CRACKEN

Smart Contract Security Audit

Project: ADAM

Aug 11, 2022

ADAM

Contract Address

0x5f026f015773C3250EdD3Cf9EcBCC0e2Ff5e712E

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 ADAM 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	ADAM
Token Symbol	ADAM
Token Decimal	18
Total Supply	45,000,000,000
Platform	BSC
Buy Tax Fee	8%
Sell Tax Fee	8%
Contract Creation Date	Aug 09, 2022
Liquidity Status	Not Available
Liquidity Lockup Time	Not Available
Compiler Version	v0.6.12+commit.27d51765
Optimization	Yes with 200 runs
Contract Address	0x5f026f015773C3250EdD3Cf9EcBCC0e2Ff5e712E
Deployer Address	0x1dCB1F9F6C3fA4C694fdC13327A9623bf1297A07
Owner Address	0x1dCB1F9F6C3fA4C694fdC13327A9623bf1297A07

Source Code

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

https://bscscan.com/address/ 0x5f026f015773C3250EdD3Cf9EcBCC0e2Ff5e712E

CRACKEN

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	0
🟠 Medium-Risk	0
🟡 Low-Risk	0
🔵 Informational	3
Total	3

🔵 **Informational: Implementation of certain corrective actions or accepting the risk.**

Set max buy / sell tax fee

Description:

The owner can set fees up to 30%

```
function setFee(
        uint256 _ADARewardsFee,
        uint256 _ADARShare,
        uint256 _ADABLShare
    ) public onlyOwner {
        require(_ADARewardsFee < 10, "ADAM: Fee exceed limit");
        require(_ADARShare < 10, "ADAM: Fee exceed limit");
        require(_ADABLShare < 10, "ADAM: Fee exceed limit");
        require(_ADARewardsFee >= 0, "ADAM: Fee must bigger than zero");
        require(_ADARShare >= 0, "ADAM: Fee must bigger than zero");
        require(_ADABLShare >= 0, "ADAM: Fee must bigger than zero");
        ADARewardsFee = _ADARewardsFee;
        ADARShare = _ADARShare;
        ADABLShare = _ADABLShare;
    }
```

🔵 **Informational: Implementation of certain corrective actions or accepting the risk.**

Set AntiBot

Description:

The owner can set anti bot function.

```
// anti bot
        address public _PresaleAddress =
0x000000000000000000000000000000000000dEaD;
        bool public liquidityLaunched = false; // to track if launchLiquidity function has been
called
        bool public isFirstLaunch = true; // to track if launchLiquidity function has been called
        uint256 public lastSnipeTaxBlock; // set to blocks after liq added
        uint8 public snipeBlocks = 0;
```

🔵 **Informational: Implementation of certain corrective actions or accepting the risk.**

Change the gas fee

Description:

The owner can change the gas fee.

```
function updateGasForProcessing(uint256 newValue) public onlyOwner {
            require(newValue >= 200000 && newValue <= 500000, "ADA: gasForProcessing
must be between 200,000 and 500,000");
            require(newValue != gasForProcessing, "ADA: Cannot update gasForProcessing
to same value");
            emit GasForProcessingUpdated(newValue, gasForProcessing);
            gasForProcessing = newValue;
    }
```

Privileged Functions

onlyOwner

Function Name	Parameters	Visibility
decreaseAllowance	Address spender, unit256 subtractedValue	External
excludeFromDividends	Address account	Public
excludeFromFees	Address account, bool excluded	External
excludedMultipleAccountsFromFees	address[] calldata accounts,bool excluded	Public
renounceOwnership	None	Public
setDevWallet	address payable wallet	External
setFee	uint256_ADARewardsFee,uint256 _ADARShare,uint256 _ADABLShare	Public
setFoundWalletAddress	address payable wallet	External

Contract Ownership

The contract ownership of ADAM is not currently 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 0x1dCB1F9F6C3fA4C694fdC13327A9623bf1297A07 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	0xc97210e35bb15ce477ec24914ade7f8a247a2437
ADAM Reserves	0.00 ADAM
BNB Reserves	0.00 BNB
Liquidity Value	$0 USD
Liquidity Ownership	The token does not have liquidity at the moment of the audit

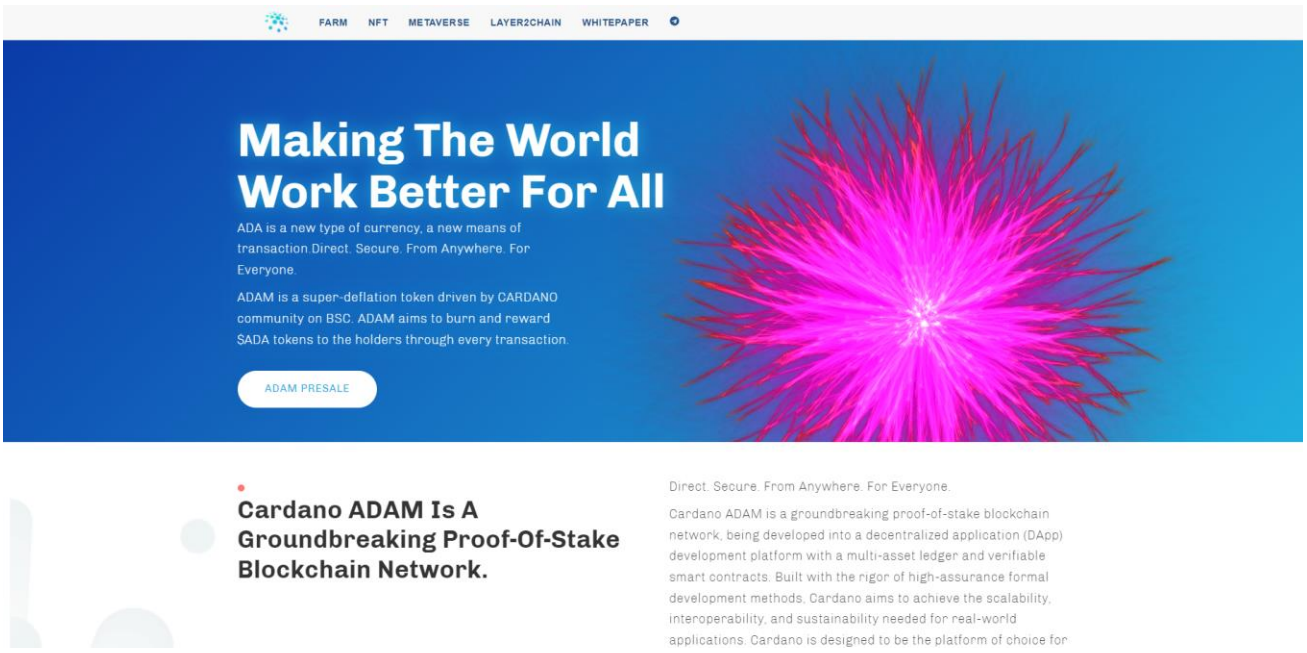
Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0x000000000000000000000000000000000000dead	22,050,000,000	49.0000%
2	0x1dcb1f9f6c3fa4c694fdc13327a9623bf1297a07	18,900,000,000	42.0000%
3	0xcfac2ab8aa2e6b1e70bfcb7c170b38e36ca3891c	900,000,000	2.0000%
4	0x1f92b7e29fab5a8d4411de78825c88690d0e3f1d	900,000,000	2.0000%
5	0xdbdb5806630489d7841dd8cb7fa38c9d3e5e2a11	900,000,000	2.0000%

Social Media Check

Social Media Type	Link	Result
Website	https://ADAMtoken.tech	Checked
Twitter	https://twitter.com/ ADAMBinance/	Checked
Telegram	https://t.me/ ADAMTokenGlobal/	Checked

Website Review



Making The World Work Better For All

ADA is a new type of currency, a new means of transaction.Direct. Secure. From Anywhere. For Everyone.

ADAM is a super-deflation token driven by CARDANO community on BSC. ADAM aims to burn and reward $ADA tokens to the holders through every transaction.

Cardano ADAM Is A Groundbreaking Proof-Of-Stake Blockchain Network.

Direct. Secure. From Anywhere. For Everyone.

Cardano ADAM is a groundbreaking proof-of-stake blockchain network, being developed into a decentralized application (DApp) development platform with a multi-asset ledger and verifiable smart contracts. Built with the rigor of high-assurance formal development methods, Cardano aims to achieve the scalability, interoperability, and sustainability needed for real-world applications. Cardano is designed to be the platform of choice for

- Mobile Friendly
- Contains no code errors
- SSL Secured
- No spelling errors

Audit Conclusion

- The owner cannot pause trading
- The owner cannot mint new tokens
- The owner cannot blacklist users
- The owner cannot set the max transaction amount.
- The owner can change the buy/sell fee up to 30%.
- The owner can change the gas fee
- The contract has antibot function

AUDIT IS PASSED