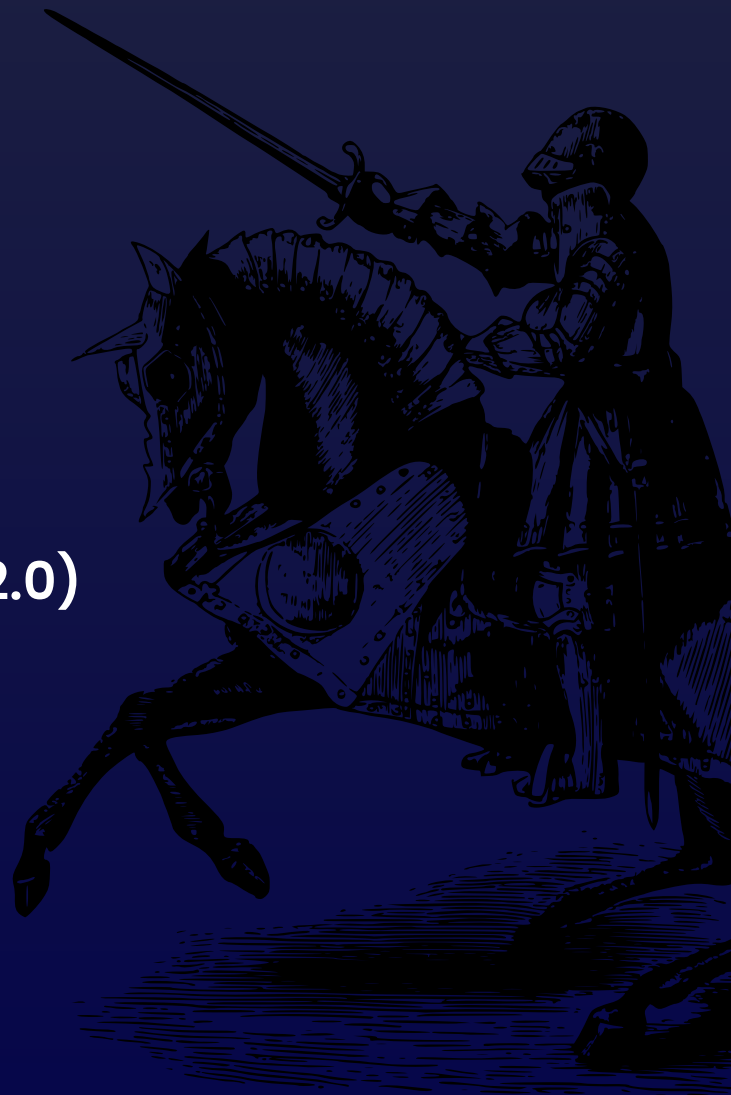


AUDIT REPORT

SecureWise

BABY PEPE AI 2.0 (BABYPEPEAI2.0)



Quick Result

Quick Result	Status
Owner can mint ?	Not Detected
Owner can update tax over 25% ?	Not Detected
Owner can pause trade ?	Not Detected
Owner can enable trading ?	Not Detected
Owner can add Blacklist ?	Not Detected
Owner can set Max Tx ?	Not Detected
Owner can set Max Wallet Amount ?	Not Detected
KYC ?	Not Done

Page 6, 10 for more details

Baby Pepe Ai 2.0 (BabypepeAi2.0) as **PASSED** the smart contract audit with **Critical Issues**.

Findings

Risk Classification	Description
High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, of the contract and its functions. Must be fixed as soon as possible.
Medium	Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Must be fixed as soon as possible.
Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
Informational	A vulnerability that have informational character but is not effecting any of the code

Severity	Found	Pending	Resolved
High	0	4	0
Medium	0	1	0
Low	0	2	0
Informational	0	1	0
Total	0	8	0

Contents

01	Quick Result
02	Findings
04	Overview
05	Auditing Approach and Methodologies
06	Findings Summary
07	Function Privileges
08	Inheritance Graph
10	Manual Review
18	Disclaimer

Overview

Token Name: Baby Pepe Ai 2.0 (**BabypepeAI2.0**)

Language: Solidity

Contract Address: 0x4e55a24a7623222811880305D4C8B1b25FBFC613

Network: Binance Smart Chain

Supply: 420690000000

KYC: Not done

Website: <https://babypepeai.pro/>

Twitter: <http://twitter.com/babypepeai2>

Telegram: <https://t.me/BABYPEPEAI2>

Report Date: June 30, 2023

Testnet:

<https://testnet.bscscan.com/address/0x4d4a7c3e909de178ff426bf15a0361f30e922d2e>

Auditing Approach and Methodologies

SecureWise has performed starting with analyzing the code, issues, code quality, and libraries. Reviewed line-by-line by our team. Finding any potential issue like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

Methodology

- Understanding the size, scope and functionality of your project's source code
- Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Testing and automated analysis of the Smart Contract to determine proper logic has been followed throughout the whole process
- Deploying the code on testnet using multiple live test
- Analyzing a program to determine the specific input that causes different parts of a program to execute its functions.
- Checking whether all the libraries used in the code are on the latest version.

Goals

Smart Contract System is secure, resilient and working according to the specifications and without any vulnerabilities.

Risk Classification

High: Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, of the contract and its functions. Must be fixed as soon as possible.

Medium: Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Must be fixed as soon as possible.









Low: Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.

Informational: A vulnerability that have informational character but is not effecting any of the code

Findings Summary

SecureWise has applied the automated and manual analysis of Smart Contract and were reviewed for common contract vulnerabilities and centralized exploits

Findings

	User can be charged fees up to 99% indirect when trading at first block
	Owner has the authority to change the target liquidity and denominator values without limit
	Owner has the authority to update swap back settings and can set swap threshold "0"
	Owner has the authority to set distribution criteria and can set min period and distribution "0"
	Auto liquidity is transferred to an externally owned account.
	Accounts can be excluded by the owner from receiving rewards.
	The current fee percentage is fixed 6% and cannot be changed.
	Owner has the authority to exclude account from fees

Page 10 for more details

Baby Pepe Ai 2.0 (BabypepeAi2.0) as **PASSED the smart contract audit with **Critical Issues**.**

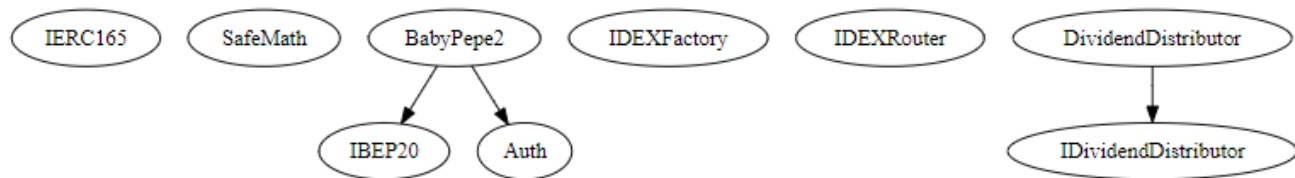
Function Privileges

```

| **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 | 🔒 | ● | |
| |||||
| **BabyPepe2** | 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 | |
| | setIsFeeExempt | External | ! | ● | onlyOwner |
| | burn | External | ! | ● | NO | |
| | transfer | External | ! | ● | NO | |
| | 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 | 🔒 | | |
| | setIsDividendExempt | 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 | |

```

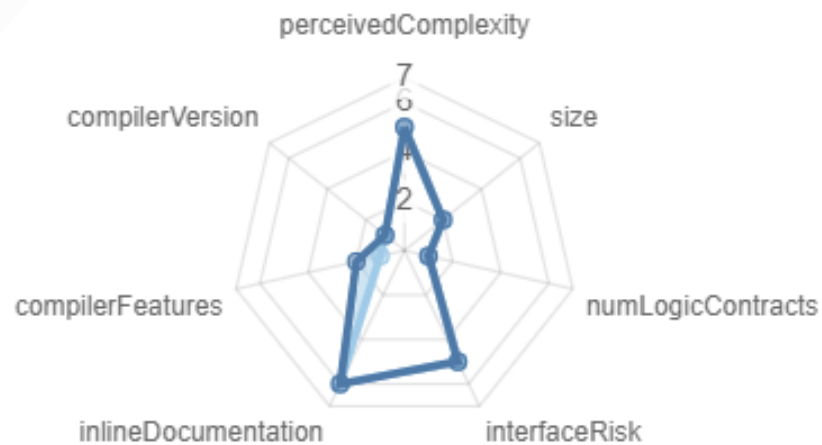

Inheritance Graph



Source Lines



Risk



Manual Review

High Risk

User can be charged fees up to 99% indirect when trading at first block

```
function getTotalFee(bool selling) public view returns (uint256) {
    uint256 feeDenominator = selling
        ? sellFeeDenominator
        : buyFeeDenominator;
    uint256 totalFee = selling ? totalSellFee : totalBuyFee;
    if (launchedAt + 1 >= block.number) {
        return feeDenominator.sub(1);
    }
    if (selling) {
        return getMultipliedFee();
    }
    return totalFee;
}
```

Description

If the contract is being executed for the first time (within the same block it was launched), the **getTotalFee** function would return **feeDenominator - 1** as the total fee. If the feeDenominator is set to 100, for example, the total fee applied in the first block would be 99% (100 - 1). This means that a user could potentially be charged fees up to 99% of their transaction amount during the initial launch phase of the contract.

Recommendation

Charging such high fees can be considered unreasonable or even exploitative. It is important to ensure that the fee structure is fair, transparent, and aligns with the expectations of the users.

Manual Review

High Risk

Owner has the authority to change the target liquidity and denominator values without limit

```
function setTargetLiquidity(uint256 _target, uint256 _denominator)
    external
    onlyOwner
{
    targetLiquidity = _target;
    targetLiquidityDenominator = _denominator;
}
```

```
uint256 dynamicLiquidityFee = isOverLiquified(
    targetLiquidity,
    targetLiquidityDenominator
)
? 0
: liquidityFee;
uint256 amountToLiquify = balanceOf(address(this))
    .mul(dynamicLiquidityFee)
    .div(totalFee)
    .div(2);
```

Description

The contract is using variables that may be set to zero as denominators. As a result, the transactions will revert.

Recommendation

The contract should prevent those variables to be set to zero or should not allow to execute the corresponding statements.

Manual Review

High Risk

Owner has the authority to update swap back settings and can set swap threshold "0"

```
function setSwapBackSettings(bool _enabled, uint256 _amount)
    external
    onlyOwner
{
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

Description

Setting swapThreshold to 0 can disable sell and transfers. This is because internal swap is performed even in case of swapThreshold being equal to 0.

Recommendation

Mitigate this issue, ensure that swapThreshold is always greater than a 0

Manual Review

High Risk

Owner has the authority to set distribution criteria and can set min period and distribution "0"

```
function setDistributionCriteria(  
    uint256 _minPeriod,  
    uint256 _minDistribution  
) external override onlyToken {  
    minPeriod = _minPeriod;  
    minDistribution = _minDistribution;  
}
```

Description

If the owner sets arbitrary limits for the minimum period and minimum distribution, it can introduce risks depending on the specific functionality of the contract. For example, if the minimum period is set too high or the minimum distribution is set too low, it could affect the intended distribution mechanism. This could potentially lead to unfair or inefficient distribution of tokens.

Recommendation

Mitigate these risks and vulnerabilities, it is advisable to implement proper validation checks on the input values

Manual Review

Medium Risk

Auto liquidity is transferred to an externally owned account.

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

Description

Owner of the contract received LP tokens generated from autoliquidity, this LP tokens can be used to remove a portion of liquidity pool.

Recommendation

Mitigate this risk burn or lock new LP tokens.

Manual Review

Low Risk

Accounts can be excluded by the owner from receiving rewards.

```
function setIsDividendExempt(address holder, bool exempt)
    external
    onlyOwner
{
    require(holder != address(this) && holder != pair);
    isDividendExempt[holder] = exempt;
    if (exempt) {
        distributor.setShare(holder, 0);
    } else {
        distributor.setShare(holder, _balances[holder]);
    }
}
```

Description

setIsDividendExempt function, designed to exclude a specified account from receiving dividends.

Recommendation

Implementing error-handling mechanisms to gracefully manage potential exceptions is also recommended. Ensure that appropriate access control mechanisms are in place to restrict the `excludeFromDividends` function to only be called by the contract owner

Manual Review

Low Risk

The current fees percentage is fixed 6% and cannot be changed.

```
uint256 liquidityBuyFee = 0;  
uint256 marketingBuyFee = 400;  
uint256 projectBuyFee = 0;  
uint256 totalBuyFee = 600;  
uint256 buyFeeDenominator = 10000;  
uint256 buyfeeburn = 200;  
  
uint256 liquiditySellFee = 0;  
uint256 marketingSellFee = 400;  
uint256 projectSellFee = 0;  
uint256 totalSellFee = 600;  
uint256 sellFeeDenominator = 10000;  
uint256 sellfeeburn = 200;
```

Description

Total Buy fee 6% Total sell fee 6% and it can not be change by the owner

Recommendation

No specific recommendation is necessary for the taxes management at this time. Taxes are reasonable limit

Manual Review

Informational

Owner has the authority to exclude account from fees

```
function setIsFeeExempt(address holder, bool exempt) external onlyOwner {  
    isFeeExempt[holder] = exempt;  
}
```

Description

setIsFeeExempt allows the contract owner to modify the exclusion status of an account from fees by updating the `isFeeExempt` mapping.

Recommendation

No specific recommendation is necessary for the **setIsFeeExempt** function at this time. However, it is important to ensure that the function is being used appropriately and that the owner's ability to exclude or include accounts from fees is clearly documented and understood.

Disclaimer

SecureWise provides the smart contract audit of solidity. Audit and report are for informational purposes only and not, nor should be considered, as an endorsement to engage with, invest in, participate, provide an incentive, or disapprove, criticise, discourage, or purport to provide an opinion on any particular project or team.

This audit report doesn't provide any warranty or guarantee regarding the nature of the technology analysed. These reports, in no way, provide investment advice, nor should be used as investment advice of any sort. Investors must always do their own research and manage their risk.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and SecureWise and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) SecureWise owe no duty of care towards you or any other person, nor does SecureWise make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SecureWise hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SecureWise hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SecureWise, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.



AUDIT REPORT

SecureWise



securewise.org



t.me/securewisehub



twitter.com/securewiseAudit

