GUARDIAN AUDITS

SMART CONTRACT SECURITY AUDIT OF

# Hamsters of Opera

# Summary - Preliminary Report

**Audit Firm -** Guardian Audits

**Client Firm** - Hamsters of Opera

**Final Report Date - Preliminary Report**

## Audit Summary

After a line by line manual analysis and automated review, Guardian Audits has concluded that:

- Hamsters of Opera's smart contracts have a MEDIUM RISK SEVERITY
- Hamsters of Opera's smart contracts have an **ACTIVE OWNERSHIP**
- Important operator privileges – `setTaxOffice`, `setTaxRate`, `setLockUp`, `setOperator`, `allocateSeignorage`, `hamsterWheelSetLockUp`, `setBondDepletionFloorPercent`, `setBootstrap`, `setDiscountPercent`, `setExtraFunds`, `setHamsterOracle`, `setHamsterPriceCeiling`, `setHamsterWheel`, `setMaxDebtRatioPercent`, `setMaxExpansionTiersEntry`, `setMaxPremiumRate`, `setMaxSupplyContractionPercent`, `setMaxSupplyExpansionPercents`, `setMintingFactorForPayingDebt`, `setPremiumPercent`, `setPremiumThreshold`, `setSupplyTiersEntry`.
- Hamsters of Opera's smart contract owner has multiple "write" privileges. Centralization risk correlated to the active ownership is **HIGH**

Notice that the examined smart contracts are not resistant to internal exploit. For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

🔗 Blockchain network: **Fantom Opera**

✅ Verify the authenticity of this report on Guardian's GitHub: https://github.com/guardianaudits

# Table of Contents

# Project Overview

## Project Summary

| | |
|---|---|
| Project Name | Hamsters Of Operator |
| Language | Solidity |
| Codebase | https://github.com/hamster-money/hamster-contracts |
| Commit | ee153c18241a0a8ff23d021e58cf318f5a849f3f |

## Audit Summary

| | |
|---|---|
| Delivery Date | Preliminary Report |
| Audit Methodology | Static Analysis, Manual Review |

## Vulnerability Summary

| Vulnerability Level | Total | Pending | Declined | Acknowledged | Partially Resolved | Resolved |
|---|---|---|---|---|---|---|
| Critical | 0 | 0 | 0 | 0 | 0 | 0 |
| High | 0 | 0 | 0 | 0 | 0 | 0 |
| Medium | 3 | 3 | 0 | 0 | 0 | 0 |
| Low | 14 | 14 | 0 | 0 | 0 | 0 |

# Audit Scope & Methodology

## Scope

| ID | File | SHA-1 Checksum |
|---|---|---|
| HAM | Hamster.sol | f0a453d6715354eb96a039212c1f24e656d4b806 |
| SHARE | HShare.sol | 6c560db0dd2f3dad1553852245bd4c632f4e5941 |
| BOND | HBond.sol | a803852d69a6af4dbefb151f994857217ecba818 |
| REWARD | HamsterRewardPool.sol | 67ea84224f686182ee5f9a739a8ebbc913faa7a4 |
| WHEEL | HamsterWheel.sol | b8f137ec23368b6cb82ba9f75ee3aa0458f18fad |
| TRS | Treasury.sol | 9a196a6c3b6b5770d45bb8ecca32706cf7ef0abc |
| TAX | TaxOfficeV2.sol | 534f49026453d5e44c3d368d4ab28a8733de3694 |
| ZAP | HamsterZapper.sol | a269d20a7aa8e42766bb8c47d64a6f0212cacf5c |

## Methodology

The auditing process pays special attention to the following considerations:

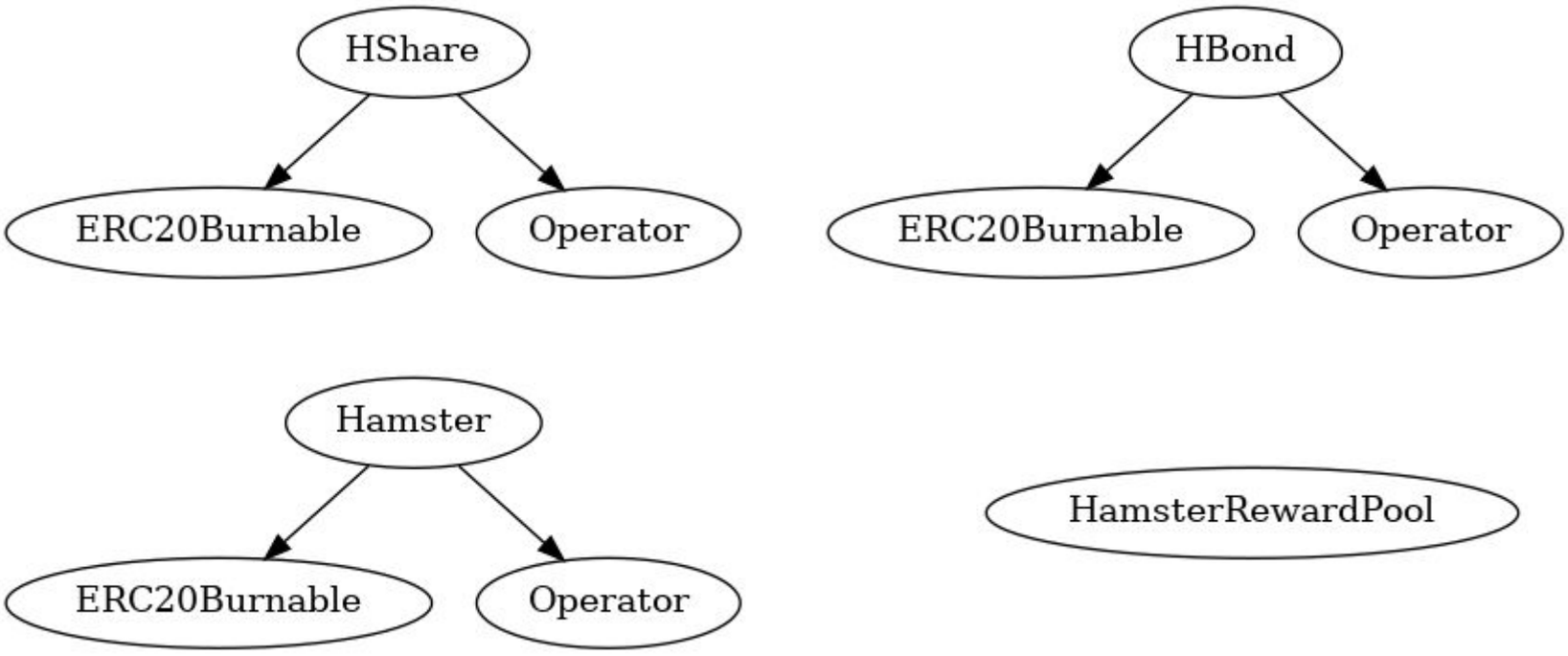
- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

# Audit Scope & Methodology

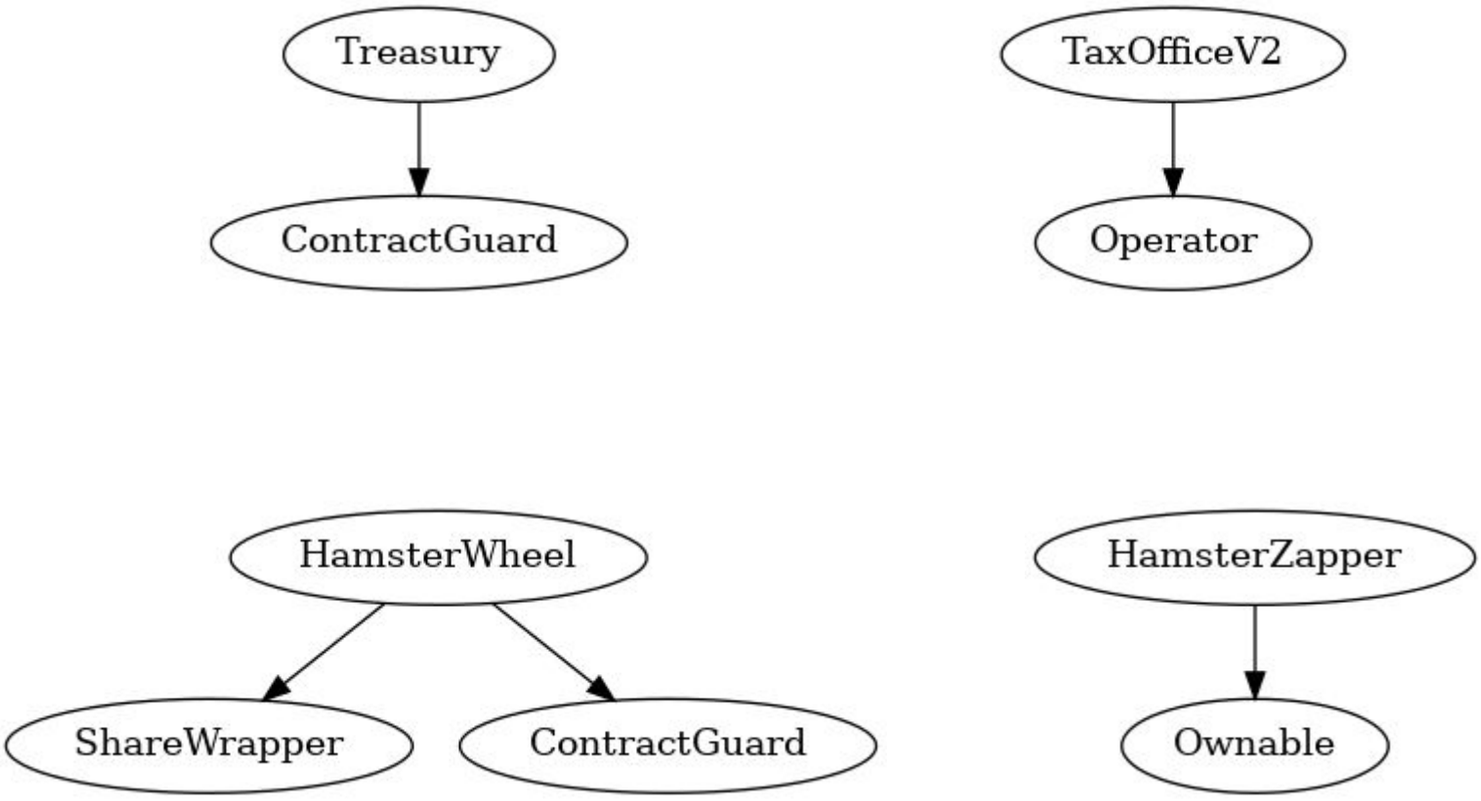
## Vulnerability Classifications

| Vulnerability Level | Classification |
| --- | --- |
| 🔴 Critical | Easily exploitable by anyone, causing loss/manipulation of assets or data. |
| 🟠 High | Arduously exploitable by a subset of addresses, causing loss/manipulation of assets or data. |
| 🟡 Medium | Inherent risk of future exploits that may or may not impact the smart contract execution. |
| 🟢 Low | Minor deviation from best practices. |

# Inheritance Graph - Protocol Tokens And Pool



# Inheritance Graph - Protocol Core And Zapper



# Findings & Resolutions

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| HAM-1 | Uncapped Tax | Centralization / Privilege | Medium | Unresolved |
| HAM-2 | Unchecked Return Value | Control Flow | Low | Unresolved |
| SHARE-1 | Block Timestamp | Tx Manipulation | Low | Unresolved |
| SHARE-2 | Immutability Modifiers | Mutability | Low | Unresolved |
| REWARD-1 | Block Timestamp | Tx Manipulation | Low | Unresolved |
| REWARD-2 | Memory Usage | Optimization | Low | Unresolved |
| REWARD-3 | Immutability Modifiers | Mutability | Low | Unresolved |
| REWARD-4 | `poolInfo` Denial of Service | Denial of Service | Low | Unresolved |
| WHEEL-1 | Arbitrary Lockup | Centralization / Privilege | Low | Unresolved |
| WHEEL-2 | Missing Events | Events | Low | Unresolved |
| TRS-1 | Immutability Modifiers | Mutability | Low | Unresolved |
| TRS-2 | Centralization Risk | Centralization / Privilege | Medium | Unresolved |
| TAX-1 | Immutability Modifiers | Mutability | Low | Unresolved |

# Findings & Resolutions

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| TAX-2 | Tax Inclusion Manipulation | Centralization / Privilege | Low | Unresolved |
| TAX-3 | Unchecked  Return Value | Control Flow | Low | Unresolved |
| TAX-4 | Centralization Risk | Centralization / Privilege | Medium | Unresolved |
| TAX-5 | Missing Events | Events | Low | Unresolved |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

# HAM-1 | Uncapped Tax

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | Medium | Hamster.sol:159 | Unresolved |

## Description

The `setTaxRate` function allows for a tax as high as 99.99% to be imposed, which can lead to near total loss of funds for users.

## Recommendation

Require a more strict cap on the `taxRate` and/or timelock the `setTaxRate` function.

## Resolution

# HAM-2 | Unchecked Return Value

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Control Flow | ● Low | Hamster.sol:122 | Unresolved |

## Description

The `governanceRecoverUnsupported` function uses `transfer` which provides a return value that should be checked. Not all ERC20 implementations revert in case of failure, so it is important to have some logic in the event these executions fail.

## Recommendation

Check the return value, or opt for a `safeTransfer` alternative.

## Resolution

# SHARE-1 | Block Timestamp

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Tx Manipulation | ● Low | HShare.sol: 25, 32, 68, 73 | Unresolved |

## Description

Possibly dangerous reliance on `block.timestamp`. `block.timestamp` can be manipulated by validators.

## Recommendation

Rely on `block.number` instead, or ensure resilience to `block.timestamp` manipulation.

## Resolution

# SHARE-2 | Immutability Modifiers

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Mutability | ● Low | HShare.sol | Unresolved |

## Description

The `communityFundAllocation`, `devFundAllocation`, `vestingDuration`, `startTime`, `endTime`, `communityFundRewardRate`, and `devFundRewardRate` variables are only set in the constructor, and should therefore be declared `immutable`.

## Recommendation

Declare them as `immutable`.

## Resolution

# REWARD-1 | Block Timestamp

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Tx Manipulation | Low | HamsterRewardPool.sol: 39, 40, 95, 123, 132, 133, 136, 184, 236, 241, 249, 253 | Unresolved |

## Description

Possibly dangerous reliance on `block.timestamp`. `block.timestamp` can be manipulated by validators.

## Recommendation

Rely on `block.number` instead, or ensure resilience to `block.timestamp` manipulation.

## Resolution

# REWARD-2 | Memory Usage

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Optimization | Low | HamsterRewardPool.sol: 35, 151, 170, 188, 210 | Unresolved |

## Description

The `pool` variable is often declared `storage` when it is not modified.

## Recommendation

Declare it as `memory` to save on gas.

## Resolution

# REWARD-3 | Immutability Modifiers

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Mutability | Low | HamsterRewardPool.sol | Unresolved |

## Description

The `hamster` and `poolStartTime` variables are only set in the constructor, and should therefore be declared `immutable`.

## Recommendation

Declare them as `immutable`.

## Resolution

# REWARD-4 | poolInfo Denial of Service

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Denial of Service | Low | HamsterRewardPool.sol | Unresolved |

## Description

The `operator` can use `add` to extend the `poolInfo` list. If `poolInfo` becomes significantly long it can cause high gas consumption for the `governanceRecoverUnsupported`, `massUpdatePools`, `checkPoolDuplicate`, `add`, and `set` functions.

If the gas consumption were to exceed the transaction limit as a result, these functions would be rendered useless.

## Recommendation

Timelock the `add` function or limit the maximum size of `poolInfo`.

## Resolution

# WHEEL-1 | Arbitrary Lockup

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | ● Low | HamsterWheel.sol: 124 | Unresolved |

## Description

Using `setLockup`, the `operator` can arbitrarily set the `withdrawLockupEpochs` and `rewardLockupEpochs` as high as 56 epochs retroactively after an address has locked.

## Recommendation

Timelock the `setLockup` sufficiently such that all current locks can become unlocked before the new lockup is applied, or refactor the logic such that the new lockup only applies to new lockers.

## Resolution

# WHEEL-2 | Missing Events

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Events | Low | HamsterWheel.sol | Unresolved |

## Description

The `setOperator` and `setLockup` functions change state that affects stakeholders so they should emit corresponding events.

## Recommendation

Add event emissions to `setOperator` and `setLockup`.

## Resolution

# TRS-1 | Immutability Modifiers

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Mutability | ● Low | Treasury.sol | Unresolved |

## Description

The `hamster` and `poolStartTime` variables are only set in the constructor, and should therefore be declared `immutable`.

## Recommendation

Declare them as `immutable`.

## Resolution

# TRS-2 | Centralization Risk

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | Medium | Treasury.sol | Unresolved |

## Description

The `operator` address is not a multi-sig and has potentially dangerous permissions for `hamsterWheelSetOperator`, `hamsterWheelAllocateSeigniorage`, `hamsterWheelSetLockUp`, `setBondDepletionFloorPercent`, `setBootstrap`, `setDiscountPercent`, `setExtraFunds`, `setHamsterOracle`, `setHamsterPriceCeiling`, `setHamsterWheel`, `setMaxDebtRatioPercent`, `setMaxExpansionTiersEntry`, `setMaxPremiumRate`, `setMaxSupplyContractionPercent`, `setMaxSupplyExpansionPercents`, `setMintingFactorForPayingDebt`, `setPremiumPercent`, `setPremiumThreshold`, `setSupplyTiersEntry`

## Recommendation

Make the `operator` a multi-sig and/or introduce a timelock for the community to monitor events.

## Resolution

# TAX-1 | Immutability Modifiers

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Mutability | Low | TaxOfficeV2.sol: 124 | Unresolved |

## Description

The `hamster` and `router` variables are only set in the constructor, and should therefore be declared `immutable` for gas optimization.

## Recommendation

Declare them as `immutable`.

## Resolution

# TAX-2 | Tax Inclusion Manipulation

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | ● Low | TaxOfficeV2.sol | Unresolved |

## Description

If `setTaxExclusionForAddress` was called by the `operator` and set to true for an address, that address can call `taxFreeTransferFrom` with an excluded sender. Afterwards, the sender would be included in the tax.

## Recommendation

Introduce a timelock such that the community can monitor what the `operator` sets.

## Resolution

# TAX-3 | Unchecked Return Value

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Control Flow | Low | TaxOfficeV2.sol: 84, 102 | Unresolved |

## Description

The `addLiquidityTaxFree` function uses `transfer` which provides a return value that should be checked. Not all ERC20 implementations  revert in case of failure, it is important to have some logic in the event these executions fail.

## Recommendation

Check the return value, or opt for a `safeTransfer` alternative.

## Resolution

# TAX-4 | Centralization Risk

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Centralization / Privilege | Medium | TaxOfficeV2.sol | Unresolved |

## Description

The `operator` address is not a multi-sig and has potentially dangerous permissions for `disableAutoCalculateTax`, `enableAutoCalculateTax`, `excludeAddressFromTax`, `includeAddressInTax`, `setBurnThreshold`, `setTaxCollectorAddress`, `setTaxExclusionForAddress`, `setTaxRate`, `setTaxTiersRate`, `setTaxTiersTwap`, `setTaxableHamsterOracle`, `transferTaxOffice`

Most notably `transferTaxOffice` sets the `taxOffice` for the `hamster` contract, potentially compromising the `hamster` `taxOffice` permissioned functions as well.

## Recommendation

Make the `operator` a multi-sig and/or introduce a timelock for the community to monitor events.

## Resolution

# TAX-5 | Missing Events

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Events | ● Low | TaxOfficeV2.sol | Unresolved |

## Description

The `disableAutoCalculateTax`, `enableAutoCalculateTax`, `excludeAddressFromTax`, `includeAddressInTax`, `setBurnThreshold`, `setTaxCollectorAddress`, `setTaxExclusionForAddress`, `setTaxRate`, `setTaxTiersRate`, `setTaxTiersTwap`, `setTaxableHamsterOracle`, `transferTaxOffice` functions change state that affects stakeholders so they should emit corresponding events.

## Recommendation

Add event emissions to these functions.

## Resolution

# Auditor’s Verdict

After a line by line manual analysis and automated review, Guardian Audits has concluded that:

- Hamsters of Opera’s  smart contracts have a MEDIUM RISK SEVERITY
- Hamsters of Opera’s smart contracts have an **ACTIVE OWNERSHIP**
- Important operator  privileges – `setTaxOffice`, `setTaxRate`, `setLockUp`, `setOperator`, `allocateSeignorage`, `hamsterWheelSetLockUp`, `setBondDepletionFloorPercent`, `setBootstrap`, `setDiscountPercent`, `setExtraFunds`, `setHamsterOracle`, `setHamsterPriceCeiling`, `setHamsterWheel`, `setMaxDebtRatioPercent`, `setMaxExpansionTiersEntry`, `setMaxPremiumRate`, `setMaxSupplyContractionPercent`, `setMaxSupplyExpansionPercents`, `setMintingFactorForPayingDebt`, `setPremiumPercent`, `setPremiumThreshold`, `setSupplyTiersEntry`.
- Hamsters of Opera’s smart contract owner has multiple “write” privileges. Centralization risk correlated to the active ownership is **HIGH**

# Disclaimer

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is 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 Guardian to perform a security assessment. This report does 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.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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. Guardian’s position is that each company and individual are responsible for their own due diligence and continuous security. Guardian’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.

The assessment services provided by Guardian is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract’s safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

# About Guardian Audits

Founded in 2022 by DeFi experts, Guardian Audits is a leading audit firm in the DeFi smart contract space. With every audit report, Guardian Audits upholds best-in-class security while achieving our mission to relentlessly secure DeFi.

To learn more, visit https://guardianaudits.com

To view our audit portfolio, visit https://github.com/guardianaudits

To book an audit, message https://t.me/guardianaudits