HackSafe

Smart Contract Security Audit Report

BAD APES

December 2022

Security Status

★★★★☆

www.hacksafe.io

Audit Details

Audited project
BAD APES

Deployer address
0x153b202f6c6e570f13c27371cda6ae2c8768dca6

Client contacts
BAD APES Team

Blockchain
Binance smart chain

Website
Not provided

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

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 constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (HackSafe) owe no duty of care towards you or any other person, nor does HackSafe 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 HackSafe 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 effect in relation to the report. Except and only to the extent that it is prohibited by law, HackSafe hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HackSafe, 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 loss or 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.

Procedure

Step 1 - In-Depth Manual Review

Manual line-by-line code reviews to ensure the logic behind each function is sound and safe from various attack vectors. This is the most important and lengthy portion of the audit process (as automated tools often cannot find the nuances that lead to exploits such as flash loan attacks).

Step 2 - Automated Testing

Simulation of a variety of interactions with your Smart Contract on a test blockchain leveraging a combination of automated test tools and manual testing to determine if any security vulnerabilities exist.

Step 3 – Leadership Review

The engineers assigned to the audit will schedule meetings with our leadership team to review the contracts, any comments or findings, and ask questions to further apply adversarial thinking to discuss less common attack vectors.

Step 4 - Resolution of Issues

Consulting with the team to provide our recommendations to ensure the code's security and optimize its gas efficiency, if possible. We assist project team's in resolving any outstanding issues or implementing our recommendations.

Step 5 - Published Audit Report

Boiling down results and findings into an easy-to-read report tailored to the project. Our audit reports highlight resolved issues and any risks that exist to the project or its users, along with any remaining suggested remediation measures. Diagrams are included at the end of each report to help users understand the interactions which occur within the project.

Background

HackSafe was commissioned by to BAD APES perform an audit of smart contracts:

- https://bscscan.com/token/0xc4f5424ef52499fa496a07f3fe9daab88553d4c3#code

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contract Details

Token contract details for 06.12.2022

Token Type	: Utility
Contract name	: CoinToken
Contract address	: 0xC4F5424eF52499fa496a07f3fE9DaAb88553D4C3
Total supply	: 99,288,209,496,791,300.300249
Token ticker	: BAYC
Decimals	: 18
Token Holders	: 411
Transactions count	: 6,814
Compiler version	: v0.8.7+commit.e28d00a7
Contract deployer address	: 0x153b202f6c6e570f13c27371cda6ae2c8768dca6
Owner address	: 0x03e37d9cfa9a911bdce76adf1c3d92fd86636659

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **“Secure”**. This token contract does contain owner control, which do not make it fully decentralized as owner does have control over smart contract.

Insecure	Poor secured	Secure	Well-secured

You are here 

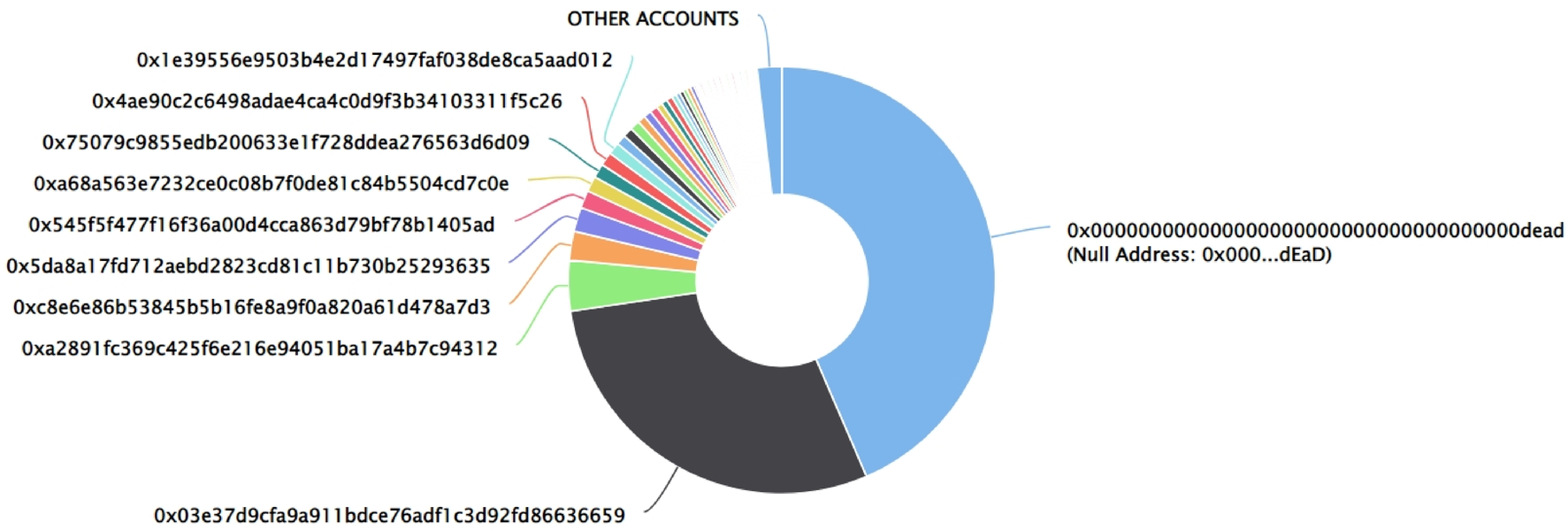
We used various tools like Slither, Mythril and Remix IDE. At the same time this finding is based on critical analysis of the manual audit. All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the issues checking status.

We found 0 critical, 0 high, 1 medium and 0 low.

BAD APES Distribution

The top 100 holders collectively own 98.15% (97,452,168,336,832,700.00 Tokens) of BAD APES

Token Total Supply: 99,288,209,496,791,300.30 Token | Total Token Holders: 41



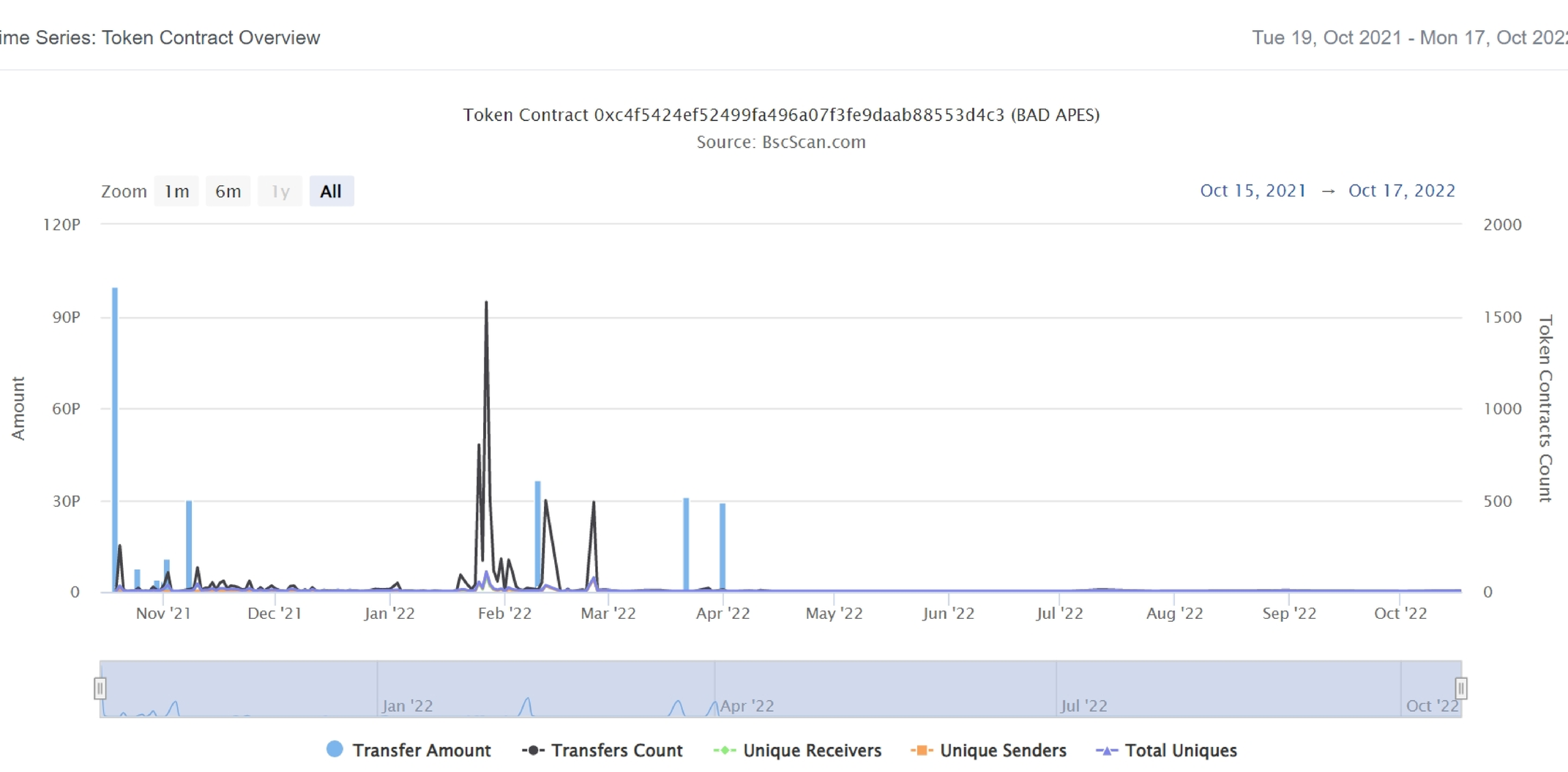
BAD APES Top 20 Token Holders

(A total of 97,452,168,336,832,700.00 tokens held by the top 100 accounts from the total supply of 99,288,209,496,791,300.30 token)

Rank	Address	Quantity (Token)	Percentage
1	Null Address: 0x000...dEaD	43,215,053,047,305,800.2699001491286767	43.5249%
2	0x03e37d9cfa9a911bdce76adf1c3d92fd86636659	28,958,323,759,714,900.722870352326173849	29.1659%
3	0xa2891fc369c425f6e216e94051ba17a4b7c94312	3,796,833,150,499,390.553224633680206702	3.8241%
4	0xc8e6e86b53845b5b16fe8a9f0a820a61d478a7d3	2,164,519,406,956,020.09990292873887644	2.1800%
5	0x5da8a17fd712aebd2823cd81c11b730b25293635	1,827,938,544,346,060.992332783942895834	1.8410%
6	0x545f5f477f16f36a00d4cca863d79bf78b1405ad	1,321,391,675,872,970	1.3309%
7	0xa68a563e7232ce0c08b7f0de81c84b5504cd7c0e	1,174,570,378,553,750	1.1830%
8	0x75079c9855edb200633e1f728ddea276563d6d09	1,031,130,100,548,900.028853409093166212	1.0385%
9	0x4ae90c2c6498adae4ca4c0d9f3b34103311f5c26	1,027,749,081,234,530	1.0351%
10	0x1e39556e9503b4e2d17497faf038de8ca5aad012	920,014,394,407,292.218598166276610089	0.9266%
11	0xb884ffc2ce978aab2d4306cd307410e6cf082aa3	758,862,778,134,292.104469422092336806	0.7643%
12	0xc713ff7e5cf9d26963e0dc076ca12e0618f5efd2	734,295,407,175,400.181371360391213066	0.7396%
13	0x1be8fd2fe2b6516880d303a5bd44dbb673a56dff	734,106,486,596,095	0.7394%
14	0x7921ec61854c95e8ff812cbe250ea1cda97fc0d8	587,286,697,149,594.182747967299461307	0.5915%
15	0x1cf47b263f96d72b2c6359bec9cf0044969bea05	587,285,189,276,877	0.5915%
16	0x768af20bc05440be1507519ea82e69fff97fa19d	587,285,189,276,876	0.5915%
17	0x873fea10d7f80483fa06efd992305596e9409847	440,815,661,552,938.915325449153200175	0.4440%
18	0x12559e49cc05e11f8b68b817f7a1e8fd153368ea	440,463,891,957,657	0.4436%
19	0xd1e73f6e79d143e5049010049eec60728106cf7b	440,463,891,957,657	0.4436%
20	0x9e7eb4972c6bd39dcf793f08141e1e3e2d8be74e	354,804,807,238,167.208261255104888729	0.3573%

BAD APES Distribution

BAD APES Overview



Contract functions details

+Context
    -[Int] _msgSender
    -[Int] _msgData

+[Int] IBEP20
    -[Ext] totalSupply
    -[Ext] balanceOf
    -[Ext] transfer
    -[Ext] allowance
    -[Ext] approve
    -[Ext] transferFrom

+[Lib] SafeMath
    -[Int] add
    -[Int] sub
    -[Int] sub
    -[Int] mul
    -[Int] div
    -[Int] mod
    -[Int] mod

+[Lib] Address
    -[Int] isContract
    -[Int] sendValue
    -[Int] functionCall
    -[Int] functionCall
    -[Int] functionCallWithValue
    -[Int] functionCallWithValue
    -[Pvt] _functionCallWithValue

+Ownable (Context)
    -[Pub] owner
    -[Pub] renounceOwnership #
      -modifiers: onlyOwner
    -[Pub] transferOwnership #
      -modifiers: onlyOwner

+CoinToken (Context, IBEP20, Ownable)
    -<constructor>
    -[Pub] name

Contract functions details

-[Pub] symbol
-[Pub] decimals
-[Pub] totalSupply
-[Pub] balanceOf
-[Pub] transfer #
-[Pub] allowance
-[Pub] approve #
-[Pub] transferFrom #
-[Pub] increaseAllowance #
-[Pub] decreaseAllowance #
-[Pub] isExcluded
-[Pub] totalFees
-[Pub] totalBurn
-[Pub] totalCharity
-[Pub] deliver
-[Pub] reflectionFromToken #
-[Pub] tokenFromReflection #
-[Ext] excludeAccount #
  -modifiers: onlyOwner
-[Ext] includeAccount
  -modifiers: onlyOwner
-[Ext] setAsCharityAccount #
  -modifiers: onlyOwner
-[Pub] updateFee #
  -modifiers: onlyOwner
-[Pvt] _approve #
-[Pvt] _transfer #
-[Pvt] _transferStandard #
-[Pvt] _standardTransferContent #
-[Pvt] _transferToExcluded #
-[Pvt] _excludedFromTransferContent #
-[Pvt] _transferFromExcluded #
-[Pvt] _excludedToTransferContent #
-[Pvt] _transferBothExcluded #
-[Pvt] _reflectFee #
-[Pvt] _getValues
-[Pvt] _getTBasics
-[Pvt] getTTransferAmount

Contract functions details

-[Pvt] _getRBasics
-[Pvt] _getRTransferAmount
-[Pvt] _getRate
-[Pvt] _getCurrentSupply
-[Pvt] _sendToCharity
-[Pvt] removeAllFee
-[Pvt] restoreAllFee
-[Pvt] _getTaxFee

($) = payable function
= non-constant function

Issues Checking Status

No.	Title	Status
1.	Unlocked Compiler Version	Passed
2.	Missing Input Validation	Passed
3.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
4.	Possible delays in data delivery	Passed
5.	Oracle calls.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Medium Issue
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	Private use data leaks.	Passed
13.	Malicious Event log.	Passed
14.	Scoping and Declarations.	Passed
15.	Uninitialized storage pointers.	Passed
16.	Arithmetic accuracy.	Passed
17.	Design Logic.	Passed
18.	Safe Open Zeppelin contracts implementation and usage.	Passed
19.	Incorrect Naming State Variable	Passed
20.	Too old version	Passed

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution.

Security Issues

- **Critical Severity Issues**

  No critical severity issue found.

- **High Severity Issues**

  No high severity issue found.

- **Medium Severity Issues**

  One medium severity issue found.

1. Out of gas limit

- **Description**

  The function includeInReward() uses the loop to find and remove addresses from the _excluded list. Function will be aborted with OUT_OF_GAS exception if there will be a long excluded addresses list.

  The function _getCurrentSupply also uses the loop for evaluating total supply. It also could be aborted with OUT_OF_GAS exception if there will be a long excluded addresseses list.

- **Recommendation**

  Use EnumerableSet instead of array or do not use long arrays.

- **Low Severity Issues**

  No low severity issue found.

Centralization

Owner privileges :

- BAD APES Contract:
    - Owner can transfer/ renounce ownership.
    - Owner can exclude/include account.
    - Owner can set charity account.
    - Owner can update fee.

This smart contract has some functions which can be executed by the admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble, as smart contract ownership has not been renounced. Following are Admin functions:

- transferOwnership
- renounceOwnership
- excludeAccount
- includeAccount
- setasCharityAccount
- updateFee

Conclusion

Smart contract contains medium severity issues! The further transfer and operations with the fund raised are not related to this particular contract.

HackSafe note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.