auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

BabyTrump

DATED : 03 June 24'

MANUAL TESTING

Centralization – Buy and Sell Fees.

Severity: High

Function: setBuy, setSell

Status: Open

Overview:

The owner can set the buy and sell fees up to 100%, which is not recommended.

```
function setBuy(uint256 newFundFee, uint256 newLpFee) public
onlyOwner{
    _buyFundFee = newFundFee;
    _buyLPFee = newLpFee;
  }
function setSell(uint256 newFundFee, uint256 newLpFee) public
onlyOwner{
    _sellFundFee = newFundFee;
    _sellLPFee = newLpFee;
  }
```

Suggestion:

It is recommended that no fees in the contract should be more than 25% of the contract.

MANUAL TESTING

Centralization – Missing Require Check.

Severity: High

Function: setFundAddress

Status: Open

Overview:

The owner can set any arbitrary address excluding zero address as this is not recommended because if the owner sets the address to the contract address, then the ETH will not be sent to that address. The transaction will fail, leading to a potential honeypot in the contract.

```
f function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _isExcludeFromFee[addr] = true;
  }
```

Suggestion:

It is recommended that the address should not be able to be set as a contract address.

AUDIT SUMMARY

Project name – BabyTrump

Date: 03 June, 2024

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

Audit Status: **HIGH RISK MAJOR FLAG**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	2	0	4	0
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/address/0x20dfbd9397b1718f4fa5d255aae6da1cf62368be#code

Token Information

Token Address:
0x21eF5cFA8c5E67B8AE216992676448BE04b49223

Name: BabyTrump

Symbol: BabyTrump

Decimals: 9

Network: BscScan

Token Type: BEP-20

Owner: 0xC1Dc381575b0eaeD97aD535AEfa1F0a8DfaDEc21

Deployer: 0xC1Dc381575b0eaeD97aD535AEfa1F0a8DfaDEc21

Token Supply: 10000000000

Checksum: A17acbefe2a12642d388659dffd20221

Testnet:
https://testnet.bscscan.com/address/0x20dfbd9397b1718f4fa5d255aae6da1cf62368be#code

TOKEN OVERVIEW

Buy Fee: 30-100%

Sell Fee: 30-100%

Transfer Fee: 0%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: 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

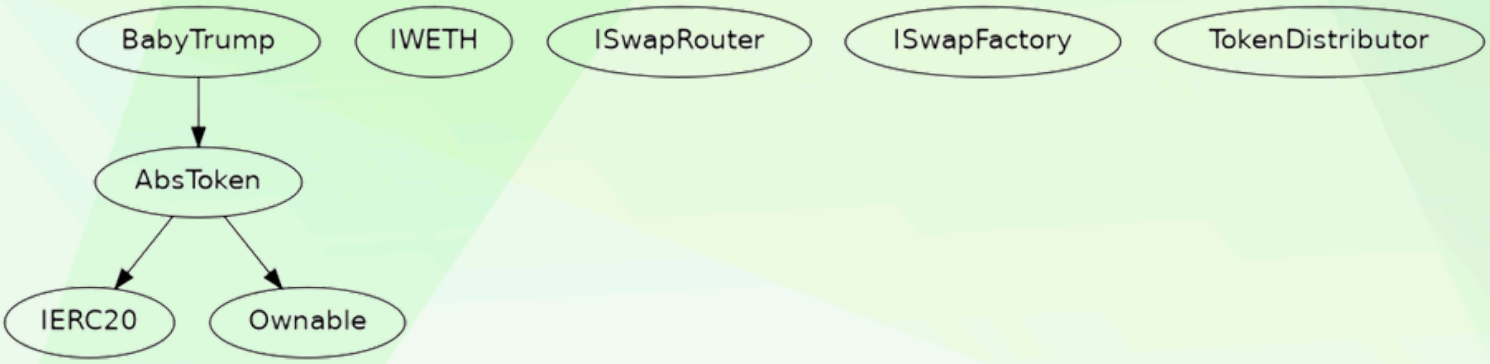
CLASSIFICATION OF RISK

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	0
Low-Risk	4
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- The owner can transfer ownership.
- The owner can renounce ownership.
- The owner can set a buy/sell fee to more than 100%.
- The owner can set a new airdrop number.
- The owner can set a new airdrop enable.
- The owner can set a fund address.
- The owner can claim tokens.
- The owner can whitelist multiple addresses.

STATIC ANALYSIS

```
INFO:Detectors:
AbsToken._transfer(address,address,uint256).takeFee (BabyTrump.sol#250) is a local variable never initialized
AbsToken._transfer(address,address,uint256).isSell (BabyTrump.sol#251) is a local variable never initialized
AbsToken._tokenTransfer(address,address,uint256,bool,bool).feeAmount (BabyTrump.sol#295) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
TokenDistributor.constructor(address) (BabyTrump.sol#93-95) ignores return value by IERC20(token).approve(msg.sender,uint256(- uint256(0))) (BabyTrump.sol#94)
AbsToken.constructor(address,string,string,uint8,uint256,address) (BabyTrump.sol#135-173) ignores return value by IERC20(WBNBAddress).approve(address(swapRouter),MAX) (BabyTrump.sol#146)
AbsToken.swapTokenForFund(uint256,uint256) (BabyTrump.sol#322-360) ignores return value by _swapRouter.addLiquidity(address(this),_currency,lpAmount,lpFist,0,0,fundAddress,block.timestamp) (BabyTrump.sol#355-3
57)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
AbsToken.allowance(address,address).owner (BabyTrump.sol#200) shadows:
        - Ownable.owner() (BabyTrump.sol#71-73) (function)
AbsToken._approve(address,address,uint256).owner (BabyTrump.sol#227) shadows:
        - Ownable.owner() (BabyTrump.sol#71-73) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
AbsToken.setFundAddress(address).addr (BabyTrump.sol#382) lacks a zero-check on :
        - fundAddress = addr (BabyTrump.sol#383)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in AbsToken.transferFrom(address,address,uint256) (BabyTrump.sol#209-215):
        External calls:
        - _transfer(sender,recipient,amount) (BabyTrump.sol#210)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,address(_tokenDistributor),block.timestamp) (BabyTrump.sol#330-336)
                - FIST.transferFrom(address(_tokenDistributor),address(this),fundAmount) (BabyTrump.sol#344)
                - IWETH(_currency).withdraw(fundAmount) (BabyTrump.sol#345)
                - FIST.transferFrom(address(_tokenDistributor),fundAddress,fundAmount) (BabyTrump.sol#348)
                - FIST.transferFrom(address(_tokenDistributor),address(this),fistBalance - fundAmount) (BabyTrump.sol#350)
                - _swapRouter.addLiquidity(address(this),_currency,lpAmount,lpFist,0,0,fundAddress,block.timestamp) (BabyTrump.sol#355-357)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (BabyTrump.sol#210)
                - recipient.transfer(amount) (BabyTrump.sol#363)
        State variables written after the call(s):
        - _allowances[sender][msg.sender] = _allowances[sender][msg.sender] - amount (BabyTrump.sol#212)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in AbsToken._transfer(address,address,uint256) (BabyTrump.sol#242-285):
        External calls:
        - swapTokenForFund(numTokensSellToFund,swapFee) (BabyTrump.sol#274)
                - _swapRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(tokenAmount - lpAmount,0,path,address(_tokenDistributor),block.timestamp) (BabyTrump.sol#330-336)
                - FIST.transferFrom(address(_tokenDistributor),address(this),fundAmount) (BabyTrump.sol#344)
                - IWETH(_currency).withdraw(fundAmount) (BabyTrump.sol#345)
                - FIST.transferFrom(address(_tokenDistributor),fundAddress,fundAmount) (BabyTrump.sol#348)
                - FIST.transferFrom(address(_tokenDistributor),address(this),fistBalance - fundAmount) (BabyTrump.sol#350)
                - _swapRouter.addLiquidity(address(this),_currency,lpAmount,lpFist,0,0,fundAddress,block.timestamp) (BabyTrump.sol#355-357)
        External calls sending eth:
```

```
INFO:Detectors:
Variable ISwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (BabyTrump.sol#45) is too similar to ISwapRouter.addLiquidity(address,address,uint256,uint256,u
int256,uint256,address,uint256).amount0Desired (BabyTrump.sol#46)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
BabyTrump.constructor() (BabyTrump.sol#423-431) uses literals with too many digits:
        - AbsToken(address(0xD99D1c33F9fC3444f8101754aBC46c52416550D1),BabyTrump,BabyTrump,9,10000000000,address(0x969c171CC20d04Cd8F089EB406Df9Ee94bD7A933)) (BabyTrump.sol#423-430)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
AbsToken._currency (BabyTrump.sol#113) should be immutable
AbsToken._mainPair (BabyTrump.sol#127) should be immutable
AbsToken._swapRouter (BabyTrump.sol#112) should be immutable
AbsToken._tokenDistributor (BabyTrump.sol#118) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:BabyTrump.sol analyzed (8 contracts with 93 detectors), 46 result(s) found
```

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- Set Buy (passed):

https://testnet.bscscan.com/tx/0x5fa2d3f80b2b6256a7d5062c5b163a9aab7f26c70062e0df153358c08254d15a

2- Set Sell (passed):

https://testnet.bscscan.com/tx/0xabb476eec5282829820d3f620f39dc7775b3859057236b853f4fb5cd6c23fcd0

3- Set Fund Address (passed):

https://testnet.bscscan.com/tx/0xf982ae8a8756986057092784533 5f71e3c5a13262fa1273be21545b2561100ad

4- Approve (passed):

https://testnet.bscscan.com/tx/0x08055e4b99dd245c21670aa4ec5f67d7d8c624938f93a0f95469ae5e62863881

5- Multi W Ls (passed):

https://testnet.bscscan.com/tx/0x73be9307d1aa481b0c32676573ff60eda16f5b7472c74abec17f8ca00cca72f8

6- Set W Ls (passed):

https://testnet.bscscan.com/tx/0x92b75355d6a258c8c538864494e4ad226acb8da50a69af29481347a2685143d3

7- Launch (passed):

https://testnet.bscscan.com/tx/0x73276a53c3e4f7ebb325680baad125490befa5972456cf0f36615b8994494265

MANUAL TESTING

Centralization – Buy and Sell Fees.

Severity: High

Function: setBuy, setSell

Status: Open

Overview:

The owner can set the buy and sell fees up to 100%, which is not recommended.

```
function setBuy(uint256 newFundFee, uint256 newLpFee) public
onlyOwner{
    _buyFundFee = newFundFee;
    _buyLPFee = newLpFee;
  }
function setSell(uint256 newFundFee, uint256 newLpFee) public
onlyOwner{
    _sellFundFee = newFundFee;
    _sellLPFee = newLpFee;
  }
```

Suggestion:

It is recommended that no fees in the contract should be more than 25% of the contract.

MANUAL TESTING

Centralization – Missing Require Check.

Severity: High

Function: setFundAddress

Status: Open

Overview:

The owner can set any arbitrary address excluding zero address as this is not recommended because if the owner sets the address to the contract address, then the ETH will not be sent to that address. The transaction will fail, leading to a potential honeypot in the contract.

```
f function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _isExcludeFromFee[addr] = true;
  }
```

Suggestion:

It is recommended that the address should not be able to be set as a contract address.

MANUAL TESTING

Centralization – Unsafe Usage of tx.origin

Severity: Low

Subject: Tx.origin

Status: Open

Overview:

Avoid using TX.origin for authorization, another contract can have a method that will call your contract (where the user has some funds for instance) and your contract will authorize that transaction as your address is in tx. origin.

```
 if (trasnferDelayEnabled) {
if (
          to != owner() &&
          to != address(dexRouter) &&
          to != address(dexPair)
        ) {
require(
           _holderLastTransferTimestamp[tx.origin] <
            block.number,
"_transfer:: Transfer Delay enabled. Only one purchase per block
allowed."
          );
          _holderLastTransferTimestamp[tx.origin] = block.number;
        }
      }
```

Suggestion:

You should use msg. sender for authorization (if another contract calls your contract msg.sender will be the address of the contract and not the address of the user who called the contract.

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 setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _isExcludeFromFee[addr] = true;
  }
function setBuy(uint256 newFundFee, uint256 newLpFee) public
onlyOwner{
    _buyFundFee = newFundFee;
    _buyLPFee = newLpFee;
  }
function setSell(uint256 newFundFee, uint256 newLpFee) public
onlyOwner{
    _sellFundFee = newFundFee;
    _sellLPFee = newLpFee;
  }
```

Suggestion:

Emit an event for critical changes.

MANUAL TESTING

Centralization – Local Variable Shadowing

Severity: Low

Status: Open

Function: _approve and allowance

Overview:

```
function allowance(address owner, address spender) public view override
returns (uint256) {
    return _allowances[owner][spender];
  }
function _approve(address owner, address spender, uint256 amount)
private {
    _allowances[owner][spender] = amount;
    emit Approval(owner, spender, amount);
  }
```

Suggestion:

Rename the local variable that shadows another component.

MANUAL TESTING

Optimization

Severity: Low

Subject: Old Pragma Solidity version

Status: Open

Overview:
It is considered best practice to pick one compiler version and stick with it. With a floating pragma, contracts may accidentally be deployed using an outdated.

pragma solidity ^0.8.18;

Suggestion:
Adding the latest constant version of solidity is recommended, as this prevents the unintentional deployment of a contract with an outdated compiler that contains unresolved bugs.

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