Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: BabyMusk Token
Platform: Binance Smart Chain
Language: Solidity
Date: December 25th, 2021

Table of contents

Email: audit@EtherAuthority.io

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the BabyMusk team to perform the Security audit of the BabyMusk Token smart contract code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on December 25th, 2021.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

BabyMusk is a standard BEP20 token smart contract with functionalities like dividend rewards, swapping. This audit only considers BabyMusk token smart contracts, and does not cover any other smart contracts in the platform.

Audit scope

Name	**Code Review and Security Analysis Report for BabyMusk Token Smart Contract**
Platform	**BSC / Solidity**
File	BabyMUSK.sol
File MD5 Hash	F291212640A19FFCE8E4121C30349FC2
Online code	0x0b1ff525e092a98210ed150f8b08313f646847d6
Audit Date	December 25th, 2021

Email: audit@EtherAuthority.io

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
Tokenomics: • Name: BabyMUSK • Symbol: BabyMUSK • Decimals: 9 • Total Supply: 420 Quadrillion	**YES, This is valid.**
• Dividend Rewards Fee: 10% • Liquidity Fee: 5% • Marketing Fee: 7% • Sell fee Increase Factor: 1.2% • Gas for Processing: 300000 • Swap Tokens At Amount: 1 Billion	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**

Audit Summary

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

We used various tools like Slither, Solhint 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 Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 1 medium and 3 low and some very low level issues. These issues are not critical ones.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Moderated
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Moderated
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderated
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Moderated
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Moderated
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Email: audit@EtherAuthority.io

Code Quality

This audit scope has 1 smart contract file. Smart contract contains Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in BabyMusk Token are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the BabyMusk Token.

The BabyMusk Token team has **not** provided scenario and unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are **not well** commented on smart contracts.

Documentation

We were given a BabyMusk Token smart contracts code in the form of a BSCscan web link.The hash of that code is mentioned above in the table.

As mentioned above, code parts are **not well** commented. So it is not easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	receive	external	Passed	No Issue
3	whitelistDxSale	write	access only Owner	No Issue
4	setSellFeeFactor	external	access only Owner	No Issue
5	setMinTokenBalForDividends	external	access only Owner	No Issue
6	excludeFromRewards	external	access only Owner	No Issue
7	setSellStatus	external	access only Owner	No Issue
8	ChangeDividendRewardsFee	external	Function input parameters lack of check, ChangeDividend RewardsFee function has meaningless functionality	Refer Audit Findings
9	changeDividendAddress	external	Function input parameters lack of check	Refer Audit Findings
10	changeLiquidityFee	external	Function input parameters lack of check	Refer Audit Findings
11	changeMarketingFee	external	Function input parameters lack of check	Refer Audit Findings
12	changeMarketingWallet	external	Function input parameters lack of check	Refer Audit Findings
13	blacklistAddress	write	Critical operation lacks event log	Refer Audit Findings
14	unBlockAddress	write	Critical operation lacks event log	Refer Audit Findings
15	updateDividendTracker	write	access only Owner	No Issue
16	updateUniswapV2Router	write	access only Owner	No Issue
17	excludeFromFees	write	access only Owner	No Issue
18	excludeMultipleAccountsFrom Fees	write	Infinite Loop	Refer Audit Findings

19	setAutomatedMarketMakerPair	write	access only Owner	No Issue
20	_setAutomatedMarketMakerPair	write	Passed	No Issue
21	updateLiquidityWallet	write	access only Owner	No Issue
22	updateGasForProcessing	write	access only Owner	No Issue
23	updateClaimWait	external	access only Owner	No Issue
24	getClaimWait	external	Passed	No Issue
25	isExcludedFromRewards	read	Passed	No Issue
26	getTotalDividendsDistributed	external	Passed	No Issue
27	isExcludedFromFees	read	Passed	No Issue
28	withdrawableDividendOf	read	Passed	No Issue
29	dividendTokenBalanceOf	read	Passed	No Issue
30	getAccountDividendsInfo	external	Passed	No Issue
31	getAccountDividendsInfoAtIndex	external	Passed	No Issue
32	processDividendTracker	external	Passed	No Issue
33	claim	external	Passed	No Issue
34	getLastProcessedIndex	external	Passed	No Issue
35	getNumberOfDividendTokenHolders	external	Passed	No Issue
36	sendBNBToMarketing	write	Passed	No Issue
37	_transfer	internal	Passed	No Issue
38	swapAndLiquify	write	Passed	No Issue
39	swapTokensForEth	write	Passed	No Issue
40	swapTokensForDividend	write	Passed	No Issue
41	addLiquidity	write	Passed	No Issue
42	swapAndSendDividends	write	Passed	No Issue
43	name	read	Passed	No Issue
44	symbol	read	Passed	No Issue
45	decimals	read	Passed	No Issue
46	totalSupply	read	Passed	No Issue
47	balanceOf	read	Passed	No Issue
48	transfer	write	Passed	No Issue
49	allowance	read	Passed	No Issue
50	approve	write	Passed	No Issue
51	transferFrom	write	Passed	No Issue
52	increaseAllowance	write	Passed	No Issue
53	decreaseAllowance	write	Passed	No Issue
54	_transfer	internal	Passed	No Issue
55	_mint	internal	Passed	No Issue
56	_burn	internal	Passed	No Issue
57	_approve	internal	Passed	No Issue
58	_setupDecimals	internal	Passed	No Issue
59	_beforeTokenTransfer	internal	Passed	No Issue

60	owner	read	Passed	No Issue
61	onlyOwner	modifier	Passed	No Issue
62	renounceOwnership	write	access only Owner	No Issue
63	transferOwnership	write	access only Owner	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

(1) Total Fee is not getting update:

```
constructor() public ERC20("BabyMUSK", "BabyMUSK") {
    uint256 _dividendRewardsFee = 10;
    uint256 _liquidityFee = 5;
    uint256 _marketingFee = 7;

    dividendRewardsFee = _dividendRewardsFee;
    liquidityFee = _liquidityFee;
    marketingFee = _marketingFee;
    totalFees = _dividendRewardsFee.add(_liquidityFee).add(marketingFee);
```

```
uint256 marketingSwap = contractTokenBalance.mul(marketingFee).div(totalFees);
sendBNBToMarketing(marketingSwap);

uint256 swapTokens = contractTokenBalance.mul(liquidityFee).div(totalFees);
swapAndLiquify(swapTokens);
```

Total fees is the sum of all 3 fees - dividend rewards fee, liquidity fee and marketing fee. On changing any of these fees, totalfees has not been updated. So that can lead wrong calculation.

Resolution: We suggest updating totalfees on update of each fee.

Low

(1) Function input parameters lack of check:

Variable validation is not performed in below functions :

- ChangeDividendRewardsFee
- changeDividendAddress
- changeLiquidityFee
- changeMarketingFee

- changeMarketingWallet
- setSellFeeFactor
- setMinTokenBalForDividends

Resolution: There should be some limit for values as this affects the calculation and if the address passed, then it should not be 0 addresses.

(2) Infinite Loop:

In the excludeMultipleAccountsFromFees function, the loop does not have an upper length limit ,which costs more gas.

Resolution: We suggest using some limit for accounts while executing this function.

(3) Critical operation lacks event log:

Missing event log for:

- blacklistAddress
- unBlockAddress

Resolution: We suggest writing an event log for listed events.

Very Low / Informational / Best practices:

(1) Solidity version:

Using the latest solidity will prevent any compiler level bugs.

Resolution: We suggest using version >0.8.0.

(2) Multiple pragma defined:

There are multiple pragma defined.

Resolution: We suggest using only one pragma at the top of the code.

(3) ChangeDividendRewardsFee function has meaningless functionality

ChangeDividendRewardsFee function is used to change _dividendRewardsFee, but that variable has not been used anywhere except the constructor. So that function is useless.

Resolution: We suggest either removing that function or adding some meaningful functionality into that function.

Centralization

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. Following are Admin functions:

- whitelistDxSale: The Owner can add _presaleAddress and _routerAddress in the whitelist.
- setSellFeeFactor: The Owner can set a new sell fee factor.
- setMinTokenBalForDividends: The Owner can set a minimum token balance for dividends.
- excludeFromRewards: The Owner can set account for exclude From Dividends.
- setSellStatus: The Owner can set sell status enabled.
- ChangeDividendRewardsFee: The Owner can change dividend rewards fee.
- changeDividendAddress: The Owner can change dividend address.
- changeLiquidityFee: The Owner can change the liquidity fee.
- changeMarketingFee: The Owner can change the marketing fee.
- changeMarketingWallet: The Owner can change the marketing wallet address.
- blacklistAddress: The Owner can set the wallet address in blacklist.
- unBlockAddress: The Owner can unBlock wallet addresses in the blacklist.
- updateDividendTracker: The Owner can update dividend tracker address.
- updateUniswapV2Router: The Owner can update uniswap v2 router address.
- excludeFromFees: The Owner can check if BabyMUSK: Account is already the value of 'excluded' or not.
- excludeMultipleAccountsFromFees: The Owner can exclude multiple accounts from fees.
- setAutomatedMarketMakerPair: The Owner can set an automated market maker pair.
- updateLiquidityWallet: The Owner can update the liquidity wallet address.

- updateGasForProcessing: The Owner can check if the BabyMUSK: gasForProcessing must be between 200,000 and 500,000 and the value is not the same then set a new gas value.
- updateClaimWait: The Owner can update the claim wait value.

Conclusion

We were given a contract code. And we have used all possible tests based on given objects as files. We observed some issues in the smart contracts, but they are not critical ones. So, **it's good to go to production**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured"**.

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

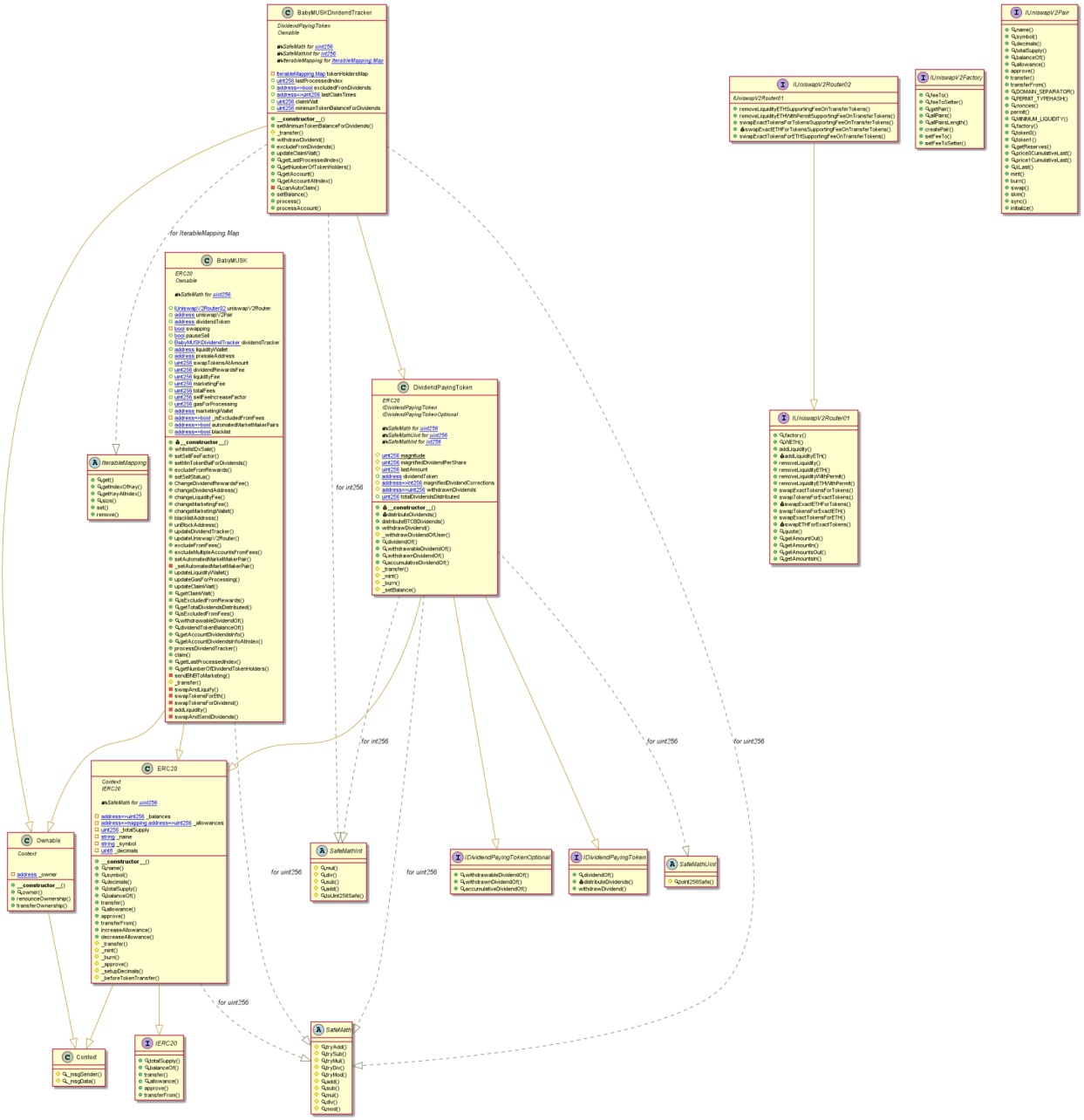
Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can’t guarantee explicit security of the audited smart contracts.

Appendix

Code Flow Diagram - BabyMusk Token



Slither Results Log

Slither log >> BabyMUSK.sol

```
INFO:Detectors:
BabyMUSK.sendBNBToMarketing(uint256) (BabyMUSK.sol#1474-1477) sends eth to arbitrary user
        Dangerous calls:
        - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
BabyMUSK.addLiquidity(uint256,uint256) (BabyMUSK.sol#1634-1649) sends eth to arbitrary user
        Dangerous calls:
        - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (BabyMUSK
40-1647)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations
INFO:Detectors:
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
                - success = IERC20(dividendToken).transfer(address(dividendTracker),dividends) (BabyMUSK.sol#1654)
                - dividendTracker.distributeBTCBDividends(dividends) (BabyMUSK.sol#1657)
                - uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,recipient,block.timestamp
MUSK.sol#1624-1630)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        State variables written after the call(s):
        - super._transfer(from,address(this),fees) (BabyMUSK.sol#1548)
                - _balances[sender] = _balances[sender].sub(amount,ERC20: transfer amount exceeds balance) (BabyMUSK.sol#674)
                - _balances[recipient] = _balances[recipient].add(amount) (BabyMUSK.sol#675)
        - super._transfer(from,to,amount) (BabyMUSK.sol#1551)
                - _balances[sender] = _balances[sender].sub(amount,ERC20: transfer amount exceeds balance) (BabyMUSK.sol#674)
```

```
                - _balances[recipient] = _balances[recipient].add(amount) (BabyMUSK.sol#675)
        - swapping = false (BabyMUSK.sol#1525)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566) performs a multiplication on the result of a division:
        -fees = amount.mul(totalFees).div(100) (BabyMUSK.sol#1539)
        -fees = fees.mul(sellFeeIncreaseFactor).div(100) (BabyMUSK.sol#1543)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
Contract locking ether found:
        Contract BabyMUSKDividendTracker (BabyMUSK.sol#1664-1875) has payable functions:
         - DividendPayingToken.receive() (BabyMUSK.sol#1000-1001)
         - DividendPayingToken.distributeDividends() (BabyMUSK.sol#1016-1027)
         - IDividendPayingToken.distributeDividends() (BabyMUSK.sol#425)
        But does not have a function to withdraw the ether
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#contracts-that-lock-ether
INFO:Detectors:
Reentrancy in DividendPayingToken._withdrawDividendOfUser(address) (BabyMUSK.sol#1051-1067):
        External calls:
        - success = IERC20(dividendToken).transfer(user,_withdrawableDividend) (BabyMUSK.sol#1056)
        State variables written after the call(s):
        - withdrawnDividends[user] = withdrawnDividends[user].sub(_withdrawableDividend) (BabyMUSK.sol#1059)
Reentrancy in BabyMUSK.updateDividendTracker(address) (BabyMUSK.sol#1332-1347):
        External calls:
        - newDividendTracker.excludeFromDividends(address(newDividendTracker)) (BabyMUSK.sol#1339)
        - newDividendTracker.excludeFromDividends(address(this)) (BabyMUSK.sol#1340)
        - newDividendTracker.excludeFromDividends(owner()) (BabyMUSK.sol#1341)
        - newDividendTracker.excludeFromDividends(address(uniswapV2Router)) (BabyMUSK.sol#1342)
        State variables written after the call(s):
        - dividendTracker = newDividendTracker (BabyMUSK.sol#1346)
Reentrancy in BabyMUSK.whitelistDxSale(address,address) (BabyMUSK.sol#1275-1282):
        External calls:
        - dividendTracker.excludeFromDividends(_presaleAddress) (BabyMUSK.sol#1277)
        - dividendTracker.excludeFromDividends(_routerAddress) (BabyMUSK.sol#1280)
        State variables written after the call(s):
        - excludeFromFees(_routerAddress,true) (BabyMUSK.sol#1281)
                - _isExcludedFromFees[account] = excluded (BabyMUSK.sol#1357)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
BabyMUSK._transfer(address,address,uint256).iterations (BabyMUSK.sol#1559) is a local variable never initialized
BabyMUSK._transfer(address,address,uint256).claims (BabyMUSK.sol#1559) is a local variable never initialized
BabyMUSK._transfer(address,address,uint256).lastProcessedIndex (BabyMUSK.sol#1559) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
BabyMUSK.claim() (BabyMUSK.sol#1461-1463) ignores return value by dividendTracker.processAccount(msg.sender,false) (BabyMUSK.sol#146
BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566) ignores return value by dividendTracker.process(gas) (BabyMUSK.
9-1564)
BabyMUSK.addLiquidity(uint256,uint256) (BabyMUSK.sol#1634-1649) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAm
ddress(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (BabyMUSK.sol#1640-1647)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
DividendPayingToken.constructor(string,string)._name (BabyMUSK.sol#995) shadows:
        - ERC20._name (BabyMUSK.sol#502) (state variable)
DividendPayingToken.constructor(string,string)._symbol (BabyMUSK.sol#995) shadows:
        - ERC20._symbol (BabyMUSK.sol#503) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
BabyMUSK.whitelistDxSale(address,address)._presaleAddress (BabyMUSK.sol#1275) lacks a zero-check on :
                - presaleAddress = _presaleAddress (BabyMUSK.sol#1276)
BabyMUSK.changeDividendAddress(address).newAddress (BabyMUSK.sol#1306) lacks a zero-check on :
                - dividendToken = newAddress (BabyMUSK.sol#1307)
BabyMUSK.changeMarketingWallet(address).newAddress (BabyMUSK.sol#1318) lacks a zero-check on :
                - marketingWallet = newAddress (BabyMUSK.sol#1319)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
```

```
INFO:Detectors:
Variable 'BabyMUSK._transfer(address,address,uint256).claims (BabyMUSK.sol#1559)' in BabyMUSK._transfer(address,address,uint256) (Ba
sol#1481-1566) potentially used before declaration: ProcessedDividendTracker(iterations,claims,lastProcessedIndex,true,gas,tx.origin
MUSK.sol#1560)
Variable 'BabyMUSK._transfer(address,address,uint256).iterations (BabyMUSK.sol#1559)' in BabyMUSK._transfer(address,address,uint256)
USK.sol#1481-1566) potentially used before declaration: ProcessedDividendTracker(iterations,claims,lastProcessedIndex,true,gas,tx.or
BabyMUSK.sol#1560)
Variable 'BabyMUSK._transfer(address,address,uint256).lastProcessedIndex (BabyMUSK.sol#1559)' in BabyMUSK._transfer(address,address,
) (BabyMUSK.sol#1481-1566) potentially used before declaration: ProcessedDividendTracker(iterations,claims,lastProcessedIndex,true,g
rigin) (BabyMUSK.sol#1560)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables
INFO:Detectors:
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        State variables written after the call(s):
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - _allowances[owner][spender] = amount (BabyMUSK.sol#736)
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
```

```
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
                - success = IERC20(dividendToken).transfer(address(dividendTracker),dividends) (BabyMUSK.sol#1654)
                - dividendTracker.distributeBTCBDividends(dividends) (BabyMUSK.sol#1657)
                - uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,recipient,block.timestamp
MUSK.sol#1624-1630)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        State variables written after the call(s):
        - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
                - _allowances[owner][spender] = amount (BabyMUSK.sol#736)
Reentrancy in BabyMUSK.constructor() (BabyMUSK.sol#1227-1269):
        External calls:
        - _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BabyMUSK
45-1246)
        State variables written after the call(s):
        - uniswapV2Pair = _uniswapV2Pair (BabyMUSK.sol#1249)
        - uniswapV2Router = _uniswapV2Router (BabyMUSK.sol#1248)
Reentrancy in BabyMUSK.constructor() (BabyMUSK.sol#1227-1269):
        External calls:
        - _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BabyMUSK
45-1246)
        - _setAutomatedMarketMakerPair(_uniswapV2Pair,true) (BabyMUSK.sol#1251)
                - dividendTracker.excludeFromDividends(pair) (BabyMUSK.sol#1381)
        - dividendTracker.excludeFromDividends(address(dividendTracker)) (BabyMUSK.sol#1254)
        - dividendTracker.excludeFromDividends(address(this)) (BabyMUSK.sol#1255)
        - dividendTracker.excludeFromDividends(owner()) (BabyMUSK.sol#1256)
        - dividendTracker.excludeFromDividends(address(_uniswapV2Router)) (BabyMUSK.sol#1257)
        State variables written after the call(s):
        - _mint(owner(),420000000000000000 * (10 ** 9)) (BabyMUSK.sol#1268)
```

```
                - _balances[account] = _balances[account].add(amount) (BabyMUSK.sol#694)
        - excludeFromFees(liquidityWallet,true) (BabyMUSK.sol#1261)
                - _isExcludedFromFees[account] = excluded (BabyMUSK.sol#1357)
        - excludeFromFees(address(this),true) (BabyMUSK.sol#1262)
                - _isExcludedFromFees[account] = excluded (BabyMUSK.sol#1357)
        - _mint(owner(),420000000000000000 * (10 ** 9)) (BabyMUSK.sol#1268)
                - _totalSupply = _totalSupply.add(amount) (BabyMUSK.sol#693)
Reentrancy in BabyMUSKDividendTracker.processAccount(address,bool) (BabyMUSK.sol#1863-1873):
        External calls:
        - amount = _withdrawDividendOfUser(account) (BabyMUSK.sol#1864)
                - success = IERC20(dividendToken).transfer(user,_withdrawableDividend) (BabyMUSK.sol#1056)
        State variables written after the call(s):
        - lastClaimTimes[account] = block.timestamp (BabyMUSK.sol#1867)
Reentrancy in BabyMUSK.swapAndLiquify(uint256) (BabyMUSK.sol#1568-1589):
        External calls:
        - swapTokensForEth(half) (BabyMUSK.sol#1580)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - addLiquidity(otherHalf,newBalance) (BabyMUSK.sol#1586)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (BabyMUSK.sol#1586)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        State variables written after the call(s):
        - addLiquidity(otherHalf,newBalance) (BabyMUSK.sol#1586)
                - _allowances[owner][spender] = amount (BabyMUSK.sol#736)
Reentrancy in BabyMUSK.whitelistDxSale(address,address) (BabyMUSK.sol#1275-1282):
        External calls:
        - dividendTracker.excludeFromDividends(_presaleAddress) (BabyMUSK.sol#1277)
        State variables written after the call(s):
        - excludeFromFees(_presaleAddress,true) (BabyMUSK.sol#1278)
                - _isExcludedFromFees[account] = excluded (BabyMUSK.sol#1357)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in BabyMUSK._setAutomatedMarketMakerPair(address,bool) (BabyMUSK.sol#1376-1385):
```

```
        External calls:
        - dividendTracker.excludeFromDividends(pair) (BabyMUSK.sol#1381)
        Event emitted after the call(s):
        - SetAutomatedMarketMakerPair(pair,value) (BabyMUSK.sol#1384)
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BabyMUSK.sol#737)
                - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
        - SwapAndLiquify(half,newBalance,otherHalf) (BabyMUSK.sol#1588)
                - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
                - success = IERC20(dividendToken).transfer(address(dividendTracker),dividends) (BabyMUSK.sol#1654)
```

```
                - uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,recipient,block.timestamp
MUSK.sol#1624-1630)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BabyMUSK.sol#737)
                - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
        - SendDividends(tokens,dividends) (BabyMUSK.sol#1658)
                - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
        - Transfer(sender,recipient,amount) (BabyMUSK.sol#676)
                - super._transfer(from,address(this),fees) (BabyMUSK.sol#1548)
        - Transfer(sender,recipient,amount) (BabyMUSK.sol#676)
                - super._transfer(from,to,amount) (BabyMUSK.sol#1551)
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
                - success = IERC20(dividendToken).transfer(address(dividendTracker),dividends) (BabyMUSK.sol#1654)
                - dividendTracker.distributeBTCBDividends(dividends) (BabyMUSK.sol#1657)
                - uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,recipient,block.timestamp
MUSK.sol#1624-1630)
        - dividendTracker.setBalance(address(from),balanceOf(from)) (BabyMUSK.sol#1553)
        - dividendTracker.setBalance(address(to),balanceOf(to)) (BabyMUSK.sol#1554)
        - dividendTracker.process(gas) (BabyMUSK.sol#1559-1564)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
```

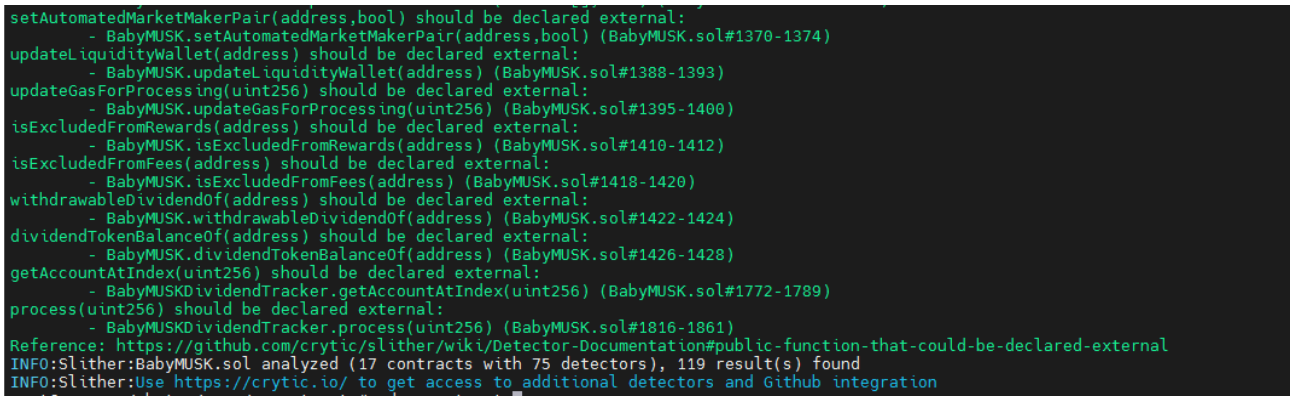
```
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        Event emitted after the call(s):
        - ProcessedDividendTracker(iterations,claims,lastProcessedIndex,true,gas,tx.origin) (BabyMUSK.sol#1560)
Reentrancy in BabyMUSK.constructor() (BabyMUSK.sol#1227-1269):
        External calls:
        - _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BabyMUSK
45-1246)
        - _setAutomatedMarketMakerPair(_uniswapV2Pair,true) (BabyMUSK.sol#1251)
                - dividendTracker.excludeFromDividends(pair) (BabyMUSK.sol#1381)
        Event emitted after the call(s):
        - SetAutomatedMarketMakerPair(pair,value) (BabyMUSK.sol#1384)
                - _setAutomatedMarketMakerPair(_uniswapV2Pair,true) (BabyMUSK.sol#1251)
Reentrancy in BabyMUSK.constructor() (BabyMUSK.sol#1227-1269):
        External calls:
        - _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (BabyMUSK
45-1246)
        - _setAutomatedMarketMakerPair(_uniswapV2Pair,true) (BabyMUSK.sol#1251)
                - dividendTracker.excludeFromDividends(pair) (BabyMUSK.sol#1381)
        - dividendTracker.excludeFromDividends(address(dividendTracker)) (BabyMUSK.sol#1254)
        - dividendTracker.excludeFromDividends(address(this)) (BabyMUSK.sol#1255)
        - dividendTracker.excludeFromDividends(owner()) (BabyMUSK.sol#1256)
        - dividendTracker.excludeFromDividends(address(_uniswapV2Router)) (BabyMUSK.sol#1257)
        Event emitted after the call(s):
        - ExcludeFromFees(account,excluded) (BabyMUSK.sol#1359)
                - excludeFromFees(address(this),true) (BabyMUSK.sol#1262)
        - ExcludeFromFees(account,excluded) (BabyMUSK.sol#1359)
                - excludeFromFees(liquidityWallet,true) (BabyMUSK.sol#1261)
        - Transfer(address(0),account,amount) (BabyMUSK.sol#695)
                - _mint(owner(),420000000000000000 * (10 ** 9)) (BabyMUSK.sol#1268)
Reentrancy in BabyMUSKDividendTracker.processAccount(address,bool) (BabyMUSK.sol#1863-1873):
        External calls:
        - amount = _withdrawDividendOfUser(account) (BabyMUSK.sol#1864)
                - success = IERC20(dividendToken).transfer(user,_withdrawableDividend) (BabyMUSK.sol#1056)
        Event emitted after the call(s):
```

```
        - Claim(account,amount,automatic) (BabyMUSK.sol#1868)
Reentrancy in BabyMUSK.processDividendTracker(uint256) (BabyMUSK.sol#1456-1459):
        External calls:
        - (iterations,claims,lastProcessedIndex) = dividendTracker.process(gas) (BabyMUSK.sol#1457)
        Event emitted after the call(s):
        - ProcessedDividendTracker(iterations,claims,lastProcessedIndex,false,gas,tx.origin) (BabyMUSK.sol#1458)
Reentrancy in BabyMUSK.swapAndLiquify(uint256) (BabyMUSK.sol#1568-1589):
        External calls:
        - swapTokensForEth(half) (BabyMUSK.sol#1580)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestam
yMUSK.sol#1602-1608)
        - addLiquidity(otherHalf,newBalance) (BabyMUSK.sol#1586)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (BabyMUSK.sol#1586)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BabyMUSK.sol#737)
                - addLiquidity(otherHalf,newBalance) (BabyMUSK.sol#1586)
        - SwapAndLiquify(half,newBalance,otherHalf) (BabyMUSK.sol#1588)
Reentrancy in BabyMUSK.swapAndSendDividends(uint256) (BabyMUSK.sol#1651-1660):
        External calls:
        - swapTokensForDividend(tokens,address(this)) (BabyMUSK.sol#1652)
                - uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount,0,path,recipient,block.timestamp
MUSK.sol#1624-1630)
        - success = IERC20(dividendToken).transfer(address(dividendTracker),dividends) (BabyMUSK.sol#1654)
        - dividendTracker.distributeBTCBDividends(dividends) (BabyMUSK.sol#1657)
        Event emitted after the call(s):
        - SendDividends(tokens,dividends) (BabyMUSK.sol#1658)
Reentrancy in BabyMUSK.updateDividendTracker(address) (BabyMUSK.sol#1332-1347):
        External calls:
        - newDividendTracker.excludeFromDividends(address(newDividendTracker)) (BabyMUSK.sol#1339)
        - newDividendTracker.excludeFromDividends(address(this)) (BabyMUSK.sol#1340)
        - newDividendTracker.excludeFromDividends(owner()) (BabyMUSK.sol#1341)
        - newDividendTracker.excludeFromDividends(address(uniswapV2Router)) (BabyMUSK.sol#1342)
        - UpdateDividendTracker(newAddress,address(dividendTracker)) (BabyMUSK.sol#1344)
Reentrancy in BabyMUSK.whitelistDxSale(address,address) (BabyMUSK.sol#1275-1282):
        External calls:
        - dividendTracker.excludeFromDividends(_presaleAddress) (BabyMUSK.sol#1277)
        Event emitted after the call(s):
        - ExcludeFromFees(account,excluded) (BabyMUSK.sol#1359)
                - excludeFromFees(_presaleAddress,true) (BabyMUSK.sol#1278)
Reentrancy in BabyMUSK.whitelistDxSale(address,address) (BabyMUSK.sol#1275-1282):
        External calls:
        - dividendTracker.excludeFromDividends(_presaleAddress) (BabyMUSK.sol#1277)
        - dividendTracker.excludeFromDividends(_routerAddress) (BabyMUSK.sol#1280)
        Event emitted after the call(s):
        - ExcludeFromFees(account,excluded) (BabyMUSK.sol#1359)
                - excludeFromFees(_routerAddress,true) (BabyMUSK.sol#1281)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
BabyMUSKDividendTracker.getAccount(address) (BabyMUSK.sol#1727-1770) uses timestamp for comparisons
        Dangerous comparisons:
        - nextClaimTime > block.timestamp (BabyMUSK.sol#1767-1769)
BabyMUSKDividendTracker.canAutoClaim(uint256) (BabyMUSK.sol#1791-1797) uses timestamp for comparisons
        Dangerous comparisons:
        - lastClaimTime > block.timestamp (BabyMUSK.sol#1792)
        - block.timestamp.sub(lastClaimTime) >= claimWait (BabyMUSK.sol#1796)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566) compares to a boolean constant:
        -to == uniswapV2Pair && pauseSell == true && from != owner() (BabyMUSK.sol#1498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
INFO:Detectors:
Context._msgData() (BabyMUSK.sol#79-82) is never used and should be removed
DividendPayingToken._transfer(address,address,uint256) (BabyMUSK.sol#1107-1113) is never used and should be removed
ERC20._setupDecimals(uint8) (BabyMUSK.sol#747-749) is never used and should be removed
SafeMath.div(uint256,uint256,string) (BabyMUSK.sol#939-942) is never used and should be removed
SafeMath.mod(uint256,uint256) (BabyMUSK.sol#901-904) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (BabyMUSK.sol#959-962) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (BabyMUSK.sol#773-777) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (BabyMUSK.sol#809-812) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (BabyMUSK.sol#819-822) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (BabyMUSK.sol#794-802) is never used and should be removed
SafeMath.trySub(uint256,uint256) (BabyMUSK.sol#784-787) is never used and should be removed
SafeMathInt.div(int256,int256) (BabyMUSK.sol#460-466) is never used and should be removed
SafeMathInt.mul(int256,int256) (BabyMUSK.sol#450-458) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Function IUniswapV2Router01.WETH() (BabyMUSK.sol#87) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (BabyMUSK.sol#251) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (BabyMUSK.sol#252) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (BabyMUSK.sol#269) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (BabyMUSK.sol#1073) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (BabyMUSK.sol#1080) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (BabyMUSK.sol#1087) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (BabyMUSK.sol#1097) is not in mixedCase
Constant DividendPayingToken.magnitude (BabyMUSK.sol#972) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter BabyMUSK.whitelistDxSale(address,address)._presaleAddress (BabyMUSK.sol#1275) is not in mixedCase
Parameter BabyMUSK.whitelistDxSale(address,address)._routerAddress (BabyMUSK.sol#1275) is not in mixedCase
Function BabyMUSK.ChangeDividendRewardsFee(uint256) (BabyMUSK.sol#1302-1304) is not in mixedCase
Parameter BabyMUSKDividendTracker.getAccount(address)._account (BabyMUSK.sol#1727) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (BabyMUSK.sol#80)" inContext (BabyMUSK.sol#74-83)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in BabyMUSK._transfer(address,address,uint256) (BabyMUSK.sol#1481-1566):
        External calls:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        External calls sending eth:
        - sendBNBToMarketing(marketingSwap) (BabyMUSK.sol#1517)
                - marketingWallet.transfer(address(this).balance) (BabyMUSK.sol#1476)
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,liquidityWallet,block.timestamp) (
K.sol#1640-1647)
        State variables written after the call(s):
        - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
```

```
                - _allowances[owner][spender] = amount (BabyMUSK.sol#736)
        - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
                - _allowances[owner][spender] = amount (BabyMUSK.sol#736)
        - super._transfer(from,address(this),fees) (BabyMUSK.sol#1548)
                - _balances[sender] = _balances[sender].sub(amount,ERC20: transfer amount exceeds balance) (BabyMUSK.sol#674)
                - _balances[recipient] = _balances[recipient].add(amount) (BabyMUSK.sol#675)
        - super._transfer(from,to,amount) (BabyMUSK.sol#1551)
                - _balances[sender] = _balances[sender].sub(amount,ERC20: transfer amount exceeds balance) (BabyMUSK.sol#674)
                - _balances[recipient] = _balances[recipient].add(amount) (BabyMUSK.sol#675)
        - swapping = false (BabyMUSK.sol#1525)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (BabyMUSK.sol#737)
                - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
        - Approval(owner,spender,amount) (BabyMUSK.sol#737)
                - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
        - ProcessedDividendTracker(iterations,claims,lastProcessedIndex,true,gas,tx.origin) (BabyMUSK.sol#1560)
        - SendDividends(tokens,dividends) (BabyMUSK.sol#1658)
                - swapAndSendDividends(sellTokens) (BabyMUSK.sol#1523)
        - SwapAndLiquify(half,newBalance,otherHalf) (BabyMUSK.sol#1588)
                - swapAndLiquify(swapTokens) (BabyMUSK.sol#1520)
        - Transfer(sender,recipient,amount) (BabyMUSK.sol#676)
                - super._transfer(from,to,amount) (BabyMUSK.sol#1551)
        - Transfer(sender,recipient,amount) (BabyMUSK.sol#676)
                - super._transfer(from,address(this),fees) (BabyMUSK.sol#1548)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (BabyMUSK.s
is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (B
.sol#93)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (BabyMUSK.sol#1052) is too similar to BabyMUSKDi
racker.getAccount(address).withdrawableDividends (BabyMUSK.sol#1732)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
BabyMUSK.constructor() (BabyMUSK.sol#1227-1269) uses literals with too many digits:
        - _mint(owner(),420000000000000000 * (10 ** 9)) (BabyMUSK.sol#1268)
BabyMUSK.updateGasForProcessing(uint256) (BabyMUSK.sol#1395-1400) uses literals with too many digits:
        - require(bool,string)(newValue >= 200000 && newValue <= 500000,BabyMUSK: gasForProcessing must be between 200,000 and 500,0
```

```
byMUSK.sol#1396)
BabyMUSK.slitherConstructorVariables() (BabyMUSK.sol#1150-1662) uses literals with too many digits:
        - swapTokensAtAmount = 1000000000 * (10 ** 9) (BabyMUSK.sol#1168)
BabyMUSK.slitherConstructorVariables() (BabyMUSK.sol#1150-1662) uses literals with too many digits:
        - gasForProcessing = 300000 (BabyMUSK.sol#1179)
BabyMUSKDividendTracker.getAccountAtIndex(uint256) (BabyMUSK.sol#1772-1789) uses literals with too many digits:
        - (0x0000000000000000000000000000000000000000,- 1,- 1,0,0,0,0,0) (BabyMUSK.sol#1783)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
DividendPayingToken.lastAmount (BabyMUSK.sol#975) is never used in BabyMUSKDividendTracker (BabyMUSK.sol#1664-1875)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
BabyMUSK.swapTokensAtAmount (BabyMUSK.sol#1168) should be constant
DividendPayingToken.dividendToken (BabyMUSK.sol#977) should be constant
DividendPayingToken.lastAmount (BabyMUSK.sol#975) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
get(IterableMapping.Map,address) should be declared external:
        - IterableMapping.get(IterableMapping.Map,address) (BabyMUSK.sol#296-298)
getIndexOfKey(IterableMapping.Map,address) should be declared external:
        - IterableMapping.getIndexOfKey(IterableMapping.Map,address) (BabyMUSK.sol#300-305)
getKeyAtIndex(IterableMapping.Map,uint256) should be declared external:
        - IterableMapping.getKeyAtIndex(IterableMapping.Map,uint256) (BabyMUSK.sol#307-309)
size(IterableMapping.Map) should be declared external:
        - IterableMapping.size(IterableMapping.Map) (BabyMUSK.sol#313-315)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (BabyMUSK.sol#383-386)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (BabyMUSK.sol#392-396)
name() should be declared external:
        - ERC20.name() (BabyMUSK.sol#524-526)
symbol() should be declared external:
        - ERC20.symbol() (BabyMUSK.sol#532-534)
decimals() should be declared external:
        - ERC20.decimals() (BabyMUSK.sol#549-551)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (BabyMUSK.sol#575-578)
```

```
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (BabyMUSK.sol#583-585)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (BabyMUSK.sol#594-597)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (BabyMUSK.sol#612-616)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (BabyMUSK.sol#630-633)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (BabyMUSK.sol#649-652)
distributeDividends() should be declared external:
        - DividendPayingToken.distributeDividends() (BabyMUSK.sol#1016-1027)
distributeBTCBDividends(uint256) should be declared external:
        - DividendPayingToken.distributeBTCBDividends(uint256) (BabyMUSK.sol#1030-1041)
withdrawDividend() should be declared external:
        - BabyMUSKDividendTracker.withdrawDividend() (BabyMUSK.sol#1697-1699)
        - DividendPayingToken.withdrawDividend() (BabyMUSK.sol#1045-1047)
dividendOf(address) should be declared external:
        - DividendPayingToken.dividendOf(address) (BabyMUSK.sol#1073-1075)
withdrawnDividendOf(address) should be declared external:
        - DividendPayingToken.withdrawnDividendOf(address) (BabyMUSK.sol#1087-1089)
whitelistDxSale(address,address) should be declared external:
        - BabyMUSK.whitelistDxSale(address,address) (BabyMUSK.sol#1275-1282)
blacklistAddress(address) should be declared external:
        - BabyMUSK.blacklistAddress(address) (BabyMUSK.sol#1322-1324)
unBlockAddress(address) should be declared external:
        - BabyMUSK.unBlockAddress(address) (BabyMUSK.sol#1326-1328)
updateDividendTracker(address) should be declared external:
        - BabyMUSK.updateDividendTracker(address) (BabyMUSK.sol#1332-1347)
updateUniswapV2Router(address) should be declared external:
        - BabyMUSK.updateUniswapV2Router(address) (BabyMUSK.sol#1349-1353)
excludeMultipleAccountsFromFees(address[],bool) should be declared external:
        - BabyMUSK.excludeMultipleAccountsFromFees(address[],bool) (BabyMUSK.sol#1362-1368)
```



Solidity Static Analysis

BabyMUSK.sol

Security

Transaction origin:

Use of tx.origin: "tx.origin" is useful only in very exceptional cases.
If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.
more
Pos: 1631:90:

Transaction origin:

Use of tx.origin: "tx.origin" is useful only in very exceptional cases.
If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.
more
Pos: 1733:97:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in BabyMUSK.swapTokensForEth(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1764:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in BabyMUSK.swapTokensForDividend(uint256,address): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1786:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 1802:12:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 1819:12:

Gas & Economy

Gas costs:

Gas requirement of function BabyMUSK.transferOwnership is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 455:4:

Gas costs:

Gas requirement of function BabyMUSK.whitelistDxSale is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 1448:4:

Gas costs:

Gas requirement of function BabyMUSK.setMinTokenBalForDividends is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 1463:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point.
Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 1536:8:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 283:4:

Miscellaneous

Constant/View/Pure functions:

IterableMapping.set(struct IterableMapping.Map,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 362:4:

Constant/View/Pure functions:

BabyMUSK.getClaimWait() : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 1579:4:

Constant/View/Pure functions:

BabyMUSK.isExcludedFromRewards(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 1583:4:

Similar variable names:

ERC20._setupDecimals(uint8) : Variables have very similar names "_decimals" and "decimals_". Note: Modifiers are currently not considered by this static analysis.

Pos: 877:20:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 1506:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

more

Pos: 1510:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.

Pos: 1203:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.

Pos: 1266:11:

Similar variable names:

ERC20._mint(address,uint256) : Variables have very similar names "account" and "amount". Note: Modifiers are currently not considered by this static analysis.

Pos: 823:39:

Solhint Linter

BabyMUSK.sol

```
BabyMUSK.sol:11:1: Error: Compiler version ^0.6.0 does not satisfy
the r semver requirement
BabyMUSK.sol:89:1: Error: Compiler version >=0.6.0 <0.8.0 does not
satisfy the r semver requirement
BabyMUSK.sol:114:1: Error: Compiler version >=0.6.2 does not satisfy
the r semver requirement
BabyMUSK.sol:118:5: Error: Function name must be in mixedCase
BabyMUSK.sol:211:1: Error: Compiler version >=0.6.2 does not satisfy
the r semver requirement
BabyMUSK.sol:256:1: Error: Compiler version >=0.5.0 does not satisfy
the r semver requirement
BabyMUSK.sol:275:1: Error: Compiler version >=0.5.0 does not satisfy
the r semver requirement
BabyMUSK.sol:292:5: Error: Function name must be in mixedCase
BabyMUSK.sol:293:5: Error: Function name must be in mixedCase
BabyMUSK.sol:310:5: Error: Function name must be in mixedCase
BabyMUSK.sol:330:1: Error: Compiler version ^0.6.6 does not satisfy
the r semver requirement
BabyMUSK.sol:396:1: Error: Compiler version ^0.6.0 does not satisfy
the r semver requirement
BabyMUSK.sol:463:1: Error: Compiler version ^0.6.0 does not satisfy
the r semver requirement
BabyMUSK.sol:488:1: Error: Compiler version ^0.6.0 does not satisfy
the r semver requirement
BabyMUSK.sol:529:1: Error: Compiler version ^0.6.6 does not satisfy
the r semver requirement
BabyMUSK.sol:575:1: Error: Compiler version ^0.6.6 does not satisfy
the r semver requirement
BabyMUSK.sol:593:1: Error: Compiler version ^0.6.0 does not satisfy
the r semver requirement
BabyMUSK.sol:894:94: Error: Code contains empty blocks
BabyMUSK.sol:900:1: Error: Compiler version ^0.6.0 does not satisfy
the r semver requirement
BabyMUSK.sol:1117:1: Error: Compiler version ^0.6.6 does not satisfy
the r semver requirement
BabyMUSK.sol:1140:29: Error: Constant name must be in capitalized
SNAKE_CASE
BabyMUSK.sol:1163:89: Error: Code contains empty blocks
BabyMUSK.sol:1168:30: Error: Code contains empty blocks
BabyMUSK.sol:1227:9: Error: Possible reentrancy vulnerabilities.
Avoid state changes after transfer.
BabyMUSK.sol:1320:1: Error: Compiler version ^0.6.6 does not satisfy
the r semver requirement
BabyMUSK.sol:1323:1: Error: Contract has 18 states declarations but
allowed no more than 15
BabyMUSK.sol:1444:32: Error: Code contains empty blocks
BabyMUSK.sol:1475:5: Error: Function name must be in mixedCase
BabyMUSK.sol:1631:91: Error: Avoid to use tx.origin
BabyMUSK.sol:1661:53: Error: Use double quotes for string literals
BabyMUSK.sol:1726:72: Error: Code contains empty blocks
BabyMUSK.sol:1726:81: Error: Code contains empty blocks
```

```
BabyMUSK.sol:1727:68: Error: Code contains empty blocks
BabyMUSK.sol:1727:77: Error: Code contains empty blocks
BabyMUSK.sol:1733:98: Error: Avoid to use tx.origin
BabyMUSK.sol:1735:19: Error: Code contains empty blocks
BabyMUSK.sol:1780:13: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:1802:13: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:1819:13: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:1940:58: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:1941:71: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:1965:28: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:1969:16: Error: Avoid to make time-based decisions in
your business logic
BabyMUSK.sol:2040:39: Error: Avoid to make time-based decisions in
your business logic
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

Email: audit@EtherAuthority.io