auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

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 Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: 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 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

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

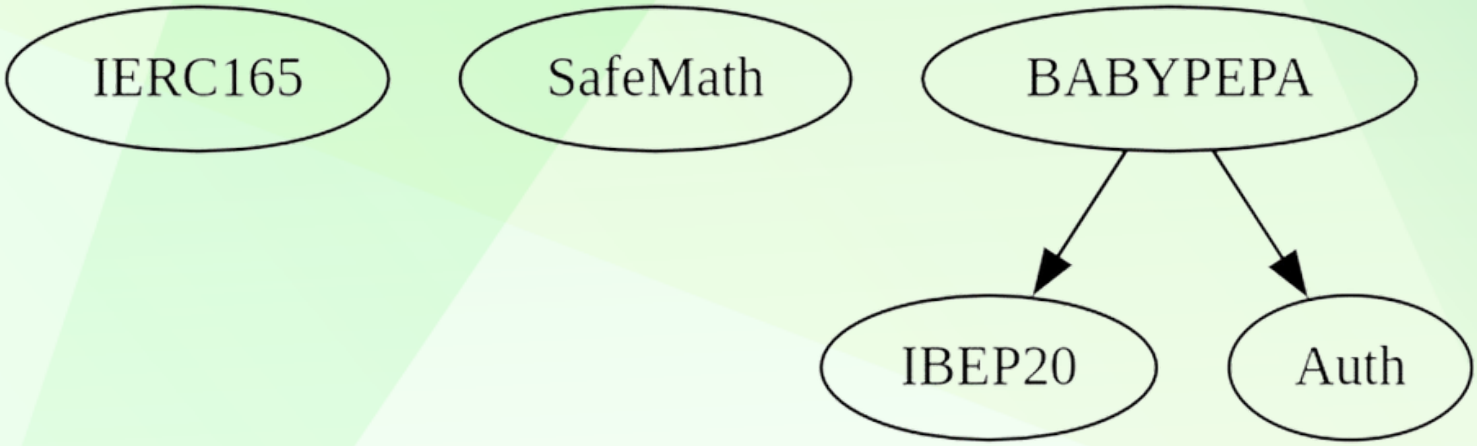
CLASSIFICATION OF RISK

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 disable buy/transfers (but not sells)
- Owner is not able to mint new tokens

CONTRACT ASSESMENT

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

CONTRACT ASSESMENT

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

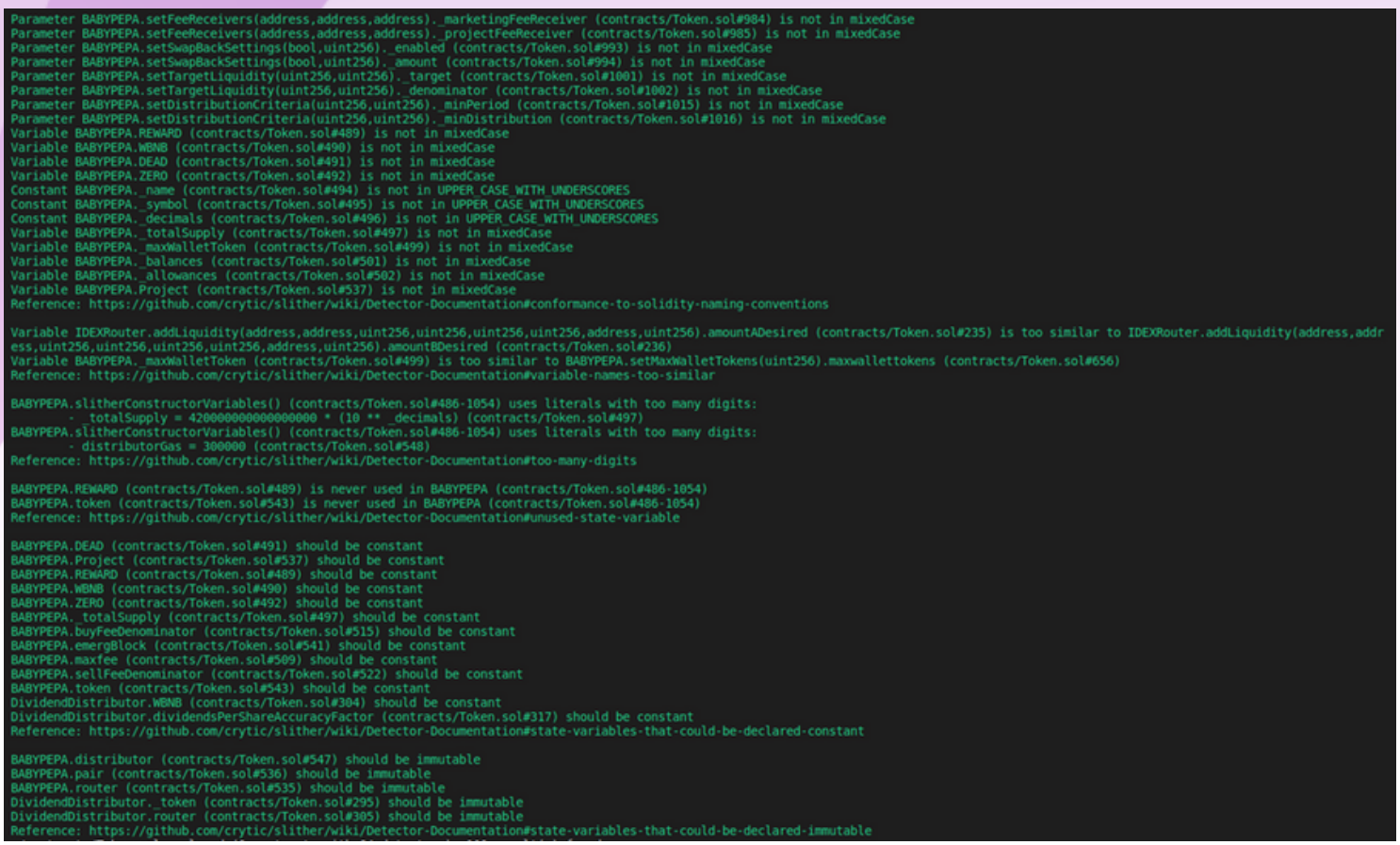
CONTRACT ASSESMENT

└	burning	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	setMaxWalletTokens	External ❗	🛑	onlyOwner
└	transferFrom	External ❗	🛑	NO ❗
└	_transferFrom	Internal 🔒	🛑	
└	_basicTransfer	Internal 🔒	🛑	
└	shouldTakeFee	Internal 🔒		
└	shouldTakeFeer	Internal 🔒		
└	getTotalFee	Public ❗		NO ❗
└	getMultipliedFee	Public ❗		NO ❗
└	takeFee	Internal 🔒	🛑	
└	shouldSwapBack	Internal 🔒		
└	swapBack	Internal 🔒	🛑	swapping
└	buyTokens	Internal 🔒	🛑	swapping
└	launched	Internal 🔒		
└	launch	Public ❗	🛑	onlyOwner
└	setIsDividendExempt	External ❗	🛑	onlyOwner
└	setIsFeeExempt	External ❗	🛑	onlyOwner
└	setIsMaxWalletExempt	External ❗	🛑	onlyOwner
└	setBuyFees	External ❗	🛑	onlyOwner
└	setSellFees	External ❗	🛑	onlyOwner
└	setFeeReceivers	External ❗	🛑	onlyOwner
└	setSwapBackSettings	External ❗	🛑	onlyOwner
└	setTargetLiquidity	External ❗	🛑	onlyOwner
└	manualSend	External ❗	🛑	NO ❗
└	setDistributionCriteria	External ❗	🛑	onlyOwner
└	claimDividend	External ❗	🛑	NO ❗
└	getUnpaidEarnings	Public ❗		NO ❗
└	setDistributorSettings	External ❗	🛑	onlyOwner
└	getCirculatingSupply	Public ❗		NO ❗
└	getLiquidityBacking	Public ❗		NO ❗
└	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._maxWalletToken (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).amountADesired (contracts/Token.sol#235) is too similar to IDEXRouter.addLiquidity(address,addr
ess,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#236)
Variable BABYPEPA._maxWalletToken (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 Resovled

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 Resovled

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 Resovled

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 faile unless ownership is transferred to the router. Although he launch function can be called manually but the downside of this issue is that owner has to fisrt enable tradings 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-liquidty generated tokens are sent to an EOA (externally owned account)

Severity: Medium
Function: swapBack
Lines: 864
Status: No Resovled

Pool shares generated from auto-liquidity are sent to an EOA, over time this pool share tokens will get accumulated and cane 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 distrbute 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**, **shouldTakeFeer**
- 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 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