



Smart Contract Audit

FOR
BABY PEPA

DATED : 15 April 23'



AUDIT SUMMARY

Project name - BABY PEPA

Date: 15 April, 2023

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

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	2	2	0	11
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 done on BSC Test network, each test has its transaction has 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 Link:

<https://testnet.bscscan.com/token/0x6837f492645276d0c9be69c41a40a63ea5db1bf6>



Token Information

Token Name : BABY PEPA

Token Symbol: BABYP

Decimals: 18

Token Supply: 420,000,000,000,000,000

Token Address:

0x9924e70AFD132Ee75c5Cd463978545572B401eA3

Checksum:

d95059c44466b7b3d6f87f342219f00f07117aa9

Owner:

0xC319dAA7D06F8CfD5A673482403C2a1b55f4c24a

(at time of audit)



TOKEN OVERVIEW

Fees:

Buy Fees: Up to 12%

Sell Fees: Up to 12%

Transfer Fees: Up to 12%

Fees Privilege: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: including and excluding form fee - changing swap threshold - enabling trades - modifying fees - changing max wallet



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 is exercised 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



Return values of low-level calls



Gasless Send



Private modifier



Using block.timestamp



Multiple Sends



Re-entrancy



Using Suicide



Tautology or contradiction



Gas Limit and Loops



Timestamp Dependence



Address hardcoded



Revert/require functions



Exception Disorder



Use of tx.origin



Using inline assembly



Integer overflow/underflow



Divide before multiply



Dangerous strict equalities



Missing Zero Address Validation



Using SHA3



Compiler version not fixed



Using throw



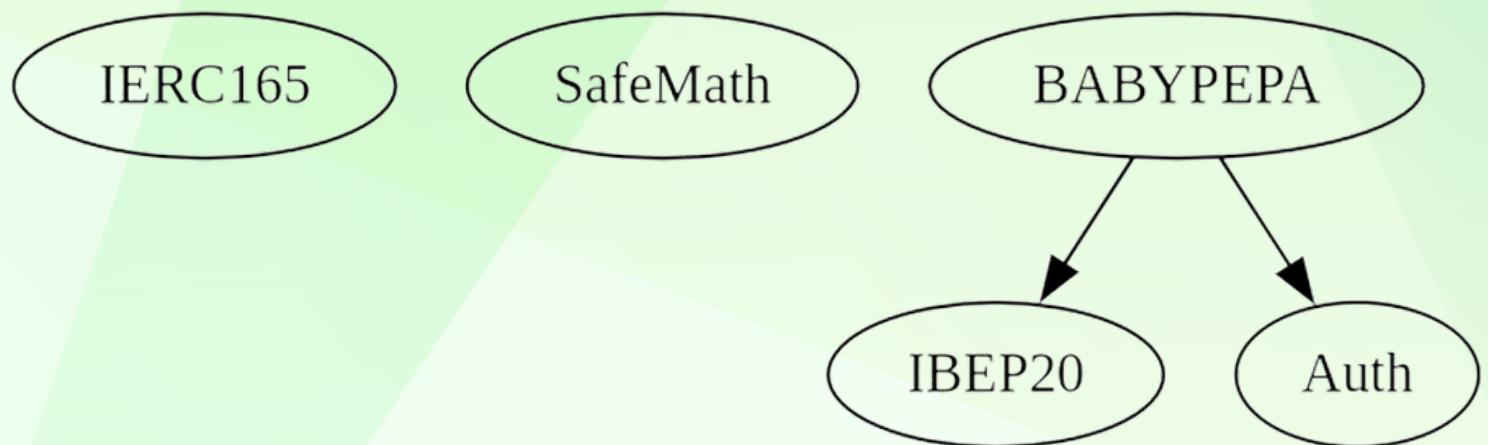
CLASSIFICATION OF RISK

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	2
◆ Medium-Risk	2
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	11

INHERITANCE TREE





POINTS TO NOTE

- Owner is not able to set buy/sell/transfer fees over 12%
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is able to buys/transfers
- Owner is not able to mint new tokens



CONTRACT ASSESSMENT

Contract	Type	Bases			
	L	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC165	Interface				
L	supportsInterface	External !	NO !		
SafeMath	Library				
L	tryAdd	Internal			
L	trySub	Internal			
L	tryMul	Internal			
L	tryDiv	Internal			
L	tryMod	Internal			
L	add	Internal			
L	sub	Internal			
L	mul	Internal			
L	div	Internal			
L	mod	Internal			
L	sub	Internal			
L	div	Internal			
L	mod	Internal			
IBEP20	Interface				
L	totalSupply	External !	NO !		
L	decimals	External !	NO !		
L	symbol	External !	NO !		
L	name	External !	NO !		
L	getOwner	External !	NO !		
L	balanceOf	External !	NO !		
L	transfer	External !		NO !	
L	burning	External !		NO !	
L	allowance	External !	NO !		
L	approve	External !		NO !	
L	transferFrom	External !		NO !	
Auth	Implementation				
L	<Constructor>	Public !		NO !	
L	isOwner	Public !	NO !		
L	transferOwnership	Public !		onlyOwner	
IDEXFactory	Interface				
L	createPair	External !		NO !	
IDEXRouter	Interface				



CONTRACT ASSESSMENT

L factory External ! NO !
L WETH External ! NO !
L addLiquidity External ! ● NO !
L addLiquidityETH External ! \$ NO !
L swapExactTokensForTokensSupportingFeeOnTransferTokens External ! ● NO !
L swapExactETHForTokensSupportingFeeOnTransferTokens External ! \$ NO !
L swapExactTokensForETHSupportingFeeOnTransferTokens External ! ● NO !
IDividendDistributor Interface
L setDistributionCriteria External ! ● NO !
L setShare External ! ● NO !
L deposit External ! \$ NO !
L process External ! ● NO !
DividendDistributor Implementation IDividendDistributor
L <Constructor> Public ! ● NO !
L setDistributionCriteria External ! ● onlyToken
L setShare External ! ● onlyToken
L deposit External ! \$ onlyToken
L process External ! ● onlyToken
L shouldDistribute Internal 🔒
L distributeDividend Internal 🔒 ●
L claimDividend External ! ● onlyToken
L getUnpaidEarnings Public ! NO !
L getCumulativeDividends Internal 🔒
L addShareholder Internal 🔒 ●
L removeShareholder Internal 🔒 ●
L setDividendTokenAddress External ! ● onlyToken
BABYPEPA Implementation IBEP20, Auth
L <Constructor> Public ! ● Auth
L <Receive Ether> External ! \$ NO !
L totalSupply External ! NO !
L decimals External ! NO !
L symbol External ! NO !
L name External ! NO !
L getOwner External ! NO !
L balanceOf Public ! NO !
L allowance External ! NO !
L approve Public ! ● NO !
L approveMax External ! ● NO !
L savetokens External ! ● onlyOwner

CONTRACT ASSESSMENT

L	burning	External	!		●	NO	!	
L	transfer	External	!		●	NO	!	
L	setMaxWalletTokens	External	!		●		●	onlyOwner
L	transferFrom	External	!		●	NO	!	
L	_transferFrom	Internal			●			
L	_basicTransfer	Internal			●			
L	shouldTakeFee	Internal						
L	shouldTakeFee	Internal						
L	getTotalFee	Public	!		NO	!		
L	getMultipliedFee	Public	!		NO	!		
L	takeFee	Internal			●			
L	shouldSwapBack	Internal						
L	swapBack	Internal			●		swapping	
L	buyTokens	Internal			●		swapping	
L	launched	Internal						
L	launch	Public	!		●		onlyOwner	
L	setIsDividendExempt	External	!		●		onlyOwner	
L	setIsFeeExempt	External	!		●		onlyOwner	
L	setIsMaxWalletExempt	External	!		●		onlyOwner	
L	setBuyFees	External	!		●		onlyOwner	
L	setSellFees	External	!		●		onlyOwner	
L	setFeeReceivers	External	!		●		onlyOwner	
L	setSwapBackSettings	External	!		●		onlyOwner	
L	setTargetLiquidity	External	!		●		onlyOwner	
L	manualSend	External	!		●	NO	!	
L	setDistributionCriteria	External	!		●		onlyOwner	
L	claimDividend	External	!		●	NO	!	
L	getUnpaidEarnings	Public	!		NO	!		
L	setDistributorSettings	External	!		●		onlyOwner	
L	getCirculatingSupply	Public	!		NO	!		
L	getLiquidityBacking	Public	!		NO	!		
L	isOverLiquified	Public	!		NO	!		

Legend

	Symbol	Meaning	
-----	-----	-----	-----
●	Function can modify state		
	Function is payable		



STATIC ANALYSIS

```
Parameter BABYPEPA.setFeeReceivers(address,address,address)._marketingFeeReceiver (contracts/Token.sol#984) is not in mixedCase
Parameter BABYPEPA.setFeeReceivers(address,address,address)._projectFeeReceiver (contracts/Token.sol#985) is not in mixedCase
Parameter BABYPEPA.setSwapBackSettings(bool,uint256)._enabled (contracts/Token.sol#993) is not in mixedCase
Parameter BABYPEPA.setSwapBackSettings(bool,uint256)._amount (contracts/Token.sol#994) is not in mixedCase
Parameter BABYPEPA.setTargetLiquidity(uint256,uint256)._target (contracts/Token.sol#1001) is not in mixedCase
Parameter BABYPEPA.setTargetLiquidity(uint256,uint256)._denominator (contracts/Token.sol#1002) is not in mixedCase
Parameter BABYPEPA.setDistributionCriteria(uint256,uint256)._minPeriod (contracts/Token.sol#1015) is not in mixedCase
Parameter BABYPEPA.setDistributionCriteria(uint256,uint256)._minDistribution (contracts/Token.sol#1016) is not in mixedCase
Variable BABYPEPA.REWARD (contracts/Token.sol#489) is not in mixedCase
Variable BABYPEPA.WBNB (contracts/Token.sol#490) is not in mixedCase
Variable BABYPEPA.DEAD (contracts/Token.sol#491) is not in mixedCase
Variable BABYPEPA.ZERO (contracts/Token.sol#492) is not in mixedCase
Constant BABYPEPA._name (contracts/Token.sol#494) is not in UPPER CASE WITH underscores
Constant BABYPEPA._symbol (contracts/Token.sol#495) is not in UPPER CASE WITH underscores
Constant BABYPEPA._decimals (contracts/Token.sol#496) is not in UPPER CASE WITH underscores
Variable BABYPEPA._totalSupply (contracts/Token.sol#497) is not in mixedCase
Variable BABYPEPA._maxMeltedToken (contracts/Token.sol#499) is not in mixedCase
Variable BABYPEPA._balances (contracts/Token.sol#501) is not in mixedCase
Variable BABYPEPA._allowances (contracts/Token.sol#502) is not in mixedCase
Variable BABYPEPA._Project (contracts/Token.sol#537) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IDEXRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountDesired (contracts/Token.sol#235) is too similar to IDEXRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#236)
Variable BABYPEPA._maxWalletTokens (contracts/Token.sol#499) is too similar to BABYPEPA.setMaxWalletTokens(uint256).maxWalletTokens (contracts/Token.sol#656)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

BABYPEPA.slitherConstructorVariables() (contracts/Token.sol#486-1054) uses literals with too many digits:
- _totalSupply = 420000000000000000 * (10 ** _decimals) (contracts/Token.sol#497)
BABYPEPA.slitherConstructorVariables() (contracts/Token.sol#486-1054) uses literals with too many digits:
- _distributorGas = 300000 (contracts/Token.sol#548)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

BABYPEPA.REWARD (contracts/Token.sol#489) is never used in BABYPEPA (contracts/Token.sol#486-1054)
BABYPEPA._token (contracts/Token.sol#543) is never used in BABYPEPA (contracts/Token.sol#486-1054)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

BABYPEPA.DEAD (contracts/Token.sol#491) should be constant
BABYPEPA._Project (contracts/Token.sol#537) should be constant
BABYPEPA.REWARD (contracts/Token.sol#489) should be constant
BABYPEPA.WBNB (contracts/Token.sol#490) should be constant
BABYPEPA.ZERO (contracts/Token.sol#492) should be constant
BABYPEPA._totalSupply (contracts/Token.sol#497) should be constant
BABYPEPA._buyFeeDenominator (contracts/Token.sol#515) should be constant
BABYPEPA._emergBlock (contracts/Token.sol#541) should be constant
BABYPEPA._maxFee (contracts/Token.sol#509) should be constant
BABYPEPA._sellFeeDenominator (contracts/Token.sol#522) should be constant
BABYPEPA._token (contracts/Token.sol#543) should be constant
DividendDistributor.WBNB (contracts/Token.sol#304) should be constant
DividendDistributor.dividendsPerShareAccuracyFactor (contracts/Token.sol#317) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

BABYPEPA._distributor (contracts/Token.sol#547) should be immutable
BABYPEPA._pair (contracts/Token.sol#536) should be immutable
BABYPEPA._router (contracts/Token.sol#535) should be immutable
DividendDistributor._token (contracts/Token.sol#295) should be immutable
DividendDistributor._router (contracts/Token.sol#305) 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

1- Adding liquidity (**passed**):

<https://testnet.bscscan.com/tx/0x5eef0635cfe231bf329de58bc677ba4f710d0781cc7e44a790a0b9a9296ce83f>

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

<https://testnet.bscscan.com/tx/0x5fdd133025041282bc3b2d31544acf14ef4baf969e76a5edf8b9bcea8df5e24c>

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

<https://testnet.bscscan.com/tx/0x55a81a1a21597e7eb3d601d679b77d5dfbcc40c34e747d4a7db47643e25033f1>

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

<https://testnet.bscscan.com/tx/0x977d14c32c3deebeefdbc6cba98bbcd534fa0f7729733e221f7fa9612939ef63>

5- Buying when not excluded from fees (up to 12% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x2fcda056a16ff2903263f2a49ec631c67f1831e546fe557eba58733f82934740>

6- Selling when not excluded from fees (up to 12% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x21331b19c2903ef7a294d48ca0bcee5bdd1e7289d30a4dd75beb9081b524d716>

7- Transferring when not excluded from fees (up to 12% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x72fd80a898400360f8f693a04b4c0d436e3881f2f705d9ab75035a11fc6b5656>



FUNCTIONAL TESTING

8- Internal swap (passed):

fee wallets received BNB

<https://testnet.bscscan.com/tx/0xac3501f3b6549d870ca3be72dbdd0d012e5b7119ecd6a479bf22fb61e7685b1e>

9- Distribution of rewards (passed):

reward tokens are distributed between holders, this can be seen in this transaction

<https://testnet.bscscan.com/tx/0xac3501f3b6549d870ca3be72dbdd0d012e5b7119ecd6a479bf22fb61e7685b1e>

8- Auto-liquidity (passed):

LP wallet received LP tokens

<https://testnet.bscscan.com/tx/0xac3501f3b6549d870ca3be72dbdd0d012e5b7119ecd6a479bf22fb61e7685b1e>



MANUAL TESTING

Centralization – Owner must enable trades for public

Severity: High

Function: launch

Lines: 634

Status: No Resolved

Owner must enable trading for holders, if trading remains disabled, no one would be able to sell their tokens.

```
if (!launched() && recipient == pair) {  
    require(_balances[sender] > 0);  
    launch();  
}  
  
function launch() public onlyOwner {  
    require(launchedAt == 0, "Already launched");  
    launchedAt = block.number;  
    launchedAtTimestamp = block.timestamp;  
}
```

Recommendation:

One solution would be transferring ownership of the contract to a trusted 3rd party (pinksale safu developers) to guarantee enabling of the trades.



MANUAL TESTING

Centralization – Max wallet with no safeguards

Severity: High

Function: setMaxWalletTokens

Lines: 634

Status: No Resolved

The current implementation does not have minimum safeguards for max wallet. This means max wallet can be set to 0 disabling all the buys/transfers for non-whitelisted wallets

```
function setMaxWalletTokens(uint256 maxwallettokens) external onlyOwner {  
    _maxWalletToken = maxwallettokens;  
}
```

Recommendation:

Ensure that the maxWalletToken is greater than a reasonable value (0.1% of supply based on PinkSale SAFU criteria).

Example:

```
function setMaxWalletTokens(uint256 maxwallettokens) external onlyOwner {  
    require(_maxWalletToken > totalSupply() / 1000);  
    _maxWalletToken = maxwallettokens;  
}
```



MANUAL TESTING

Logical – Adding liquidity is not possible if launch function is not called manually

Severity: Medium

Function: _transferFrom

Lines: 888

Status: No Resolved

In the process of adding liquidity, **pair** would be equal to recipient, this means that **launch()** function will be called in this section of the code, but since **msg.sender** is pancakeswap router and pancakeswap router is not owner of the contract, adding liquidity will be failed unless ownership is transferred to the router. Although the **launch** function can be called manually but the downside of this issue is that owner has to first enable trading before adding liquidity which enables sniper bots to abuse the initial price of the token.

```
if (!launched() && recipient == pair) {  
    require(_balances[sender] > 0);  
    launch();  
}  
  
function launch() public onlyOwner {  
    require(launchedAt == 0, "Already launched");  
    launchedAt = block.number;  
    launchedAtTimestamp = block.timestamp;  
}
```

Recommendation:

in **launch()** function check if **tx.origin** is owner, this enables adding liquidity only using owner's wallet while **msg.sender** is router



MANUAL TESTING

Centralization – Auto-liquidity generated tokens are sent to an EOA (externally owned account)

Severity: Medium

Function: swapBack

Lines: 864

Status: No Resolved

Pool shares generated from auto-liquidity are sent to an EOA, over time this pool share tokens will get accumulated and can be used to remove a significant portion of the liquidity negatively affecting token price and liquidity

```
if (amountToLiquify > 0) {  
    router.addLiquidityETH{value: amountBNBLiquidity}(  
        address(this),  
        amountToLiquify,  
        0,  
        0,  
        autoLiquidityReceiver,  
        block.timestamp  
    );  
    emit AutoLiquify(amountBNBLiquidity, amountToLiquify);  
}
```

Recommendation:

To address this issue, there are multiple ways:

- Burn new LP tokens
- Lock new LP tokens
- Lock new LP tokens in a farming contract and distribute the received reward among holders

Suggestions & Optimisation

Suggestions

- Change name of **savetokens** function to something that identifies its action, like “withdrawTokens”
- There are two functions that are doing exact same thing. One of this functions can be removed; **shouldTakeFee**, **shouldTakeFee**
- Redundant function : **buyTokens**
- Event if burn and contract are not receiving any tokens (from tax) a Transfer event is still emitted, this may cause confusion for investors
emit Transfer(sender, address(this), feeamount2);
emit Transfer(sender, DEAD, amounttoburn);

Gas optimizations

- Define **router** variable as constant
- Define **pair** variable as constant
- Define **REWARD**, **WBNB**, **DEAD**, **ZERO** as constant variables
- Redundant code here:
 - `_balances[Project] = (_totalSupply * 100) / 100;`
- Can be changed to
 - `_balances[Project] = _totalSupply;`
- Redundant code at `_transfer` function
 - `if (sender != pair && !isOwner(sender)) {}`
- Redundant code at `_transfer` function, since `emergBlock` is always set to true
 - `if (!authorizations[sender] && !authorizations[recipient]) {
 require(emergBlock, "Trading not open yet");
}`



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 specialize 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>
