



SAFU Audit

Smart Contract Auditing

HARUNO INU

SMART CONTRACT AUDIT



April 2, 2022

INTRODUCTION

| | |
|-------------------------|---|
| Client | Haruno Inu (HARUNO) |
| Language | Solidity |
| Contract address | 0x182082003E88b5203D1D2eeB1E102d948CA6d900 |
| Decimals | 5 |
| Supply | 5,000,000 |
| Platform | Binance Smart Chain |
| Compiler | v0.7.4+commit.3f05b770 |
| Optimization | Yes, with 200 runs |
| Website | https://harunoinu.com/ |
| Telegram | https://t.me/harunoinu |
| Twitter | https://twitter.com/haruno_inu |

Description

Haruno Inu is inspired by the character Haruno Sakura - one of the greatest medical-nin in the world. Haruno Inu is a decentralized financial asset that pays users with a fixed compound interest formula that is sustainable.

Haruno Inu's objective is to bring more benefits to the community through many of our competitive features as a community-managed deflationary token.

TABLE OF CONTENTS

01 INTRODUCTION

| | |
|---------------------|----|
| Introduction | 02 |
| Approach | 04 |
| Risk classification | 05 |

02 ABSTRACT

| | |
|----------|----|
| Abstract | 06 |
|----------|----|

03 VULNERABILITIES TEST

| | |
|----------------------|----|
| Vulnerabilities Test | 07 |
|----------------------|----|

04 MANUAL ANALYSIS

| | |
|---------------------|----|
| Manual analysis | 09 |
| Contract Inspection | 10 |
| Inheritance Tree | 14 |
| Important Snippets | 15 |
| Good Practices | 16 |

05 WEBSITE

| | |
|---------------|----|
| Website Audit | 17 |
|---------------|----|

06 CONCLUSIONS

| | |
|---------------|----|
| Disclaimer | 18 |
| Audit Results | 19 |
| Score | 20 |
| Summary | 21 |

Approach



Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.



Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.



Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
 - Back-doors
 - Vulnerability
 - Accuracy
 - Readability
-



Tools

- Remix IDE
- MythX, Myhrlil
- SWC Registry
- Open Zeppelin Code Analyzer
- Solidity Code Complier

RISK CLASSIFICATION

CRITICAL

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

MEDIUM

Issues on this level could potentially bring problems and should eventually be fixed.

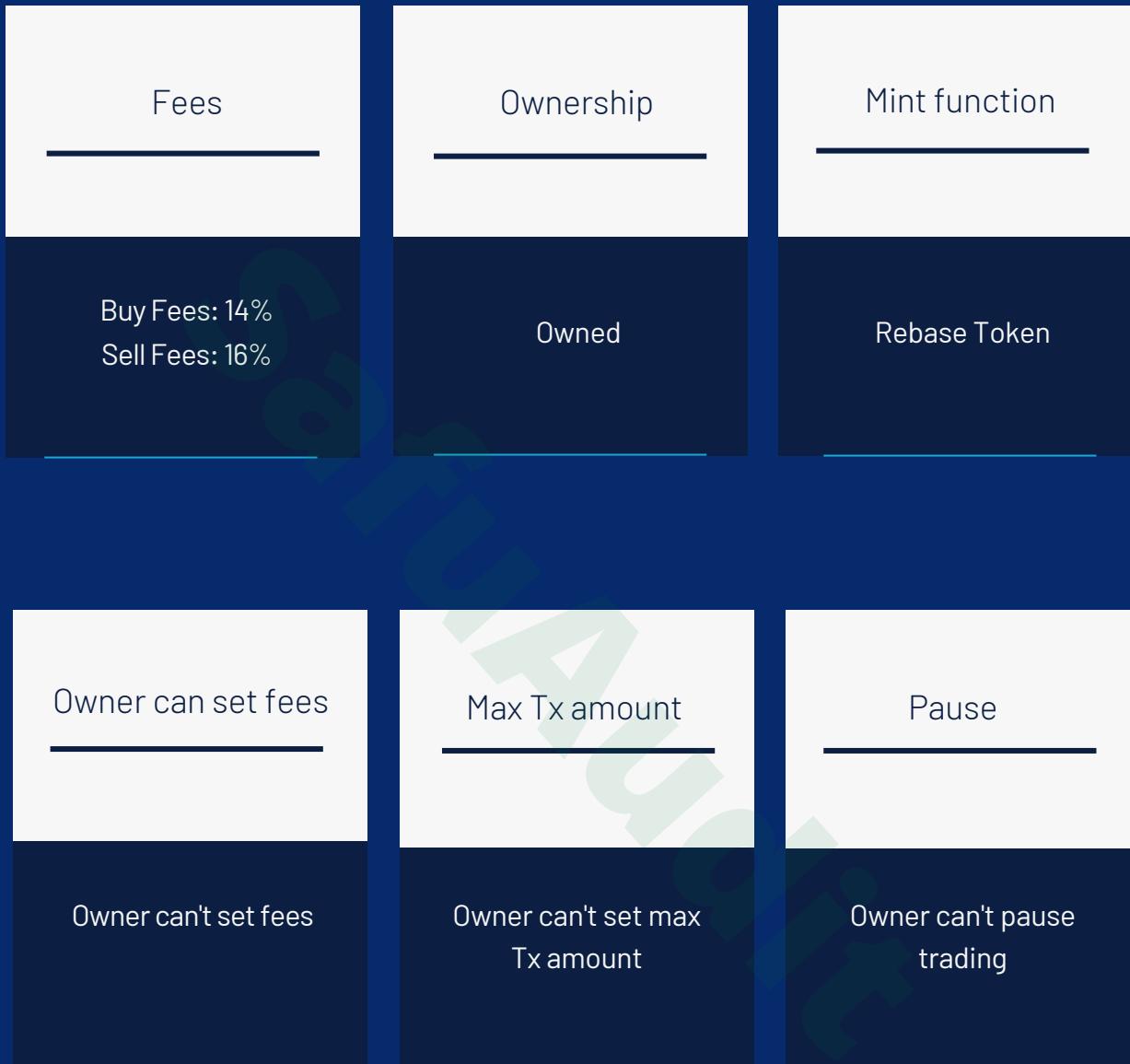
MINOR

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

INFORMATIONAL

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.

ABSTRACT



Vulnerabilities Test

| SWC ID | Description | |
|----------------|---------------------------------------|---------------|
| SWC-100 | Function Default Visibility | Passed |
| SWC-101 | Integer Overflow and Underflow | Passed |
| SWC-102 | Outdated Compiler Version | Passed |
| SWC-103 | FloatingPragma | Minor |
| SWC-104 | Unchecked Call Return Value | Passed |
| SWC-105 | Unprotected Ether Withdrawal | Passed |
| SWC-106 | Unprotected SELF-DESTRUCT Instruction | Passed |
| SWC-107 | Re-entrancy | Passed |
| SWC-108 | State Variable Default Visibility | Minor |
| SWC-109 | Uninitialized Storage Pointer | Passed |
| SWC-110 | Assert Violation | Passed |
| SWC-111 | Use of Deprecated Solidity Functions | Passed |
| SWC-112 | Delegate Call to Untrusted Callee | Passed |
| SWC-113 | DoS with Failed Call | Passed |
| SWC-114 | Transaction Order Dependence | Passed |
| SWC-115 | Authorization through tx.origin | Passed |

| | | |
|----------------|---|---------------|
| SWC-116 | Block values as a proxy for time | Passed |
| SWC-117 | Signature Malleability | Passed |
| SWC-118 | Incorrect Constructor Name | Passed |
| SWC-119 | Shadowing State Variables | Passed |
| SWC-120 | Weak Sources of Randomness from Chain Attributes | Passed |
| SWC-121 | Missing Protection against Signature Replay Attacks | Passed |
| SWC-122 | Lack of Proper Signature Verification | Passed |
| SWC-123 | Requirement Violation | Passed |
| SWC-124 | Write to Arbitrary Storage Location | Passed |
| SWC-125 | Incorrect Inheritance Order | Passed |
| SWC-126 | Insufficient Gas Griefing | Passed |
| SWC-127 | Arbitrary Jump with Function Type Variable | Passed |
| SWC-128 | DoS With Block Gas Limit | Passed |
| SWC-129 | Typographical Error | Passed |
| SWC-130 | Right-To-Left-Override control character (U+202E) | Passed |
| SWC-131 | Presence of unused variables | Passed |
| SWC-132 | Unexpected Ether balance | Passed |
| SWC-133 | Hash Collisions With Multiple Variable Length Arguments | Passed |
| SWC-134 | Message call with the hardcoded gas amount | Passed |
| SWC-135 | Code With No Effects (Irrelevant/Dead Code) | Passed |
| SWC-136 | Unencrypted Private Data On-Chain | Passed |

MANUAL ANALYSIS

The contract is verified to check if functions do and work as they should and malicious code is not inserted.

| | Tested | Result |
|---------------------------|--------|---------------|
| Transfer | Yes | Passed |
| Total Supply | Yes | Passed |
| Buy Back | Yes | N/A |
| Burn | Yes | N/A |
| Mint | Yes | N/A |
| Rebase | Yes | Passed |
| Pause | Yes | N/A |
| Blacklist | Yes | Passed |
| Lock | Yes | N/A |
| Max Transaction | Yes | N/A |
| Transfer Ownership | Yes | Passed |
| Renounce Ownership | Yes | Passed |

MANUAL AUDIT

CONTRACT INSPECTION



```
| **SafeMathInt** | Library | |||
| L | mul | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | add | Internal 🔒 | |||
| L | abs | Internal 🔒 | |||
||||||

| **SafeMath** | Library | |||
| L | add | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | mul | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | mod | Internal 🔒 | |||
||||||

| **IERC20** | Interface | |||
| L | totalSupply | External 🚫 | |NO| |
| L | balanceOf | External 🚫 | |NO| |
| L | allowance | External 🚫 | |NO| |
| L | transfer | External 🚫 | |NO| |
| L | approve | External 🚫 | |NO| |
| L | transferFrom | External 🚫 | |NO| |
||||||

| **IPancakeSwapPair** | Interface | |||
| L | name | External 🚫 | |NO| |
| L | symbol | External 🚫 | |NO| |
| L | decimals | External 🚫 | |NO| |
| L | totalSupply | External 🚫 | |NO| |
| L | balanceOf | External 🚫 | |NO| |
| L | allowance | External 🚫 | |NO| |
| L | approve | External 🚫 | |NO| |
| L | transfer | External 🚫 | |NO| |
| L | transferFrom | External 🚫 | |NO| |
| L | DOMAIN_SEPARATOR | External 🚫 | |NO| |
| L | PERMIT_TYPEHASH | External 🚫 | |NO| |
| L | nonces | External 🚫 | |NO| |
| L | permit | External 🚫 | |NO| |
```

```
| L | MINIMUM_LIQUIDITY | External | | NO | |
| L | factory | External | | NO | |
| L | token0 | External | | NO | |
| L | token1 | External | | NO | |
| L | getReserves | External | | NO | |
| L | price0CumulativeLast | External | | NO | |
| L | price1CumulativeLast | External | | NO | |
| L | kLast | External | | NO | |
| L | mint | External | | ○ | NO | |
| L | burn | External | | ○ | NO | |
| L | swap | External | | ○ | NO | |
| L | skim | External | | ○ | NO | |
| L | sync | External | | ○ | NO | |
| L | initialize | External | | ○ | NO | |
|||||||
| **IPancakeSwapRouter** | Interface | ||
| L | factory | External | | NO | |
| L | WETH | External | | NO | |
| L | addLiquidity | External | | ○ | NO | |
| L | addLiquidityETH | External | | □ | NO | |
| L | removeLiquidity | External | | ○ | NO | |
| L | removeLiquidityETH | External | | ○ | NO | |
| L | removeLiquidityWithPermit | External | | ○ | NO | |
| L | removeLiquidityETHWithPermit | External | | ○ | NO | |
| L | swapExactTokensForTokens | External | | ○ | NO | |
| L | swapTokensForExactTokens | External | | ○ | NO | |
| L | swapExactETHForTokens | External | | □ | NO | |
| L | swapTokensForExactETH | External | | ○ | NO | |
| L | swapExactTokensForETH | External | | ○ | NO | |
| L | swapETHForExactTokens | External | | □ | NO | |
| L | quote | External | | NO | |
| L | getAmountOut | External | | NO | |
| L | getAmountIn | External | | NO | |
| L | getAmountsOut | External | | NO | |
| L | getAmountsIn | External | | NO | |
| L | removeLiquidityETHSupportingFeeOnTransferTokens | External | | ○ | NO | |
| L | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External | | ○ | NO | |
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External | | ○ | NO | |
| L | swapExactETHForTokensSupportingFeeOnTransferTokens | External | | □ | NO | |
| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External | | ○ | NO |
```

```
| **IPancakeSwapFactory** | Interface | |||
| L | feeTo | External | | NO | |
| L | feeToSetter | External | | NO | |
| L | getPair | External | | NO | |
| L | allPairs | External | | NO | |
| L | allPairsLength | External | | NO | |
| L | createPair | External | | ● | NO | |
| L | setFeeTo | External | | ● | NO | |
| L | setFeeToSetter | External | | ● | NO | |
||||||

| **Ownable** | Implementation | |||
| L | <Constructor> | Public | | ● | NO | |
| L | owner | Public | | NO | |
| L | isOwner | Public | | NO | |
| L | renounceOwnership | Public | | ● | onlyOwner | |
| L | transferOwnership | Public | | ● | onlyOwner | |
| L | _transferOwnership | Internal | 🔒 | ● | |
||||||

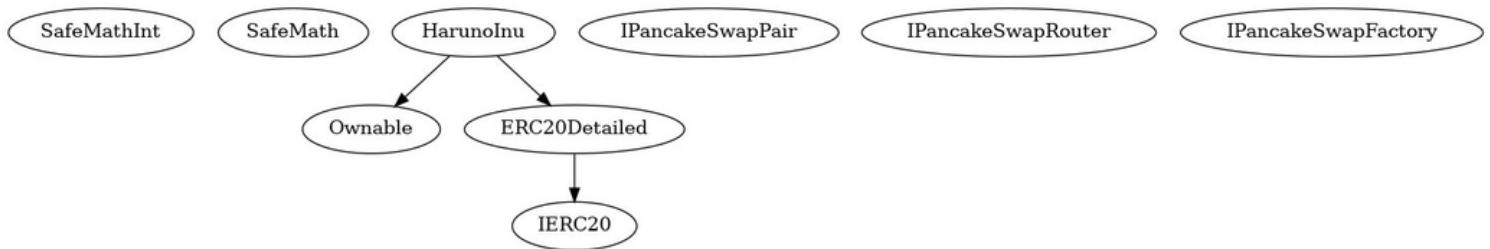
| **ERC20Detailed** | Implementation | IERC20 |||
| L | <Constructor> | Public | | ● | NO | |
| L | name | Public | | NO | |
| L | symbol | Public | | NO | |
| L | decimals | Public | | NO | |
||||||

| **Harunolnu** | Implementation | ERC20Detailed, Ownable |||
| L | <Constructor> | Public | | ● | ERC20Detailed Ownable | |
| L | startRebase | External | | ● | onlyOwner | |
| L | rebase | Internal | 🔒 | ● | |
| L | transfer | External | | ● | validRecipient | |
| L | transferFrom | External | | ● | validRecipient | |
| L | _basicTransfer | Internal | 🔒 | ● | |
| L | _transferFrom | Internal | 🔒 | ● | |
| L | takeFee | Internal | 🔒 | ● | |
| L | addLiquidity | Internal | 🔒 | ● | swapping | |
| L | swapBack | Internal | 🔒 | ● | swapping | |
| L | withdrawAllToTreasury | External | | ● | swapping onlyOwner | |
| L | shouldTakeFee | Internal | 🔒 | | |
| L | shouldRebase | Internal | 🔒 | | |
| L | shouldAddLiquidity | Internal | 🔒 | | |
| L | shouldSwapBack | Internal | 🔒 | |
```

| | | | | | | | | |
|---|----------------------|----------|---|--|--|--|-----------|---|
| L | setAutoRebase | External | ! | | | | onlyOwner | |
| L | setAutoAddLiquidity | External | ! | | | | onlyOwner | |
| L | allowance | External | ! | | | | NO | ! |
| L | decreaseAllowance | External | ! | | | | NO | ! |
| L | increaseAllowance | External | ! | | | | NO | ! |
| L | approve | External | ! | | | | NO | ! |
| L | checkFeeExempt | External | ! | | | | NO | ! |
| L | getCirculatingSupply | Public | ! | | | | NO | ! |
| L | isNotInSwap | External | ! | | | | NO | ! |
| L | manualSync | External | ! | | | | NO | ! |
| L | setFeeReceivers | External | ! | | | | onlyOwner | |
| L | getLiquidityBacking | External | ! | | | | NO | ! |
| L | setWhitelist | External | ! | | | | onlyOwner | |
| L | setBotBlacklist | External | ! | | | | onlyOwner | |
| L | setPairAddress | External | ! | | | | onlyOwner | |
| L | setLP | External | ! | | | | onlyOwner | |
| L | totalSupply | External | ! | | | | NO | ! |
| L | balanceOf | External | ! | | | | NO | ! |
| L | isContract | Internal | | | | | | |
| L | <Receive Ether> | External | ! | | | | NO | ! |

| Symbol | Meaning |
|--------|---------------------------|
| | Function can modify state |
| | Function is payable |
| | Private function |
| | Internal function |
| | Function has no modifier |

INHERITANCE TREE



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

Important Snippets



Owner can start rebase only once, at any time

```
function startRebase() external onlyOwner {
    // execute only once
    require(!_isRebaseStarted, "Rebase already started");
    if (_isRebaseStarted) return;
    _initRebaseStartTime = block.timestamp;
    _lastRebasedTime = block.timestamp;
    _autoRebase = true;
    _isRebaseStarted = true;
}
```

Auto rebase

```
function setAutoRebase(bool _flag) external onlyOwner {
    if (_flag) {
        _autoRebase = _flag;
        _lastRebasedTime = block.timestamp;
    } else {
        _autoRebase = _flag;
    }
}
```

Blacklisted contracts are not permitted to transfer their tokens

```
function setBotBlacklist(address _botAddress, bool _flag)
    external
    onlyOwner
{
    require(
        isContract(_botAddress),
        "only contract address, not allowed externally owned account"
    );
    blacklist[_botAddress] = _flag;
}
```

Transfer tokens left in contract to revenue address

```
function withdrawAllToTreasury() external swapping onlyOwner {
    uint256 amountToSwap = _gonBalances[address(this)].div(
        _gonsPerFragment
    );
    require(
        amountToSwap > 0,
        "There is no Haruno Inu token deposited in token contract"
    );
    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = router.WETH();
    router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        amountToSwap,
        0,
        path,
        harunoTreasuryReceiver,
        block.timestamp
    );
}
```

GOOD PRACTICES



- The owner cannot stop or pause the smart contract
- The owner cannot set the fees
- The owner cannot set max Tx
- The smart contract utilizes "SafeMath" to prevent overflows

```
function add(uint256 a, uint256 b) internal pure returns (uint256) {
    uint256 c = a + b;
    require(c >= a, "SafeMath: addition overflow");
    return c;
}
function sub(
    uint256 a,
    uint256 b,
    string memory errorMessage
) internal pure returns (uint256) {
    require(b <= a, errorMessage);
    uint256 c = a - b;
    return c;
}
function mul(uint256 a, uint256 b) internal pure returns (uint256) {
    if (a == 0) {
        return 0;
    }
    uint256 c = a * b;
    require(c / a == b, "SafeMath: multiplication overflow");
    return c;
}
function div(
    uint256 a,
    uint256 b,
    string memory errorMessage
) internal pure returns (uint256) {
    require(b != 0, errorMessage);
    uint256 c = a / b;
    return c;
}
function mod(uint256 a, uint256 b) internal pure returns (uint256) {
    require(b != 0);
    return a % b;
}
```

WEBSITE



| | |
|-----------------------------------|---|
| Website | https://harunoinu.com/ |
| Domain Registry | http://www.namecheap.com |
| Domain Expiry Date | 2023-03-27 |
| Response Code | 200 |
| SSL Checker and HTTPS Test | Passed |
| Deprecated HTML tags | Informational - Center tag |
| Robots.txt | Informational |
| Sitemap Test | Informational |
| SEO Friendly URL | Passed |
| Responsive Test | Passed |
| JS Error Test | Passed |
| Console Errors Test | Passed |
| Site Loading Speed Test | 4.21 seconds - Passed |
| HTTP2 Test | Passed |
| Safe Browsing Test | Passed |

DISCLAIMER

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

Accuracy of Information

SafuAudit will strive to ensure accuracy of information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyse the on-chain smart contract source code, and to provide basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only – we recommend proceeding with several independent audits. Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

AUDIT RESULTS

CRITICAL

- No critical severity issues have been found.

MEDIUM

- No medium severity issues have been found.

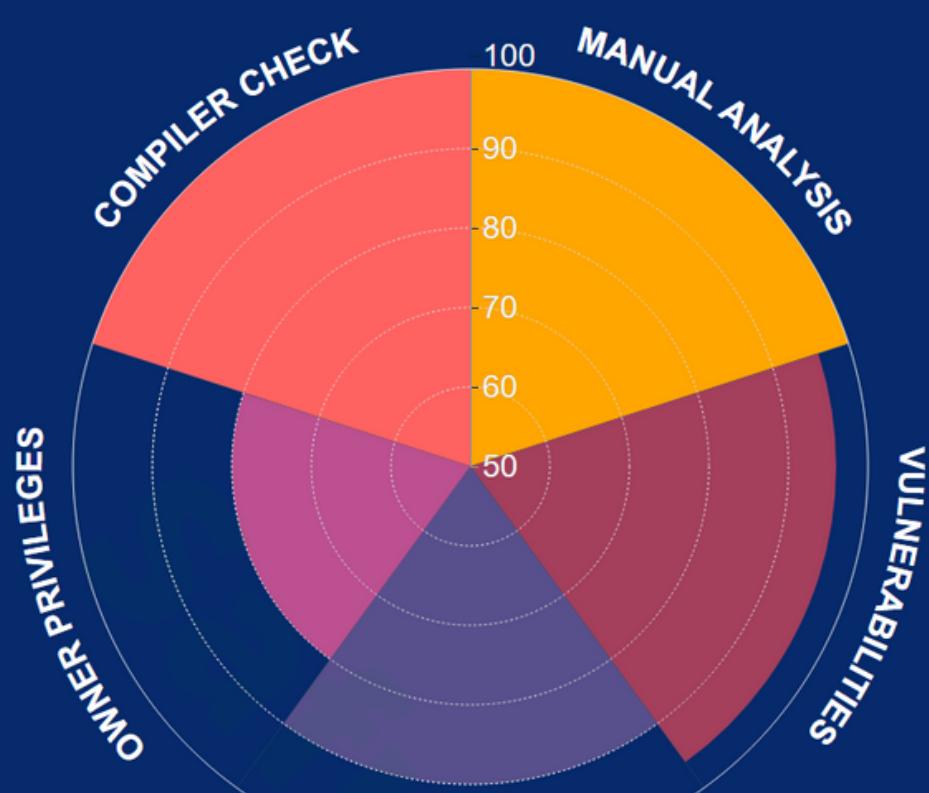
MINOR

- A floating pragma is set. 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.
- State variable visibility is not set for: _isFeeExempt, inSwap

INFORMATIONAL

- The owner can toggle liquidity add and rebase functionality at any time.
- The owner can swap all \$HARUNO in the contract for BNB and withdraw it from the contract to the "harunoTreasuryReceiver" address at any time.

SCORE



Manual Analysis



Vulnerabilities



Contract Readability



Owner Privileges

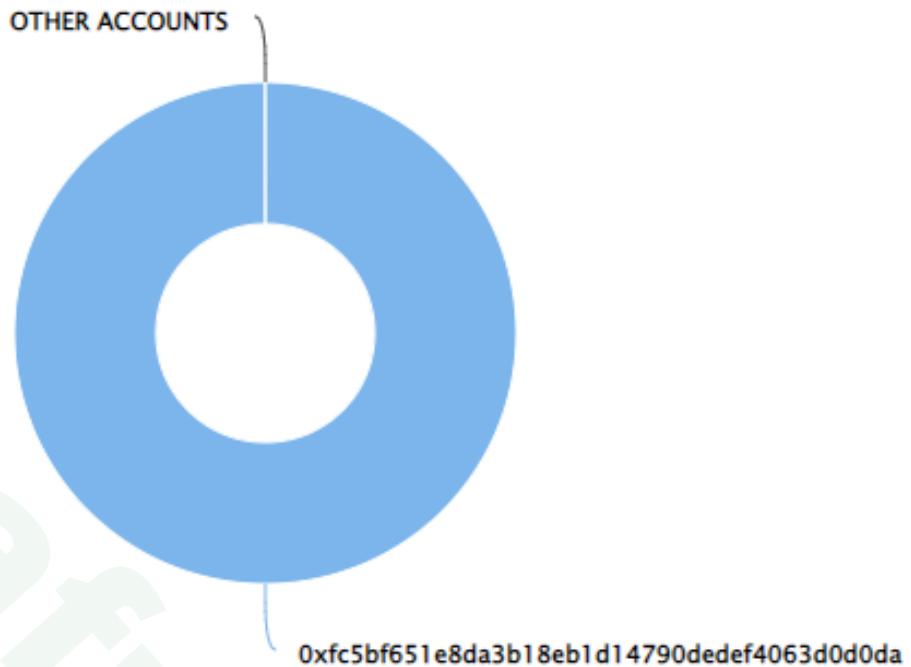


Compiler Check

Final Score: 93.2

SUMMARY

Top 10 holders



Conclusion

Project Haruno Inu does not contain any severe issues.

This is a rebase token: the owner can toggle liquidity adds and rebases from occurring at any time.

SafuAudit has tested the security based on manual and automated tests.

Please note that we don't offer any warranties for business model.



SafuAudit.com

