CRACKEN

Smart Contract Security Audit

Project: Hachiko Inu

Sep 01, 2022

Contract Address

0x66238A43794bB9076828c6839554C437e577c29e

Table of Contents

Disclaimer

The contents of this report reflect only the CRACKEN TECH audit team's understanding of the current progress and status of the security of the code audited, to verify the integrity of the code provided for the scope of this audit. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit are recommended after the issues covered are fixed. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

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.

The review does not address the compiler layer, any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from off-chain sources are not extended by this review either.

Audit Review

The source code of the Hachiko Inu was audited in order to acquire a clear impression of how the project was implemented. The Cracken Tech audit team conducted in-depth research, analysis, and scrutiny, resulting in a series of observations. A detailed list of each issue found, and vulnerabilities in the source code will be included in the audit report. The problems and potential solutions are given in this report, we will identify common sources for such problems and comments for improvement.

The auditing process will follow a routine as special considerations by Cracken:

- Review of the specifications, sources, and instructions provided to Cracken to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Cracken describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code is exercised when we run the test cases.
- Symbolic execution is analyzing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

Project Review

Token Summary

Parameter	Result
Token Name	HACHiKO
Token Symbol	HACHiKO
Token Decimal	9
Total Supply	100,000,000
Platform	BSC
Buy Tax Fee	4%
Sell Tax Fee	4%
Contract Creation Date	Sep 01, 2022
Liquidity Status	Not available
Liquidity Lockup Time	Not available
Compiler Version	v0.8.14+commit.80d49f37
Optimization	Yes with 200 runs
Contract Address	0x66238A43794bB9076828c6839554C437e577c29e
Deployer Address	0x14ded3e392b33dd962e651192a16b03e5f088986
Owner Address	0x14ded3e392b33dd962e651192a16b03e5f088986

Source Code

CRACKEN was commissioned by Hachiko Inu to perform an audit based on the following smart contract:

https://bscscan.com/address/0x66238A43794bB9076828c6839554C437e577c29e

Smart Contract Vulnerability Checks

Vulnerability	Auto-Scan	Manual-Scan	Result
Unencrypted Private Data On-Chain	Complete	Complete	Low / No Risk
Code With No Effects	Complete	Complete	Low / No Risk
Message call with hardcoded gas amount	Complete	Complete	Low / No Risk
Hash Collisions with Multiple Variable Length Arguments	Complete	Complete	Low / No Risk
Unexpected Ether balance	Complete	Complete	Low / No Risk
Presence of unused variables	Complete	Complete	Low / No Risk
Right-To-Left-Override control character (U+202E)	Complete	Complete	Low / No Risk
Typographical Error	Complete	Complete	Low / No Risk
DoS With Block Gas Limit	Complete	Complete	Low / No Risk
Arbitrary Jump with Function Type Variable	Complete	Complete	Low / No Risk
Insufficient Gas Grieving	Complete	Complete	Low / No Risk
Incorrect Inheritance Order	Complete	Complete	Low / No Risk
Write to Arbitrary Storage Location	Complete	Complete	Low / No Risk
Requirement Violation	Complete	Complete	Low / No Risk
Missing Protection against Signature Replay Attacks	Complete	Complete	Low / No Risk
Weak Sources of Randomness from Chain Attributes	Complete	Complete	Low / No Risk
Authorization through tx. origin	Complete	Complete	Low / No Risk
Delegate call to Untrusted Callee	Complete	Complete	Low / No Risk

Vulnerability	Auto-Scan	Manual-Scan	Result
Use of Deprecated Solidity Functions	Complete	Complete	Low / No Risk
Assert Violation	Complete	Complete	Low / No Risk
Reentrancy	Complete	Complete	Low / No Risk
Unprotected SELF-DESTRUCT Instruction	Complete	Complete	Low / No Risk
Unprotected Ether Withdrawal	Complete	Complete	Low / No Risk
Outdated Compiler Version	Complete	Complete	Low / No Risk
Integer Overflow and Underflow	Complete	Complete	Low / No Risk
Function Default Visibility	Complete	Complete	Low / No Risk

Manual Code Review

Classification of Issues

Severity	Description
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Informational	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
High-Risk	3
Medium-Risk	0
Low-Risk	0
Informational	0
Total	3

🔴 **High-Risk: functions make cause the rug or scam project. Must be fixed.**

Set max buy / sell tax fee

Description:

The owner can change the buy & sell fees up to 100%.

[HIGH RISK]

```
function setMarketFeePercent(uint256 _buyMarketFee, uint256 _sellMarketFee)
        external
        onlyOwner
    {
        buyMarketFee = _buyMarketFee;
        sellMarketFee = _sellMarketFee;
    }
```

Recommendation:

We recommend adding a requirement to limit the max fee amount.

🔴 **High-Risk: functions make cause the rug or scam project. Must be fixed.**

The trading function is enabled to be paused

Description:

The owner can pause the trading.

[HIGH RISK]

```
function setTradeEnabled(bool _enabled) public onlyOwner {

        tradeEnabled = _enabled;

        if (launchedAt == 0) launchedAt = block.number;

    }
```

Recommendation:

We recommend the owner disable this function.

🔴 **High-Risk: functions make cause the rug or scam project. Must be fixed.**

The blacklist function is enabled

Description:

The owner can add blacklist wallets.

[HIGH RISK]

```
function cpalaceAddressArray(address[] calldata account, bool value)
        external
        onlyOwner
    {
        for (uint256 i = 0; i < account.length; i++) {
            _isCpalaceed[account[i]] = value;
        }
    }
```

Recommendation:

We recommend that the owner should disable the blacklist function.

Privileged Functions

onlyOwner

Function Name	Parameters	Visibility
cpalaceAddressArray	address[] calldata account, bool value	External
decreaseAllowance	address spender, uint256 subtractedValue	Public
excludeMultipleAccountsFromFees	address[] calldata accounts,bool excluded	Public
increaseAllowance	address spender, uint256 addedValue	Public
multiTransfer4AirDrop	address[] calldata addresses, uint256 tokens	Public
renounceOwnership	None	Public
resetFeePercent	None	External
setMarketAddress	address market	Public
setMarketFeePercent	uint256 _buyMarketFee, uint256 _sellMarketFee	External
setNumTokensSellToMarket	uint256 num	Public
setTradeEnabled	bool _enabled	Public
transfer	address recipient, uint256 amount	External
transferFrom	address sender, address recipient, uint256 amount	Public
transferOwnership	address newOwner	Public
withdrawToken	address[] calldata tokenAddr, address recipient	Public

Contract Ownership

The contract ownership of Hachiko Inu is not currently being renounced. The ownership of the contract grants special powers to the protocol creators, making them the sole addresses that can call sensible ownable functions that may alter the state of the protocol.

The current owner is the address 0x14ded3e392b33dd962e651192a16b03e5f088986 which can be viewed: HERE

The owner wallet has the power to call the functions displayed on the privileged functions list above, if the owner wallet is compromised these privileges could be exploited.

We recommend the team renounce ownership at the right timing if possible, or gradually migrate to a time lock with governing functionalities in respect of transparency and safety considerations.

Liquidity Overview

Liquidity Information

Parameter	Result
Pair Address	0xb7671a1ab2aece36c849b4e31025368de88d3069
HACHiKO Reserves	0.00 HACHiKO
BNB Reserves	0.00 BNB
Liquidity Value	0.00 USD
Liquidity Ownership	The token does not have liquidity at the moment of the audit

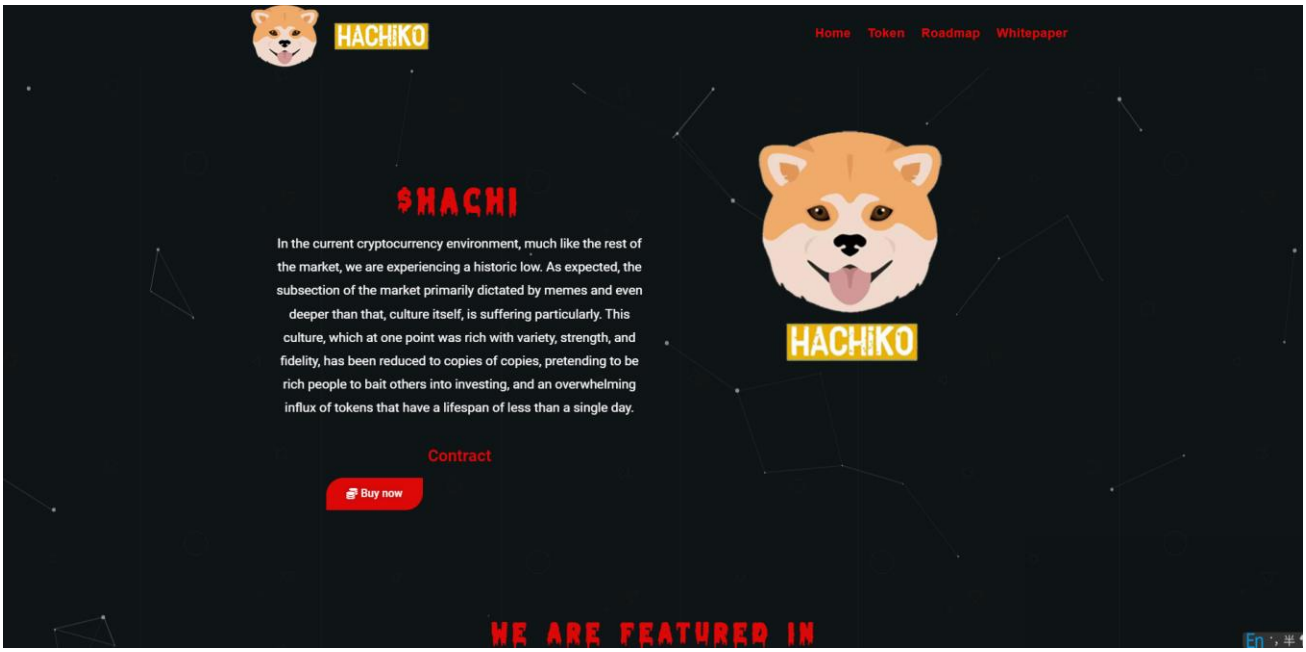
Tokenomics

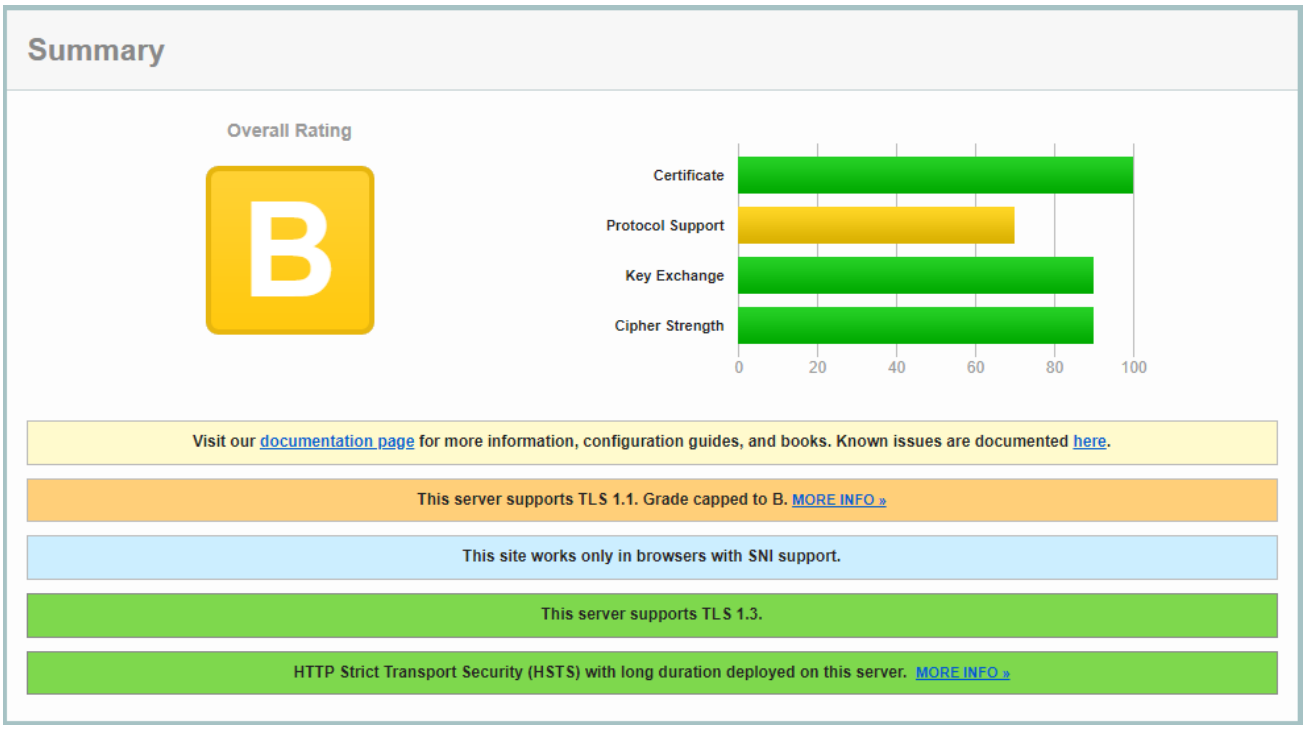
Rank	Address	Quantity (Token)	Percentage
1	0x1a1172747659a74405f291814c7faf7d988db312	100,000,000	100.0000%

Social Media Check

Social Media Type	Link	Result
Website	https://www.hachi.live/	Checked
Telegram	https://t.me/HACHiKOinuen/	Checked

Website Review





Certificate #1: RSA 2048 bits (SHA256withRSA)

Server Key and Certificate #1

Subject	hachi.live Fingerprint SHA256: dc6604751ec81d8a83b27fba1239b0f46640dab88995f4be977e3534f46e8baa Pin SHA256: 7QgiPqa+Sb7/AM1JRRgblVgExDkHeMGlZDo7KPi1rfg=
Common names	hachi.live
Alternative names	hachi.live www.hachi.live
Serial Number	036451121cd58dc4d7652d6c08f908a28be9
Valid from	Fri, 12 Aug 2022 16:59:02 UTC
Valid until	Thu, 10 Nov 2022 16:59:01 UTC (expires in 2 months and 9 days)
Key	RSA 2048 bits (e 65537)
Weak key (Debian)	No
Issuer	R3 AIA: http://r3.i.lencr.org/
Signature algorithm	SHA256withRSA
Extended Validation	No
Certificate Transparency	Yes (certificate)
OCSP Must Staple	No
Revocation information	OCSP OCSP: http://r3.o.lencr.org
Revocation status	Good (not revoked)
DNS CAA	No (more info)
Trusted	Yes Mozilla Apple Android Java Windows

- Mobile Friendly
- Contains no code errors
- SSL is secured
- No spelling errors

Audit Conclusion

- **The owner can pause trading [High-Risk]**
- The owner cannot mint new tokens
- The owner cannot set the max transaction amount
- **The owner can change the buy/sell fee up to 100% [High-Risk]**
- **The owner can blacklist wallets [High-Risk]**

  (All functions cannot be used if the ownership is renounced)