AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

NotBabyCoin

DATED : 13 March, 2024

AUDIT SUMMARY

Project name – NotBabyCoin

Date: 13 March, 2024

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	0	0	1	2
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

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

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

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

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/token/0x42bbac51a34bcd02dbb9c4e4e1b19874fc10b134#code

Token Information

Token Name : NotBabyCoin

Token Symbol: NOTB

Decimals: 18

Token Supply: 200000000

Network: BscScan

Token Type: BEP-20

Token Address:
0x7793142B62e2435d0100870710D4B5e2E4468B0e

Checksum:
99fe9e9c341ee7973b99e3c799ce8f6f

Owner:
0xd032c6102574f854ab000146AE37a5E995506284
(at time of writing the audit)

Deployer:
0xd032c6102574f854ab000146AE37a5E995506284

TOKEN OVERVIEW

Fees:

Buy Fee: 1-25%

Sell Fee: 1-25%

Transfer Fee: 3-25%

Fees Privilege: Owner

Ownership: Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

AUDIT METHODOLOGY

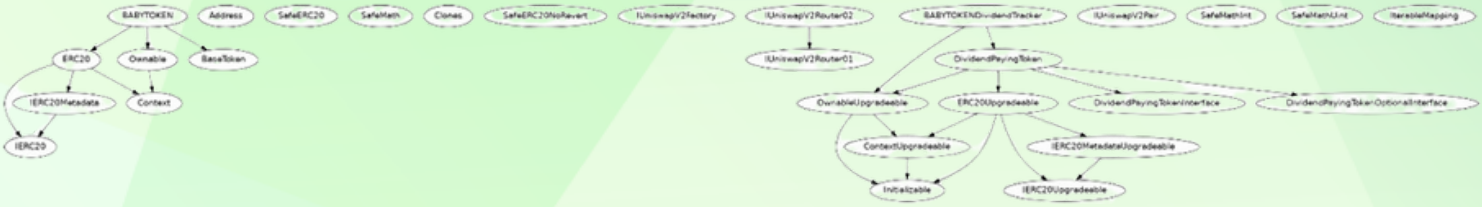
The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

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

INHERITANCE TREE



STATIC ANALYSIS

A static analysis of the code was performed using Slither.

No issues were found.

```
INFO:Detectors:
DividendPayingToken.__DividendPayingToken_init(address,string,string)._name (BABYTOKEN.sol#2462) shadows:
        - ERC20Upgradeable._name (BABYTOKEN.sol#1727) (state variable)
DividendPayingToken.__DividendPayingToken_init(address,string,string)._symbol (BABYTOKEN.sol#2463) shadows:
        - ERC20Upgradeable._symbol (BABYTOKEN.sol#1728) (state variable)
DividendPayingToken.dividendOf(address)._owner (BABYTOKEN.sol#2523) shadows:
        - OwnableUpgradeable._owner (BABYTOKEN.sol#2071) (state variable)
DividendPayingToken.withdrawableDividendOf(address)._owner (BABYTOKEN.sol#2530) shadows:
        - OwnableUpgradeable._owner (BABYTOKEN.sol#2071) (state variable)
DividendPayingToken.withdrawnDividendOf(address)._owner (BABYTOKEN.sol#2542) shadows:
        - OwnableUpgradeable._owner (BABYTOKEN.sol#2071) (state variable)
DividendPayingToken.accumulativeDividendOf(address)._owner (BABYTOKEN.sol#2556) shadows:
        - OwnableUpgradeable._owner (BABYTOKEN.sol#2071) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
BABYTOKEN.setSwapTokensAtAmount(uint256) (BABYTOKEN.sol#3073-3079) should emit an event for:
        - swapTokensAtAmount = amount (BABYTOKEN.sol#3078)
BABYTOKEN.setTokenRewardsFee(uint256) (BABYTOKEN.sol#3110-3114) should emit an event for:
        - totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee) (BABYTOKEN.sol#3112)
BABYTOKEN.setLiquiditFee(uint256) (BABYTOKEN.sol#3116-3120) should emit an event for:
        - liquidityFee = value (BABYTOKEN.sol#3117)
        - totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee) (BABYTOKEN.sol#3118)
BABYTOKEN.setMarketingFee(uint256) (BABYTOKEN.sol#3122-3126) should emit an event for:
        - marketingFee = value (BABYTOKEN.sol#3123)
        - totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee) (BABYTOKEN.sol#3124)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
BABYTOKEN.constructor(string,string,uint256,address[4],uint256[3],uint256,address,uint256)._uniswapV2Pair (BABYTOKEN.sol#3044-3045) lacks a zero-check on :
              - uniswapV2Pair = _uniswapV2Pair (BABYTOKEN.sol#3047)
BABYTOKEN.constructor(string,string,uint256,address[4],uint256[3],uint256,address,uint256).serviceFeeReceiver_ (BABYTOKEN.sol#3010) lacks a zero-check on :
              - address(serviceFeeReceiver_).transfer(serviceFee_) (BABYTOKEN.sol#3068)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
SafeERC20NoRevert.safeTransfer(IERC20,address,uint256) (BABYTOKEN.sol#1226-1238) has external calls inside a loop: (success,returndata) = address(token).cal
l(abi.encodeWithSelector(token.transfer.selector,to,value)) (BABYTOKEN.sol#1231-1233)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
```

```
INFO:Detectors:
BABYTOKENDividendTracker.getAccount(address) (BABYTOKEN.sol#2727-2774) uses timestamp for comparisons
        Dangerous comparisons:
        - nextClaimTime > block.timestamp (BABYTOKEN.sol#2771-2773)
BABYTOKENDividendTracker.canAutoClaim(uint256) (BABYTOKEN.sol#2799-2805) uses timestamp for comparisons
        Dangerous comparisons:
        - lastClaimTime > block.timestamp (BABYTOKEN.sol#2800)
        - block.timestamp.sub(lastClaimTime) >= claimWait (BABYTOKEN.sol#2804)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (BABYTOKEN.sol#530-540) uses assembly
        - INLINE ASM (BABYTOKEN.sol#536-538)
Address.verifyCallResult(bool,bytes,string) (BABYTOKEN.sol#699-719) uses assembly
        - INLINE ASM (BABYTOKEN.sol#711-714)
Clones.clone(address) (BABYTOKEN.sol#1151-1160) uses assembly
        - INLINE ASM (BABYTOKEN.sol#1152-1158)
Clones.cloneDeterministic(address,bytes32) (BABYTOKEN.sol#1169-1178) uses assembly
        - INLINE ASM (BABYTOKEN.sol#1170-1176)
Clones.predictDeterministicAddress(address,bytes32,address) (BABYTOKEN.sol#1183-1198) uses assembly
        - INLINE ASM (BABYTOKEN.sol#1188-1197)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
BABYTOKEN._transfer(address,address,uint256) (BABYTOKEN.sol#3273-3361) has a high cyclomatic complexity (14).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
Address.functionCall(address,bytes) (BABYTOKEN.sol#583-585) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (BABYTOKEN.sol#612-618) is never used and should be removed
Address.functionDelegateCall(address,bytes) (BABYTOKEN.sol#672-674) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (BABYTOKEN.sol#682-691) is never used and should be removed
Address.functionStaticCall(address,bytes) (BABYTOKEN.sol#645-647) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (BABYTOKEN.sol#655-664) is never used and should be removed
Address.sendValue(address,uint256) (BABYTOKEN.sol#558-563) is never used and should be removed
Clones.cloneDeterministic(address,bytes32) (BABYTOKEN.sol#1169-1178) is never used and should be removed
Clones.predictDeterministicAddress(address,bytes32) (BABYTOKEN.sol#1203-1209) is never used and should be removed
Clones.predictDeterministicAddress(address,bytes32,address) (BABYTOKEN.sol#1183-1198) is never used and should be removed
Context._msgData() (BABYTOKEN.sol#140-142) is never used and should be removed
ContextUpgradeable.__Context_init() (BABYTOKEN.sol#1668-1670) is never used and should be removed
ContextUpgradeable._msgData() (BABYTOKEN.sol#1678-1680) is never used and should be removed
DividendPayingToken._transfer(address,address,uint256) (BABYTOKEN.sol#2575-2590) is never used and should be removed
```

STATIC ANALYSIS

A static analysis of the code was performed using Slither.

No issues were found.

```
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (BABYTOKEN.sol#1289) is too similar
 to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (BABYTOKEN.sol#1290)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (BABYTOKEN.sol#2495) is too similar to BABYTOKENDividendTracker.getAccou
nt(address).withdrawableDividends (BABYTOKEN.sol#2734)
Variable DividendPayingToken.__DividendPayingToken_init(address,string,string)._rewardToken (BABYTOKEN.sol#2461) is too similar to BABYTOKENDividendTracker.
initialize(address,uint256).rewardToken_ (BABYTOKEN.sol#2654)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Clones.clone(address) (BABYTOKEN.sol#1151-1160) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d7300000000000000000000000000000000) (BABYTOKEN.sol#1154)
Clones.clone(address) (BABYTOKEN.sol#1151-1160) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3000000000000000000000000000000000000) (BABYTOKEN.sol#1156)
Clones.cloneDeterministic(address,bytes32) (BABYTOKEN.sol#1169-1178) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d7300000000000000000000000000000000) (BABYTOKEN.sol#1172)
Clones.cloneDeterministic(address,bytes32) (BABYTOKEN.sol#1169-1178) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3000000000000000000000000000000000000) (BABYTOKEN.sol#1174)
Clones.predictDeterministicAddress(address,bytes32,address) (BABYTOKEN.sol#1183-1198) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d7300000000000000000000000000000000) (BABYTOKEN.sol#1190)
Clones.predictDeterministicAddress(address,bytes32,address) (BABYTOKEN.sol#1183-1198) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3ff0000000000000000000000000000000000) (BABYTOKEN.sol#1192)
BABYTOKEN.constructor(string,string,uint256,address[4],uint256[3],uint256,address,uint256) (BABYTOKEN.sol#3003-3069) uses literals with too many digits:
        - gasForProcessing = 300000 (BABYTOKEN.sol#3032)
BABYTOKEN.updateGasForProcessing(uint256) (BABYTOKEN.sol#3142-3153) uses literals with too many digits:
        - require(bool,string)(newValue >= 200000 && newValue <= 500000,BABYTOKEN: gasForProcessing must be between 200,000 and 500,000) (BABYTOKEN.sol#3143-3146)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
SafeMathInt.MAX_INT256 (BABYTOKEN.sol#2196) is never used in SafeMathInt (BABYTOKEN.sol#2194-2251)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Detectors:
BABYTOKEN.dividendTracker (BABYTOKEN.sol#2953) should be immutable
BABYTOKEN.rewardToken (BABYTOKEN.sol#2955) should be immutable
BABYTOKEN.uniswapV2Pair (BABYTOKEN.sol#2949) should be immutable
BABYTOKEN.uniswapV2Router (BABYTOKEN.sol#2948) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:BABYTOKEN.sol analyzed (29 contracts with 93 detectors), 122 result(s) found
```

FUNCTIONAL TESTING

1- Approve (passed):
https://testnet.bscscan.com/tx/0xe24f15a93e0edf3f21b97ff0001576b45a02b19ce96e9a0338e2d535a3807294

2- Increase Allowance (passed):
https://testnet.bscscan.com/tx/0x9d3de17d8158f41e625f7f75b9da024dea20fc3f02c049736702125c451f7fdf

3- Decrease Allowance (passed):
https://testnet.bscscan.com/tx/0xfd74116563f7892c030603502ac9b64443ccb6846e448db32f72497da5bb59fd

4- Exclude From Fees (passed):
https://testnet.bscscan.com/tx/0x16c74d16dc4231d0de6d706c997f775902e488b73024ccfa56f42335365415ba

5- Exclude From Dividends (passed):
https://testnet.bscscan.com/tx/0x3ac146d7ead920e2e5108c750ea2fb7b4e9933aece2f072f76709a333ce99c03

6- Set Liquidity Fee (passed):
https://testnet.bscscan.com/tx/0x1f4d0e719bd514b80e48d646ace8095e242dbf4fdb5093a5432858fd281aca95

7- Set Marketing Fee (passed):
https://testnet.bscscan.com/tx/0x20347ad9caadb1d4546b792c4ebb2da9fdb8631c13651424fddcf49f3e041082

FUNCTIONAL TESTING

8- Set Token Reward Fee (passed):

https://testnet.bscscan.com/tx/0x5862df28e3b50389a92fd5ce1f9b25bd630fb80cb2e38295482ed94e75df96c8

9- Set Marketing Wallet (passed):

https://testnet.bscscan.com/tx/0x2e31b66ffc594fa0c658ff1ba10acd77b01aff22a85a26ee50b488250cc115e9

POINTS TO NOTE

- The owner can transfer ownership.
- The owner can renounce ownership.
- The owner can set the marketing wallet address.
- The owner can set Reward/Liquidity/Marketing fees of not more than 25%.
- The owner can exclude address from Dividends.
- The Owner can exclude address from fees.
- The Owner can update claim wait.
- The owner can set balance.

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	0
Medium-Risk	0
Low-Risk	1
Gas Optimization / Suggestions	2

MANUAL TESTING

Centralization – Missing Events

Severity: Low

Subject: Missing Events

Status: Open

Overview:

They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function setSwapTokensAtAmount(uint256 amount) external onlyOwner {
        require(
            amount > totalSupply() / 10 ** 5,
            "BABYTOKEN: Amount must be greater than 0.001% of total supply"
        );
        swapTokensAtAmount = amount;
    }
function setLiquiditFee(uint256 value) external onlyOwner {
        liquidityFee = value;
        totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
        require(totalFees <= 25, "Total fee is over 25%");
    }
function setMarketingFee(uint256 value) external onlyOwner {
        marketingFee = value;
        totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
        require(totalFees <= 25, "Total fee is over 25%");
    }
function setTokenRewardsFee(uint256 value) external onlyOwner {
        tokenRewardsFee = value;
        totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
        require(totalFees <= 25, "Total fee is over 25%");
    }
function setMarketingWallet(address payable wallet) external onlyOwner {
        require(
            wallet != address(0),
            "BABYTOKEN: The marketing wallet cannot be the value of zero"
        );
        require(!wallet.isContract(), "Marketing wallet cannot be a contract");
        _marketingWalletAddress = wallet;
    }
```

Suggestion:

Emit an event for critical changes.

MANUAL TESTING

Optimization

Severity: Informational

Subject: Remove Safe Math

Status: Open

Line: 913-112

Overview:

compiler version above 0.8.0 can control arithmetic overflow/underflow, it is recommended to remove the unwanted code to avoid high gas fees.

MANUAL TESTING

Optimization

Severity: Optimization

Subject: Remove unused code

Status: Open

Overview:

Unused variables are allowed in Solidity, and they do not pose a direct security issue. It is the best practice to avoid them.

```
  function _msgData() internal view virtual returns (bytes calldata) {
        return msg.data;
    }
    uint256[50] private __gap;
}
```

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