HackSafe

Smart Contract Security Audit Report

DAIKOKUTEN SAMA

April 2023

Security Status

★★★★½

www.hacksafe.io

Audit Details

Audited project
DAIKOKUTEN SAMA

Deployer address
0xC904988723de486C38be9d8FcD82710eaE6267A7

Client contacts
DAIKOKUTEN SAMA 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 DAIKOKUTEN SAMA to perform an audit of smart contracts:

- https://bscscan.com/token/0x834613c64522725b23b458aF04ED1590D189962F#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 05.04.2023

Type	: Utility
Contract name	: DKKS
Contract address	: 0x834613c64522725b23b458aF04ED1590D189962F
Total supply	: 1,000,000,000,000,000
Token Ticker	: DKKS
Decimals	: 9
Token Holders	: 8,418
Transactions count	: 26,090
Compiler version	: v0.8.4+commit.c7e474f2
Contract deployer address	: 0xC904988723de486C38be9d8FcD82710eaE6267A7
Owner address	: 0Xc904988723de486c38be9d8fcd82710eae6267a7

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, 0 medium and 0 low and some very low-level issues. These issues are not critical ones.

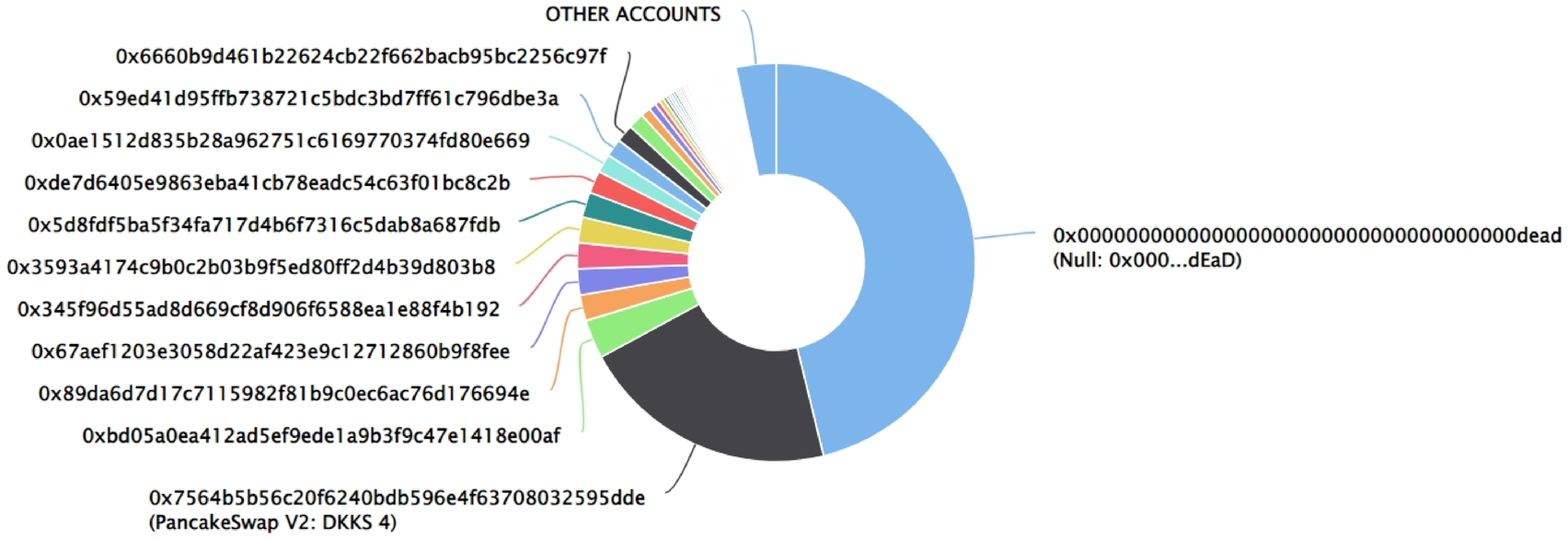
DAIKOKUTEN SAMA Token Distribution

The top 100 holders collectively own 96.73%
(967,261,216,162,899.00 Tokens) of DAIKOKUTEN SAMA

Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 8,418

DAIKOKUTEN SAMA Top 100 Token Holders

Source: BscScan.com



DAIKOKUTEN SAMA Token Top 20 Token Holders

(A total of 967,261,216,162,899.00 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	Null: 0x000...dEaD	462,051,407,084,448.065447175	46.2051%
2	PancakeSwap V2: DKKS 4	209,171,144,473,731.503277432	20.9171%
3	0xbd05a0ea412ad5ef9ede1a9b3f9c47e1418e00af	31,611,804,941,947.553191831	3.1612%
4	0x89da6d7d17c7115982f81b9c0ec6ac76d176694e	21,043,184,124,053.079279169	2.1043%
5	0x67aef1203e3058d22af423e9c12712860b9f8fee	21,038,404,003,983.915826132	2.1038%
6	0x345f96d55ad8d669cf8d906f6588ea1e88f4b192	20,884,287,257,921.250008425	2.0884%
7	0x3593a4174c9b0c2b03b9f5ed80ff2d4b39d803b8	20,840,749,654,245.912280587	2.0841%
8	0x5d8fdf5ba5f34fa717d4b6f7316c5dab8a687fdb	20,548,394,447,492.037838367	2.0548%
9	0xde7d6405e9863eba41cb78eadc54c63f01bc8c2b	18,132,322,733,624.544444584	1.8132%
10	0x0ae1512d835b28a962751c6169770374fd80e669	15,157,401,522,818.747066713	1.5157%
11	0x59ed41d95ffb738721c5bdc3bd7ff61c796dbe3a	14,432,719,026,958.450875469	1.4433%
12	0x6660b9d461b22624cb22f662bacb95bc2256c97f	14,293,214,390,352.240383583	1.4293%
13	0x39ed1d9138897e557d166a05aa7a4a1108bcf44a	13,005,574,700,705.321614568	1.3006%
14	0x7885125653a94104924b01ced47d420095d81a36	7,868,341,182,560.296457784	0.7868%
15	0xcd223b59bc62f436f1a83e40f8f6600138f0d660	5,912,451,581,821.898365274	0.5912%
16	0x0b2348dd2d8ff1642d623eb42987e2406a07b22e	3,981,015,734,346.914109257	0.3981%
17	0x6c350549846f3954146e6debe098d7888f016bc6	3,790,939,754,601.247861591	0.3791%
18	0xa16f53588974cdc862b93a96a3835c0f9adf4cd0	2,678,526,156,499.222994043	0.2679%
19	0x6773e8bdcbde40d3663d42e76b1802da91caeb56	2,367,428,710,094.659074579	0.2367%
20	0xf244ed9355b459f4efe20b3e40d7b3393b4f20e6	2,325,640,051,866.744157644	0.2326%

DAIKOKUTEN SAMA Token Distribution

DAIKOKUTEN SAMA Contract Overview

Contract functions details

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

+[Int] IERC20
    -[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] < Constructor> #
    -[Pub] owner
    -[Pub] renounceOwnership #
     - modifiers: onlyOwner
    -[Pub] transferOwnership #
     - modifiers: onlyOwner
    -[Pub] getUnlockTime
    -[Pub] getTime

Contract functions details

-[Pub] lock #
 - modifiers: onlyOwner
-[Pub] unlock #

+[Int] IUniswapV2Factory
-[Ext] feeTo
-[Ext] feeToSetter
-[Ext] getPair
-[Ext] allPairs
-[Ext] allPairsLength
-[Ext] createPair #
-[Ext] setFeeTo #
-[Ext] setFeeToSetter #

+[Int] IUniswapV2Pair
-[Ext] name
-[Ext] symbol
-[Ext] decimals
-[Ext] totalSupply
-[Ext] balanceOf
-[Ext] allowance
-[Ext] approve #
-[Ext] transfer #
-[Ext] transferFrom #
-[Ext] DOMAIN_SEPARATOR
-[Ext] PERMIT_TYPEHASH
-[Ext] nonces
-[Ext] permit #
-[Ext] MINIMUM_LIQUIDITY
-[Ext] factory
-[Ext] token0
-[Ext] token1
-[Ext] getReserves
-[Ext] price0CumulativeLast
-[Ext] price1CumulativeLast
-[Ext] kLast
-[Ext] burn #
-[Ext] swap #
-[Ext] skim #

Contract functions details

- -[Ext] sync #
- -[Ext] initialize #

+[Int] IUniswapV2Router01
- -[Ext] factory
- -[Ext] WETH
- -[Ext] addLiquidity #
- -[Ext] addLiquidityETH ($)
- -[Ext] removeLiquidity #
- -[Ext] removeLiquidityETH #
- -[Ext] removeLiquidityWithPermit #
- -[Ext] removeLiquidityETHWithPermit #
- -[Ext] swapExactTokensForTokens #
- -[Ext] swapTokensForExactTokens #
- -[Ext] swapExactETHForTokens ($)
- -[Ext] swapTokensForExactETH #
- -[Ext] swapExactTokensForETH #
- -[Ext] swapETHForExactTokens ($)
- -[Ext] quote
- -[Ext] getAmountOut
- -[Ext] getAmountIn
- -[Ext] getAmountsOut
- -[Ext] getAmountsIn

+[Int] IUniswapV2Router02 (IUniswapV2Router01)
- -[Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
- -[Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #
- -[Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
- -[Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
- -[Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #

+DKKS (Context, IERC20, Ownable)
- -[Pub] < Constructor>#
- -[Pub] name
- -[Pub] symbol
- -[Pub] decimals
- -[Pub] totalSupply
- -[Pub] balanceOf
- -[Pub] transfer #
- -[Pub] allowance

Contract functions details

-[Pub] approve #
-[Pub] transferFrom #
-[Pub] increaseAllowance #
-[Pub] decreaseAllowance #
-[Pub] isExcludedFromReward
-[Pub] totalFees
-[Pub] minimumTokensBeforeSwapAmount
-[Pub] deliver #
-[Pub] reflectionFromToken
-[Pub] tokenFromReflection
-[Pub] excludeFromReward #
 - modifiers: onlyOwner
-[Ext] includeInReward #
 - modifiers: onlyOwner
-[Pvt] _approve #
-[Pvt] _transfer #
-[Pvt] swapTokens #
 - modifiers: lockTheSwap
-[Pvt] swapTokensForEth #
-[Pvt] swapETHForTokens #
-[Pvt] addLiquidity #
-[Pvt] _tokenTransfer #
-[Pvt] _transferStandard #
-[Pvt] _transferToExcluded #
-[Pvt] _transferFromExcluded #
-[Pvt] _transferBothExcluded #
-[Pvt] _reflectFee #
-[Pvt] _getValues
-[Pvt] _getTValues
-[Pvt] _getRValues
-[Pvt] _getRate
-[Pvt] _getCurrentSupply
-[Pvt] _takeLiquidity #
-[Pvt] calculateTaxFee
-[Pvt] calculateLiquidityFee
-[Pvt] removeAllFee #
-[Pvt] restoreAllFee #
-[Pub] isExcludedFromFee

Contract functions details

-[Pub] excludeFromFee #
  - modifiers: onlyOwner
-[Pub] includeInFee #
  - modifiers: onlyOwner
-[Ext] setBuyTaxFeePercent #
  - modifiers: onlyOwner
-[Ext] setSellTaxFeePercent #
  - modifiers: onlyOwner
-[Ext] setBuyLiquidityFeePercent #
  - modifiers: onlyOwner
-[Ext] setSellLiquidityFeePercent #
  - modifiers: onlyOwner
-[Ext] setMaxTxAmount #
  - modifiers: onlyOwner
-[Ext] setMarketingDivisor #
  - modifiers: onlyOwner
-[Ext] setNumTokensSellToAddToLiquidity #
  - modifiers: onlyOwner
-[Ext] setMaxTokenHolder #
  - modifiers: onlyOwner
-[Ext] setMarketingAddress #
  - modifiers: onlyOwner
-[Pub] changeRouterVersion #
  - modifiers: onlyOwner
-[Pub] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner
-[Ext] prepareForPreSale #
  - modifiers: onlyOwner
-[Ext] goLive #
  - modifiers: onlyOwner
-[Pub] transferBatch #
-[Pvt] transferToAddressETH #
-[Ext]< Fallback> ($)

($) = payable function
= non-constant function

Issues Checking Status

No.	Title	Status
1.	Compiler error	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.	Passed
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**
  No Medium severity issues found.
- **Low Severity Issues**
  No low severity issue found.

Centralization

Owner privileges (In the period when the owner is not renounced) :

- DAIKOKUTEN SAMA Contract:
    - Owner can change buy/sell tax and liquidity fees.
    - Owner can change maximum transaction amount.
    - Owner can exclude from the fee.
    - Owner can change marketingDivisor.
    - Owner can change minimum number of tokens to add to liquidity.
    - Owner can change _maxTokenHolder value.
    - Owner can change marketing address.
    - Owner can change router address.
    - Owner can lock and unlock. By the way, using these functions the owner could retake privileges even after the ownership was renounced.
    - Owner can enable presale and live setting presets.

Conclusion

Smart contract contains no medium severity issues! Liquidity pair contract's security is not checked due to out of scope.

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.