ASTROBIATECH
BLOCKCHAIN SECURITY

MADE IN INDIA
BLOCKCHAIN SECURITY

SECURITY ASSESSMENT REPORT

PREPARED FOR
KERMIT

@astrobiatech

JULY
2023

astrobiatech.in

TABLE OF CONTENTS

SCOPE OF AUDIT

The scope of this audit was to analyze and document the KERMIT smart contract codebase for quality, security, and correctness.

CHECKED VULNERABILITIES

We have scanned the smart contract for commonly known and morespecific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

TECHNIQUES & METHODS

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all thesmart contracts.

Static Analysis

Static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step a series of automated tools are used to testsecurity of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis.Contracts were completely manually analyzed, their logic was checkedand compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

ISSUE CATEGORIES

Every issue in this report has been assigned with a severity level. There are four levels of severity and each of them has been explained below.

➢ HIGH SEVERITY ISSUES

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality and we recommend these issues to be fixed before movingto a live environment.

➢ MEDIUM SEVERITY ISSUES

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems and they should still be fixed.

➢ LOW SEVERITY ISSUES

Low level severity issues can cause minor impact and or are just warningsthat can remain unfixed for now. It would be better to fix these issues atsome point in the future.

➢ INFORMATIONAL

These are severity four issues which indicate an improvement request, ageneral question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

ISSUES TABLE

TYPE	HIGH	MEDIUM	LOW	INFORMATIONAL
OPEN	1	-	5	-
ACKNOWLWDGENT	-	-	-	-
CLOSED	-	-	-	-

INTRODUCTION

On 13-07-2023 – Astrobiatech Blockchain Security Team performed asecurity audit for KERMIT smart contracts.

CONTRACT NAME	KERMIT
CONTRACT ADDRESS	0x84d2CDC7f7c6dEfCbd2c00B606485A939c081275
BLOCKCHAIN	Binance
TOTAL SUPPLY	1000000000000
SYMBOL	kermit
DECIMALS	18

OVERVIEW

CONTRACT ADDRESS
0x84d2CDC7f7c6dEfCbd2c00B606485A939c081275

TOKEN TRACKER
Kermit (kermit)

CONTRACT CREATOR
0x1a4Cb975B959AD4Ab87EB80384Cf8f9cCC31Bd1c

OWNER ADDRESS
0x1a4Cb975B959AD4Ab87EB80384Cf8f9cCC31Bd1c

SOURCE CODE
Contract Source Code Verified at Bscscan

CONTRACT NAME
Kermit

OTHER SETTINGS
default evmVersion

COMPILER VERSION
v0.8.17+commit.8df45f5f

OPTIMIZATION ENABLED
Yes with 2000 runs

Code is truncated to fit the constraints of this document.
The code in its entirety can be viewed here.

MANUAL ANALYSIS FINDINGS

HIGH

1. Trade must be enabled by the owner

Description:-

Owner must enable trading to enable public to trade their tokens, otherwise no one would be able to buy /sell their tokens except whitelisted wallets.

Recommendation:-

To mitigate this issue you should enable trading before presale or transfer ownership to safu dev for initial days after presale.

LOW

1. Owner can exclude accounts from fees

Description:-

Excludes/Includes an address from the collectionof fees

Recommendation:-

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limitthe number of authorized users who can call these functions. The contract owner account is well secured and only accessible by authorized parties.

2. Owner can change buy fees up to 10% sell fees up to 15% at max

Description:-

Functions that allows the owner of the contract to update the buy/sell fees of the contract. These functions assumes that the input parameters are valid and do not exceed the maximum limit of 10% for buy fees and maximum limit of 15% for sell fees.

Recommendation:-

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limit the number of authorized users who can call these functions.

3. Owner can exclude/include account from rewards

Description:-

Function that allows the owner of the contract to exclude / include an address from receiving dividends

Recommendation:-

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limit the number of authorized users who can call these functions. The contract owner account is well secured and only accessible by authorized parties.

4. Owner can withdraw stuck BNB and stuck tokens

Description:-

claimStuckTokens allow the contract owner to withdraw locked or stuck ETH and ERC20 tokens from the contract. The functions are properly restricted to only be executed by the contract owner.

Recommendation:-

While the functions are currently restricted to only be called by the contract owner, it is recommended to consider implementing a more robust access control mechanism.

5. Owner can change swap setting

Description:-

Function allows the contract owner to enable ordisable the automatic swapping.

Recommendation:-

It is recommended to ensure that the contract owner account is well secured and onlyaccessible by authorized parties.

AUTOMATED ANALYSIS

```
Reentrancy in Kermit._transfer(address,address,uint256) (token.sol#667-708):
        External calls:
        - swapAndLiquify(liquidityTokens) (token.sol#695)
                -
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestam
p) (token.sol#720-725)
                - uniswapV2Router.addLiquidityETH{value:
newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (token.sol#729-736)
        - swapAndSendMarketing(marketingTokens) (token.sol#700)
                - (success) = recipient.call{value: amount}() (token.sol#74)
                -
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.t
imestamp) (token.sol#748-753)
                - address(marketingWallet).sendValue(newBalance) (token.sol#757)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (token.sol#695)
                - uniswapV2Router.addLiquidityETH{value:
newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp) (token.sol#729-736)
        - swapAndSendMarketing(marketingTokens) (token.sol#700)
                - (success) = recipient.call{value: amount}() (token.sol#74)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount) (token.sol#707)
                - _rOwned[address(this)] = _rOwned[address(this)] + rLiquidity (token.sol#598)
                - _rOwned[address(this)] = _rOwned[address(this)] + rMarketing (token.sol#608)
                - _rOwned[sender] = _rOwned[sender] - rAmount (token.sol#804)
                - _rOwned[sender] = _rOwned[sender] - rAmount (token.sol#814)
                - _rOwned[recipient] = _rOwned[recipient] + rTransferAmount (token.sol#805)
                - _rOwned[sender] = _rOwned[sender] - rAmount (token.sol#826)
                - _rOwned[sender] = _rOwned[sender] - rAmount (token.sol#837)
                - _rOwned[recipient] = _rOwned[recipient] + rTransferAmount (token.sol#816)
                - _rOwned[recipient] = _rOwned[recipient] + rTransferAmount (token.sol#827)
                - _rOwned[recipient] = _rOwned[recipient] + rTransferAmount (token.sol#839)
        Kermit._rOwned (token.sol#318) can be used in cross function reentrancies:
        - Kermit._getCurrentSupply() (token.sol#582-592)
        - Kermit._takeLiquidity(uint256) (token.sol#594-602)
        - Kermit._takeMarketing(uint256) (token.sol#604-612)
        - Kermit._transferBothExcluded(address,address,uint256) (token.sol#834-844)
        - Kermit._transferFromExcluded(address,address,uint256) (token.sol#823-832)
        - Kermit._transferStandard(address,address,uint256) (token.sol#802-810)
        - Kermit._transferToExcluded(address,address,uint256) (token.sol#812-821)
        - Kermit.balanceOf(address) (token.sol#445-448)
        - Kermit.constructor() (token.sol#375-427)
        - Kermit.deliver(uint256) (token.sol#488-495)
        - Kermit.excludeFromReward(address) (token.sol#514-521)
        - _tokenTransfer(from,to,amount) (token.sol#707)
                - _rTotal = _rTotal - rFee (token.sol#550)
        Kermit._rTotal (token.sol#332) can be used in cross function reentrancies:
        - Kermit._getCurrentSupply() (token.sol#582-592)
        - Kermit._reflectFee(uint256,uint256) (token.sol#549-552)
        - Kermit.constructor() (token.sol#375-427)
        - Kermit.deliver(uint256) (token.sol#488-495)
        - Kermit.tokenFromReflection(uint256) (token.sol#508-512)
        - _tokenTransfer(from,to,amount) (token.sol#707)
                - _tOwned[address(this)] = _tOwned[address(this)] + tMarketing (token.sol#610)
                - _tOwned[address(this)] = _tOwned[address(this)] + tLiquidity (token.sol#600)
```

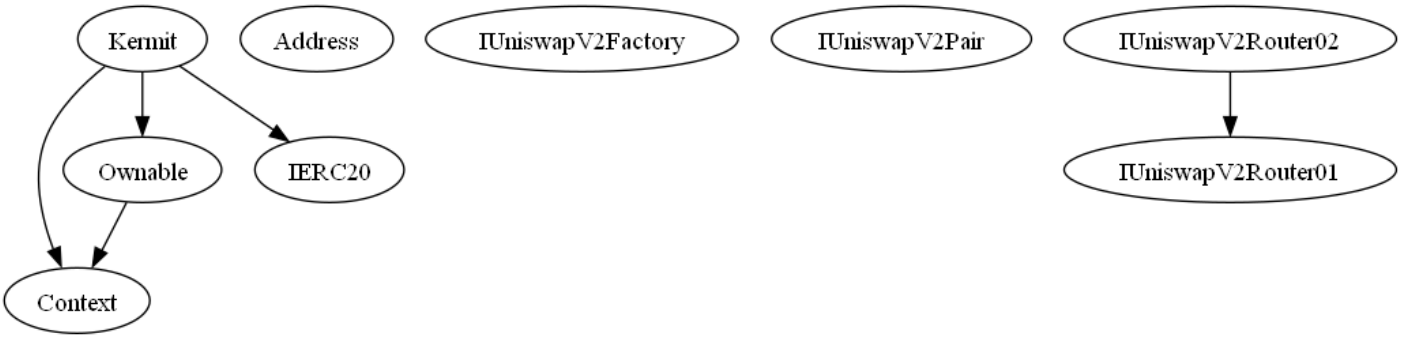
```
                - _tOwned[sender] = _tOwned[sender] - tAmount (token.sol#825)
                - _tOwned[sender] = _tOwned[sender] - tAmount (token.sol#836)
                - _tOwned[recipient] = _tOwned[recipient] + tTransferAmount (token.sol#815)
                - _tOwned[recipient] = _tOwned[recipient] + tTransferAmount (token.sol#838)
        Kermit._tOwned (token.sol#319) can be used in cross function reentrancies:
        - Kermit._getCurrentSupply() (token.sol#582-592)
        - Kermit._takeLiquidity(uint256) (token.sol#594-602)
        - Kermit._takeMarketing(uint256) (token.sol#604-612)
        - Kermit._transferBothExcluded(address,address,uint256) (token.sol#834-844)
        - Kermit._transferFromExcluded(address,address,uint256) (token.sol#823-832)
        - Kermit._transferToExcluded(address,address,uint256) (token.sol#812-821)
        - Kermit.balanceOf(address) (token.sol#445-448)
        - Kermit.constructor() (token.sol#375-427)
        - Kermit.excludeFromReward(address) (token.sol#514-521)
        - Kermit.includeInReward(address) (token.sol#523-534)
        - inSwapAndLiquify = false (token.sol#703)
        Kermit.inSwapAndLiquify (token.sol#360) can be used in cross function reentrancies:
        - Kermit._transfer(address,address,uint256) (token.sol#667-708)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
Kermit.claimStuckTokens(address) (token.sol#538-547) ignores return value by
ERC20token.transfer(msg.sender,balance) (token.sol#546)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer

INFO:Detectors:
Kermit.constructor().router (token.sol#377) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Kermit.claimStuckTokens(address) (token.sol#538-547) ignores return value by
address(msg.sender).sendValue(address(this).balance) (token.sol#541)
Kermit.swapAndLiquify(uint256) (token.sol#710-739) ignores return value by
uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,DEAD,block.timestamp)
(token.sol#729-736)
Kermit.swapAndSendMarketing(uint256) (token.sol#741-760) ignores return value by
address(marketingWallet).sendValue(newBalance) (token.sol#757)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
```

FUNCTIONAL ANALYSIS

Kermit	Implementation	Context, IERC20, Ownable		
└	<Constructor>	Public !	🔴	NO !
└	name	Public !		NO !
└	symbol	Public !		NO !
└	decimals	Public !		NO !
└	totalSupply	Public !		NO !
└	balanceOf	Public !		NO !
└	transfer	Public !	🔴	NO !
└	allowance	Public !		NO !
└	approve	Public !	🔴	NO !
└	transferFrom	Public !	🔴	NO !
└	increaseAllowance	Public !	🔴	NO !
└	decreaseAllowance	Public !	🔴	NO !
└	isExcludedFromReward	Public !		NO !
└	totalReflectionDistributed	Public !		NO !
└	deliver	Public !	🔴	NO !
└	reflectionFromToken	Public !		NO !
└	tokenFromReflection	Public !		NO !
└	excludeFromReward	Public !	🔴	onlyOwner
└	includeInReward	External !	🔴	onlyOwner
└	<Receive Ether>	External !	💸	NO !
└	claimStuckTokens	External !	🔴	onlyOwner
└	_reflectFee	Private 🔐	🔴	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🔴	
└	_takeMarketing	Private 🔐	🔴	
└	calculateTaxFee	Private 🔐		
└	calculateLiquidityFee	Private 🔐		
└	calculateMarketingFee	Private 🔐		
└	removeAllFee	Private 🔐	🔴	
└	setBuyFee	Private 🔐	🔴	
└	setSellFee	Private 🔐	🔴	
└	isExcludedFromFee	Public !		NO !
└	_approve	Private 🔐	🔴	
└	enableTrading	External !	🔴	onlyOwner
└	_transfer	Private 🔐	🔴	
└	swapAndLiquify	Private 🔐	🔴	
└	swapAndSendMarketing	Private 🔐	🔴	
└	setSwapTokensAtAmount	External !	🔴	onlyOwner
└	setSwapEnabled	External !	🔴	onlyOwner
└	_tokenTransfer	Private 🔐	🔴	
└	_transferStandard	Private 🔐	🔴	
└	_transferToExcluded	Private 🔐	🔴	
└	_transferFromExcluded	Private 🔐	🔴	
└	_transferBothExcluded	Private 🔐	🔴	
└	excludeFromFees	External !	🔴	onlyOwner
└	changeMarketingWallet	External !	🔴	onlyOwner
└	setBuyFeePercentages	External !	🔴	onlyOwner
└	setSellFeePercentages	External !	🔴	onlyOwner
└	enableWalletToWalletTransferWithoutFee	External !	🔴	onlyOwner

INHERITANCE TREE



FEES

BUY FEES	3%
Marketing	3%
Reflection	0%

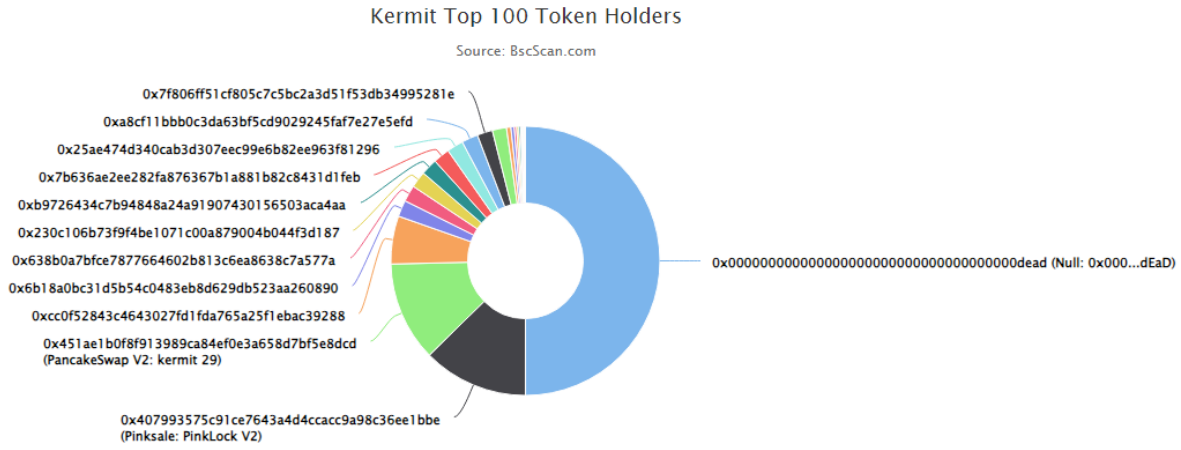
SELL FEES	6%
Marketing	3%
Reflection	3%

HOLDERS

Rank	Address	Quantity	Percentage	Analytics
1	Null: 0x000...dEaD	500,000,000,000	50.0000%	
2	Pinksale: PinkLock V2	126,500,000,000	12.6500%	
3	PancakeSwap V2: kermit 29	119,795,000,000.000000005173588444	11.9795%	
4	0xcc0f52843c4643027fd1fda765a25f1ebac39288	57,623,944,161.41895587143674535	5.7624%	
5	0x6b18a0bc31d5b54c0483eb8d629db523aa260890	19,726,442,985.884983857296463577	1.9726%	
6	0x638b0a7bfce7877664602b813c6ea8638c7a577a	19,726,442,985.884983857296463577	1.9726%	
7	0x230c106b73f9f4be1071c00a879004b044f3d187	19,726,442,985.884983857296463577	1.9726%	
8	0xb9726434c7b94848a24a91907430156503aca4aa	19,726,442,985.884983857296463577	1.9726%	
9	0x7b636ae2ee282fa876367b1a881b82c8431d1feb	19,726,442,985.884983857296463577	1.9726%	
10	0x25ae474d340cab3d307eec99e6b82ee963f81296	19,726,442,985.884983857296463577	1.9726%	
11	0xa8cf11bbb0c3da63bf5cd9029245faf7e27e5efd	19,726,442,985.884983857296463577	1.9726%	
12	0x7f806ff51cf805c7c5bc2a3d51f53db34995281e	18,411,346,786.825984933476699339	1.8411%	
13	0x0042ce294de9d388bf0a96c26ecf7cbc622312d9	17,096,250,587.7669860096569351	1.7096%	
14	0x42503a0df7c018eec332e6bddbb98ab8aa4bb086	5,260,384,796.235999695927956954	0.5260%	
15	0xd1443cdb3aa6f0710df557835010ae7b4a044d27	3,945,288,597.176996771459292715	0.3945%	
16	0xb8f9170a577842bfaae74bc63dde16141820983a	2,827,669,386.861152268075311368	0.2828%	
17	0x4a71ffb14d0f0c2bd8359325aea88d20ecf8ff53	2,630,192,398.11799784763952847	0.2630%	
18	0xea8f7d015a141aaf715abe36ed4d9d5d27bbd8fa	2,130,455,642.475578256588018066	0.2130%	
19	0x4fcae1998df7b99a967f5ed965cf1457bce755c4	1,315,096,199.058998923819764238	0.1315%	
20	0x23a7b698eb9f793aa50d3b4e535e1d825d140cbe	789,057,719.435399354291858543	0.0789%	
21	0xc313398f53ce18868b94b2eee645ea5e14fcb13b	552,340,403.60477954800430098	0.0552%	
22	0x33fd3a83b40e51c398eecb0fa93278699a58400e	526,038,479.623599569527905695	0.0526%	
23	0x404d186d0f8b11eaebbf48c5255cc458f9701764	526,038,479.623599569527905695	0.0526%	
24	0x412f555ff52b8f765522243c4f22bc94013ecf59	473,434,631.661239612575115125	0.0473%	
25	0x7649ff128b6eab62a279861734c42b74bf6348f3	447,132,707.680059634098719841	0.0447%	
26	0xbc7d36c174e9732caf05554430bd0bc39084e2cf	447,132,707.680059634098719841	0.0447%	
27	0xd79b0901b5f29f6592853efe182caade9402b033	263,019,239.811799784763952847	0.0263%	
28	0x1067895ed7ff9df6d9e41000256a887d334cf712	263,019,239.811799784763952847	0.0263%	
29	0xf51f02b77cb7c8c4611f13b337d3cc09fd280646	92,056,733.934129924667383496	0.0092%	

The top 100 holders collectively own 100.00% (1,000,000,000,000.00 Tokens) of Kermit

Token Total Supply: 1,000,000,000,000.00 Token | Total Token Holders: 29



(A total of 1,000,000,000,000.00 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000.00 token)

SUMMARY

In this report, we have considered the security of the KERMITplatform. We performed our audit according to the procedure described above. 1 high , 0 medium, 5 low, and 0 informational severity were discovered during the audit.

DISCLAIMER

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constituteinvestment advice. No one shall have any right to rely on the report or its contents, and Astrobiatech Blockchain Security and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Astrobiatech BlockchainSecurity) owe no duty of care towards you or any other person, nor does Astrobiatech Blockchain Security make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Astrobiatech Blockchain Security hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effectin relation to the report. Except and only to the extent that it is prohibited by law, Astrobiatech Blockchain Security hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Astrobiatech Blockchain Security, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic lossor damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. 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.

Proofed by Astrobiatech

ASTROBIATECH
BLOCKCHAIN SECURITY

https://astrobiatech.in

@astrobiatech