AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Baby Floki2.0

DATED : 11 July 23'

FUNCTIONAL TESTING

Centralization – Enabling Trades

Severity: High

function: enableTrading

Status: Not Resolved

Overview:

Owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens.

```
function enableTrading() public onlyOwner {
    tradingOpen = true;
}
```

Suggestion

Its suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guearanee enabling of trades.

AUDIT SUMMARY

Project name – Baby Floki2.0

Date: 11 July, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: **Passed with High Risk**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
A line by line code review has been performed by audit ace team.

2- BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/token/0x394034d387179EE95Acb15fB7ACff6f02E7B5E7b

Token Information

Token Name : Baby Floki2.0

Token Symbol: BABY FLOKI2.0

Decimals: 9

Token Supply: 1,000,000,000,000

Token Address:
0x89B840668848fa5fAcbb7e18B874aDD835d8E101

Checksum:
7c4a5e21f7eee42445f904ea4b19498c4864a85b

Owner:
0xe970045CF57c76f4389570Cb911aE33D06A636f9
(at time of writing the audit)

Deployer:
0xe970045CF57c76f4389570Cb911aE33D06A636f9

TOKEN OVERVIEW

Fees:

Buy Fees: 0-10%

Sell Fees: 0-10%

Transfer Fees: 0%

Fees Privilege: No Fees

Ownership: owned

Minting: none

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges:- Initial distribution of the tokens

- Modifying fees

- enabling trades

AUDIT METHODOLOGY

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

- Review of the specifications, sources, and instructions provided to Auditace 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 Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing 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.

VULNERABILITY CHECKLIST

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] Using throw

CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
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.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	1
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE

POINTS TO NOTE

- Owner is able to change fees in range 0-10% for buy and sells (0% transfer tax)
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to disable trades
- Owner is not able to limit buy/sell/transfer/wallet amounts
- Owner is not able to mint new tokens
- Owner must enable trades manually

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
Token	Interface			
└	transferFrom	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
IUniswapV2Factory	Interface			
└	createPair	External ❗	🛑	NO ❗
IUniswapV2Router02	Interface			
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
Context	Implementation			
└	_msgSender	Internal 🔒		
SafeMath	Library			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
Babyfloki2	Implementation	Context, IERC20, Ownable		
└	<Constructor>	Public ❗	🛑	NO ❗

CONTRACT ASSESMENT

└	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 ❗
└	tokenFromReflection	Private 🔐		
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	swapTokensForEth	Private 🔐	🛑	lockTheSwap
└	sendETHToFee	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	rescueForeignTokens	Public ❗	🛑	onlyDev
└	setNewDevAddress	Public ❗	🛑	onlyDev
└	setNewMarketingAddress	Public ❗	🛑	onlyDev
└	_transferStandard	Private 🔐	🛑	
└	_takeTeam	Private 🔐	🛑	
└	_reflectFee	Private 🔐	🛑	
└	<Receive Ether>	External ❗	💵 |NO ❗
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	manualswap	External ❗	🛑 |NO ❗
└	manualsend	External ❗	🛑 |NO ❗
└	setFee	Public ❗	🛑	onlyDev
└	enableTrading	Public ❗	🛑	onlyOwner
└	toggleSwap	Public ❗	🛑	onlyDev
└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reentrancy in Babyfloki2.transferFrom(address,address,uint256) (contracts/Token.sol#221-229):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#222)
                - _developmentAddress.transfer(amount.div(2)) (contracts/Token.sol#294)
                - _marketingAddress.transfer(amount.div(2)) (contracts/Token.sol#295)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (contracts/Token.sol#223-227
)
                - _allowances[owner][spender] = amount (contracts/Token.sol#240)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#241)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (contracts/Token.sol
#223-227)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

Variable Babyfloki2._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#326) is too similar to Babyfloki2._transferStandard(add
ress,address,uint256).tTransferAmount (contracts/Token.sol#326)
Variable Babyfloki2._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#374) is too similar to Babyfloki2._getTValues(uint256
,uint256,uint256).tTransferAmount (contracts/Token.sol#362)
Variable Babyfloki2._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#326) is too similar to Babyfloki2._getValues(uint256).t
TransferAmount (contracts/Token.sol#349)
Variable Babyfloki2._getValues(uint256).rTransferAmount (contracts/Token.sol#351) is too similar to Babyfloki2._getTValues(uint256,uint256,uint256).tTransf
erAmount (contracts/Token.sol#362)
Variable Babyfloki2._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#326) is too similar to Babyfloki2._getTValues(uint256,u
int256,uint256).tTransferAmount (contracts/Token.sol#362)
Variable Babyfloki2._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#374) is too similar to Babyfloki2._getValues(uint256)
.tTransferAmount (contracts/Token.sol#349)
Variable Babyfloki2._getValues(uint256).rTransferAmount (contracts/Token.sol#351) is too similar to Babyfloki2._getValues(uint256).tTransferAmount (contrac
ts/Token.sol#349)
Variable Babyfloki2._getValues(uint256).rTransferAmount (contracts/Token.sol#351) is too similar to Babyfloki2._transferStandard(address,address,uint256).t
TransferAmount (contracts/Token.sol#326)
Variable Babyfloki2._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#374) is too similar to Babyfloki2._transferStandard(a
ddress,address,uint256).tTransferAmount (contracts/Token.sol#326)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Babyfloki2.slitherConstructorConstantVariables() (contracts/Token.sol#125-429) uses literals with too many digits:
        - _tTotal = 1000000 * 10 ** 6 * 10 ** 9 (contracts/Token.sol#134)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

Babyfloki2._tOwned (contracts/Token.sol#129) is never used in Babyfloki2 (contracts/Token.sol#125-429)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

Babyfloki2.uniswapV2Pair (contracts/Token.sol#155) should be immutable
Babyfloki2.uniswapV2Router (contracts/Token.sol#154) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0x2b6342b2b8bbabd3bf8c4b6b647e8a18118c4d6991cd2be7ee7b8f42cc29c9d9

2- Buying when excluded (0% tax) (passed):

https://testnet.bscscan.com/tx/0xab0c641ad4fee6aa133f355518746fe9e6bd468df369ccc5abaf22f576b2655d

3- Selling when excluded (0% tax) (passed):

https://testnet.bscscan.com/tx/0x0cd43b13f65b64d5458f9af6ed0446ff0df1278b5a6ab907510dffa31d5188b3

4- Transferring when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x0ec8271624715582d0b00f77a5e43017d6523e03afb8fc1aabfa5b241cd8b308

5- Buying from a regular wallet (0-10% tax) (passed):

https://testnet.bscscan.com/tx/0x17c3a02fc3596a02e25ee80d4bc3f1075d5bec0989108d35338cd2577e89e606

6- Selling from a regular wallet (0-10% tax) (passed):

https://testnet.bscscan.com/tx/0xc52670f931196e809a2ad3b67a1da2c26b4ee77fa111bb007cd6d20a494c4418

FUNCTIONAL TESTING

7- Transferring from a regular wallet (0% tax) (passed):

https://testnet.bscscan.com/tx/0xd6557f37a6876f673f39a29a9faf2e34ecd1574ffb2485f3ca2aada82797fdba

7- Internal swap (marketing and development wallets received BNB) (passed):

https://testnet.bscscan.com/address/0x3fdBBcEB8D3AB94f4525D4c5884A748e8B4BD467#internaltx

FUNCTIONAL TESTING

Centralization – Enabling Trades

Severity: High

function: enableTrading

Status: Not Resolved

Overview:

Owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens.

```
function enableTrading() public onlyOwner {
    tradingOpen = true;
}
```

Suggestion

Its suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guearanee enabling of trades.

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace