1.3 Metahamster

| ID | Severity | Summary |
|---|---|---|
| 01 | Low | A floating pragma is set. |
| 02 | Low | State variable visibility is not set. |
| 03 | Low | Use of "tx.origin" as a part of authorization control. |

# 2 Findings

## 2.1 Metahamster

Metahamster (MHAM) is a BEP20 Token in Binance Smart Chain Mainnet. Token is implemented as BEP20 smart contract. This token has 12% transaction tax which include 4% marketing, 4% rewards and 4% liquidity fees. It has additional 3% sell tax.

### 2.1.1 Token Overview

| | |
|---|---|
| **Address** | 0x9428f4cd18896eda03633429c3f52e5244504d14 |
| **Name** | Metahamster |
| **Symbol** | MHAM |
| **Token Supply** | 10,000,000,000,000,000 |
| **Decimal** | 18 |
| **Transfer Max Size** | - |
| **Transfer Min Size** | - |
| **Wallet Max Size** | 200,000,000,000,000 |
| **Max Buy Limit** | - |
| **Max Sell Limit** | 100,000,000,000,000 |
| **Transfer Fees** | 12% |
| **Buy Fees** | 12% |
| **Sell Fees** | 15% |

### 2.1.2 Privileged Roles

The following functions can be called by the OWNER of the contract:

a) Transfer and Renounce Ownership

b) Exclude from Dividends

c) Update Claim Wait

d) Process Amount

e) Set Safe Manager

f) Open Trade

g) Include to Whitelist

h) Set Fees

i) Set Extra Fee On Sell

j) Set Max Transaction Tax

k) Set Marketing Wallet Address

l) Set Max Wallet

m) Update Uniswap V2 Router

n) Exclude Fees

o) Exclude From Max Tax

p) Exclude From All

q) Set Automated Market Maker Pair

r) Set Swap Token Amount

s) Update Gas for Processing

The following functions can be called by the SAFE MANAGER of the contract:

t) Withdraw BNB

u) Withdraw BEP20

## 2.1.3 Issues & Recommendations

| | |
|---|---|
| **Issue #01** | A floating pragma is set. |
| **Severity** | Low |
| **Line** | 14 |
| **Description** | The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code. |

| | |
|---|---|
| **Issue #02** | State variable visibility is not set. |
| **Severity** | Low |
| **Line** | 1731 |
| **Description** | It is best practice to set the visibility of state variables explicitly. The default visibility for "safeManager" is internal. Other possible visibility settings are public and private. |

| | |
|---|---|
| **Issue #03** | Use of "tx.origin" as a part of authorization control. |
| **Severity** | Low |
| **Line** | 2040 |
| **Description** | Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing. |


DEFENCERT
BLOCKCHAIN SECURITY