Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Customer: Angel Token
Website: http://angel.yt
Platform: Binance Smart Chain
Language: Solidity
Date: October 15th, 2021

Table of contents

Email: audit@EtherAuthority.io

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the Angel Token team to perform the Security audit of the Angel Token smart contract code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on October 15th, 2021.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

Angel (ANG) coin is a BEP20 standard token smart contract with other customization like: swapping, adding liquidity, reflation, etc. ANG Coin has a deflation mechanism, and every transaction will generate rewards, which will be distributed to all holding addresses.

Audit scope

Name	**Code Review and Security Analysis Report for Angel Token Smart Contract**
Platform	**BSC / Solidity**
File	Angel.sol
File MD5 Hash	AEE84C6F83F2E0E6384D3B369923C883
Online Code	https://bscscan.com/address/0x2b9dbe80a5Af34e3D49EA2d6c401A9870404607F#code
Audit Date	October 15th, 2021

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
Tokenomics: • Name: Angel • Symbol: ANG • Decimals: 9 • Total Supply: 1,000 trillion •	**YES, This is valid.**
• Tax Fee: 2% • Liquidity Fee: 10% • Buy Tax Fee: 2% • Buy Liquidity Fee: 3% • Sell Tax Fee: 2% • Sell Liquidity Fee: 6% • Buy Back Range Rate: 80% • Marketing Divisor: 3 • Maximum Tax Amount: 3,000 billions • Minimum Tokens Before Swap: 200 billions • Buy Back Divisor: 10 • Buy Back Time Interval: 5 minutes • Buy Back Maximum Time For Histories:1 Day • Swap Interval Time: 1 minute	**YES, This is valid.** **These fee percentages can be changed by the owner. So, we suggest handling the private key of the owner's wallet securely, and the owner must change these values as per the business plan.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. This token contract contains owner control, which does not make it fully decentralized.

We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.
All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 1 medium and 2 low and some very low level issues.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Moderate
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Moderate
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderate
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	Passed
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Moderate
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Email: audit@EtherAuthority.io

Code Quality

This audit scope has 1 smart contract file. Smart contract contains Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in Angel Token are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts.

The Angel Token team has **not** provided scenario and unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are **not well** commented on smart contracts.

Documentation

We were given an Angel Token smart contracts code in the form of a BSCScan web link. The hash of that code is mentioned above in the table.

As mentioned above, code parts are **not well** commented. So it is not easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Another source of information was its official website http://angel.yt which provided rich information about the project architecture and tokenomics.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	lockTheSwap	modifier	Passed	No Issue
3	name	read	Passed	No Issue
4	symbol	read	Passed	No Issue
5	decimals	read	Passed	No Issue
6	totalSupply	read	Passed	No Issue
7	balanceOf	read	Passed	No Issue
8	transfer	write	Passed	No Issue
9	allowance	read	Passed	No Issue
10	approve	write	Passed	No Issue
11	transferFrom	write	Passed	No Issue
12	increaseAllowance	write	Passed	No Issue
13	decreaseAllowance	write	Passed	No Issue
14	isExcludedFromReward	read	Passed	No Issue
15	totalFees	read	Passed	No Issue
16	minimumTokensBeforeSwapAmount	read	Passed	No Issue
17	buyBackSellLimitAmount	read	Passed	No Issue
18	deliver	write	Missing Events emitting	Refer audit finding section
19	reflectionFromToken	read	Passed	No Issue
20	tokenFromReflection	read	Passed	No Issue
21	excludeFromReward	write	access only Owner	No Issue
22	includeInReward	external	Infinite loop possibility	Refer audit finding section
23	_approve	write	Passed	No Issue
24	_transfer	write	Passed	No Issue
25	swapTokens	write	access by lockTheSwap	No Issue
26	buyBackTokens	write	access by lockTheSwap	No Issue
27	swapTokensForEth	write	Passed	No Issue
28	swapETHForTokens	write	Passed	No Issue
29	addLiquidity	write	Centralized risk in addLiquidity	Refer audit finding section
30	_tokenTransfer	write	Passed	No Issue
31	_transferStandard	write	Passed	No Issue
32	_transferToExcluded	write	Passed	No Issue
33	_transferFromExcluded	write	Passed	No Issue
34	_transferBothExcluded	write	Passed	No Issue
35	_reflectFee	write	Passed	No Issue
36	_getValues	read	Passed	No Issue

37	_getTValues	read	Passed	No Issue
38	_getRValues	write	Passed	No Issue
39	_getRate	read	Passed	No Issue
40	_getCurrentSupply	read	Passed	No Issue
41	_takeLiquidity	write	Passed	No Issue
42	calculateTaxFee	read	Passed	No Issue
43	calculateLiquidityFee	read	Passed	No Issue
44	removeAllFee	write	Passed	No Issue
45	restoreAllFee	write	Passed	No Issue
46	isExcludedFromFee	read	Missing zero address validation	Refer audit finding section
47	excludeFromFee	write	Missing zero address validation	Refer audit finding section
48	includeInFee	write	Missing zero address validation	Refer audit finding section
49	_getSellBnBAmount	read	Passed	No Issue
50	_removeOldSellHistories	write	Passed	No Issue
51	SetBuyBackMaxTimeForHistories	external	access only Owner	No Issue
52	SetBuyBackDivisor	external	access only Owner	No Issue
53	GetBuyBackTimeInterval	read	Passed	No Issue
54	SetBuyBackTimeInterval	external	access only Owner	No Issue
55	SetBuyBackRangeRate	external	access only Owner	No Issue
56	GetSwapMinutes	read	Passed	No Issue
57	SetSwapMinutes	external	access only Owner	No Issue
58	setTaxFeePercent	external	Missing Events emitting	Refer audit finding section
59	setBuyFee	external	Missing Events emitting	Refer audit finding section
60	setSellFee	external	Missing Events emitting	Refer audit finding section
61	setLiquidityFeePercent	external	Missing Events emitting	Refer audit finding section
62	setBuyBackSellLimit	external	Missing Events emitting	Refer audit finding section
63	setMaxTxAmount	external	Missing Events emitting	Refer audit finding section
64	setMarketingDivisor	external	Missing Events emitting	Refer audit finding section
65	setNumTokensSellToAddToBuyBack	external	Missing Events emitting	Refer audit finding section

66	setMarketingAddress	external	Missing zero address validation	Refer audit finding section
67	setSwapAndLiquifyEnabled	write	access only Owner	No Issue
68	setBuyBackEnabled	write	access only Owner	No Issue
69	setAutoBuyBackEnabled	write	access only Owner	No Issue
70	prepareForPreSale	external	access only Owner	No Issue
71	afterPreSale	external	access only Owner	No Issue
72	transferToAddressETH	write	Passed	No Issue
73	changeRouterVersion	write	Missing zero address validation	Refer audit finding section
74	receive	external	Passed	No Issue
75	transferForeignToken	write	Missing zero address validation	Refer audit finding section
76	Sweep	external	access only Owner	No Issue
77	setAddressFee	external	Missing zero address validation	Refer audit finding section
78	setBuyAddressFee	external	Missing zero address validation	Refer audit finding section
79	setSellAddressFee	external	Missing zero address validation	Refer audit finding section
80	_msgSender	internal	Passed	No Issue
81	_msgSender	internal	Passed	No Issue
82	owner	read	Passed	No Issue
83	onlyOwner	modifier	Passed	No Issue
84	renounceOwnership	write	access only Owner	No Issue
85	transferOwnership	write	access only Owner	No Issue
86	getUnlockTime	read	Passed	No Issue
87	getTime	read	Passed	No Issue
88	lock	write	access only Owner	No Issue
89	unlock	write	Passed	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

(1) Infinite loop possibility:

```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is not excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

If there are so many excluded accounts , then this logic will fail, as it might hit the block's gas limit. If there are very limited excluded accounts, then this will work, but will cost more gas.

Resolution: Just use a mapping that will map wallet to bool and make excluded wallets to be true. This logic will not have any gas or scalability issues.

Low

(1) Missing zero address validation:

```
function transferForeignToken(address _token, address _to) public onlyOwner returns(bool _sent){
    require(_token != address(this), "Can't let you take all native token");
    uint256 _contractBalance = IERC20(_token).balanceOf(address(this));
    _sent = IERC20(_token).transfer(_to, _contractBalance);
}
```

```
function setAddressFee(address _address, bool _enable, uint256 _addressTaxFee, uint256 _addressLiquidityFee) external onlyOwner {
    _addressFees[_address].enable = _enable;
    _addressFees[_address]._taxFee = _addressTaxFee;
    _addressFees[_address]._liquidityFee = _addressLiquidityFee;
}

function setBuyAddressFee(address _address, bool _enable, uint256 _addressTaxFee, uint256 _addressLiquidityFee) external onlyOwner {
    _addressFees[_address].enable = _enable;
    _addressFees[_address]._buyTaxFee = _addressTaxFee;
    _addressFees[_address]._buyLiquidityFee = _addressLiquidityFee;
}

function setSellAddressFee(address _address, bool _enable, uint256 _addressTaxFee, uint256 _addressLiquidityFee) external onlyOwner {
    _addressFees[_address].enable = _enable;
    _addressFees[_address]._sellTaxFee = _addressTaxFee;
    _addressFees[_address]._sellLiquidityFee = _addressLiquidityFee;
}
```

```
function changeRouterVersion(address _router) public onlyOwner returns(address _pair) {
    IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(_router);

    _pair = IUniswapV2Factory(_uniswapV2Router.factory()).getPair(address(this), _uniswapV2Router.WETH());
    if(_pair == address(0)){
        // Pair doesn't exist
        _pair = IUniswapV2Factory(_uniswapV2Router.factory())
        .createPair(address(this), _uniswapV2Router.WETH());
    }
    uniswapV2Pair = _pair;

    // Set the router of the contract variables
    uniswapV2Router = _uniswapV2Router;
}
```

```
function setMarketingAddress(address _marketingAddress) external onlyOwner {
    marketingAddress = payable(_marketingAddress);
}
```

```
function isExcludedFromFee(address account) public view returns(bool) {
    return _isExcludedFromFee[account];
}

function excludeFromFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = true;
}

function includeInFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = false;
}
```

Detect missing zero address validation in setAddressFee, setBuyAddressFee, setSellAddressFee, transferForeignToken, changeRouterVersion, setMarketingAddress, isExcludedFromFee, excludeFromFee, includeInFee, "Line no 1173 to 1190 : Check that the _address is not zero.

Resolution: Line no 1162 : check that _to is not zero.

Line no 1143 : check that _router is not zero

Line no 1106 : check that _marketingAddress is not zero

Line no 995 to 1005 : check that account address is not zero

(2) Centralized risk in addLiquidity:

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    // Approve token transfer to cover all possible scenarios
    _approve(address(this), address(uniswapV2Router), tokenAmount);

    // Add the liquidity
    uniswapV2Router.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0, // Slippage is unavoidable
        0, // Slippage is unavoidable
        owner(),
        block.timestamp
    );
}
```

In addLiquidityETH function, owner gets ANG Tokens from the Pool. If the private key of the owner's wallet is compromised, then it will create a problem.

Resolution:

Ideally this can be a governance smart contract. On another hand, the owner can accept this risk and handle the private key very securely.

(3) Owner can drain all the balance of contract:

```
function Sweep() external onlyOwner {
    uint256 balance = address(this).balance;
    payable(owner()).transfer(balance);
}
```

Resolution:

The owner can accept this risk and handle the private key very securely.

Very Low / Informational / Best practices:

(1) Missing Events emitting:

Many functions change state variables to emit events for these functions.

Functions needs events are : deliver, excludeFromFee, excludeFromReward, includeInFee, includeInReward, setLiquidityFeePercent, setMaxTxAmount,

SetSwapMinutes, setTaxFeePercent, setBuyFee, setSellFee, setBuyBackSellLimit, setMarketingDivisor, setNumTokensSellToAddToBuyBack, setMarketingAddress, setAddressFee, setBuyAddressFee, setSellAddressFee.

Resolution: For all the state variables which change runtime this needs to emit an event. Recommended to emit events for all these functions.

(2) Use latest solidity version:

```
pragma solidity ^0.8.4;
```

Using the latest solidity will prevent any compiler level bugs.
Resolution: Please use 0.8.9 which is the latest version.

(3) All functions which are not called internally, must be declared as external. It is more efficient as sometimes it saves some gas.
https://ethereum.stackexchange.com/questions/19380/external-vs-public-best-practices

Centralization

These smart contracts have some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

- excludeFromReward: The Owner can exclude from reward.
- includeInReward: The Owner can be included in the reward.
- excludeFromFee: The Owner can exclude from fee.
- includeInFee: The Owner can include from fee.
- SetBuyBackMaxTimeForHistories: The Owner can set buy back maximum time for histories.
- SetBuyBackDivisor: The Owner can set buy back divisor.
- SetBuyBackTimeInterval: The Owner can set a buy back time interval.
- SetBuyBackRangeRate: The Owner can set buy back range rate.
- SetSwapMinutes: The Owner can set swap minutes.

- setTaxFeePercent: The Owner can set the tax fee percentage.
- setBuyFee: The Owner can set a buy fee.
- setSellFee: The Owner can set a sale fee.
- setLiquidityFeePercent: The Owner can set Liquidity Fee Percentage.
- setBuyBackSellLimit: The Owner can set a buy back sell limit.
- setMaxTxAmount: The Owner can set Max tax amount.
- setMarketingDivisor: The Owner can set Marketing divisor.
- setNumTokensSellToAddToBuyBack: The Owner can set number token sales to add to buy back.
- setMarketingAddress: The Owner can set Marketing address.
- setSwapAndLiquifyEnabled: The Owner can set swap and liquify enabled.
- setBuyBackEnabled: The Owner can set buy back enabled.
- setAutoBuyBackEnabled: The Owner can set auto buy back enabled.
- prepareForPreSale: The Owner can prepare for pre-sale.
- afterPreSale: The Owner can set after pre-sale.
- changeRouterVersion: The Owner can change the router version address.
- transferForeignToken: The Owner can transfer foreign token addresses.
- Sweep: The Owner can set the sweep.
- setAddressFee: The Owner can set the address fee.
- setBuyAddressFee: The Owner can set a buy address fee.
- setSellAddressFee: The Owner can set the sell address Fee.

Conclusion

We were given a contract code. And we have used all possible tests based on given objects as files. We observed some issues in the smart contracts and those issues are not very critical ones. So, **it's good to go to production**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured".**

Email: audit@EtherAuthority.io

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

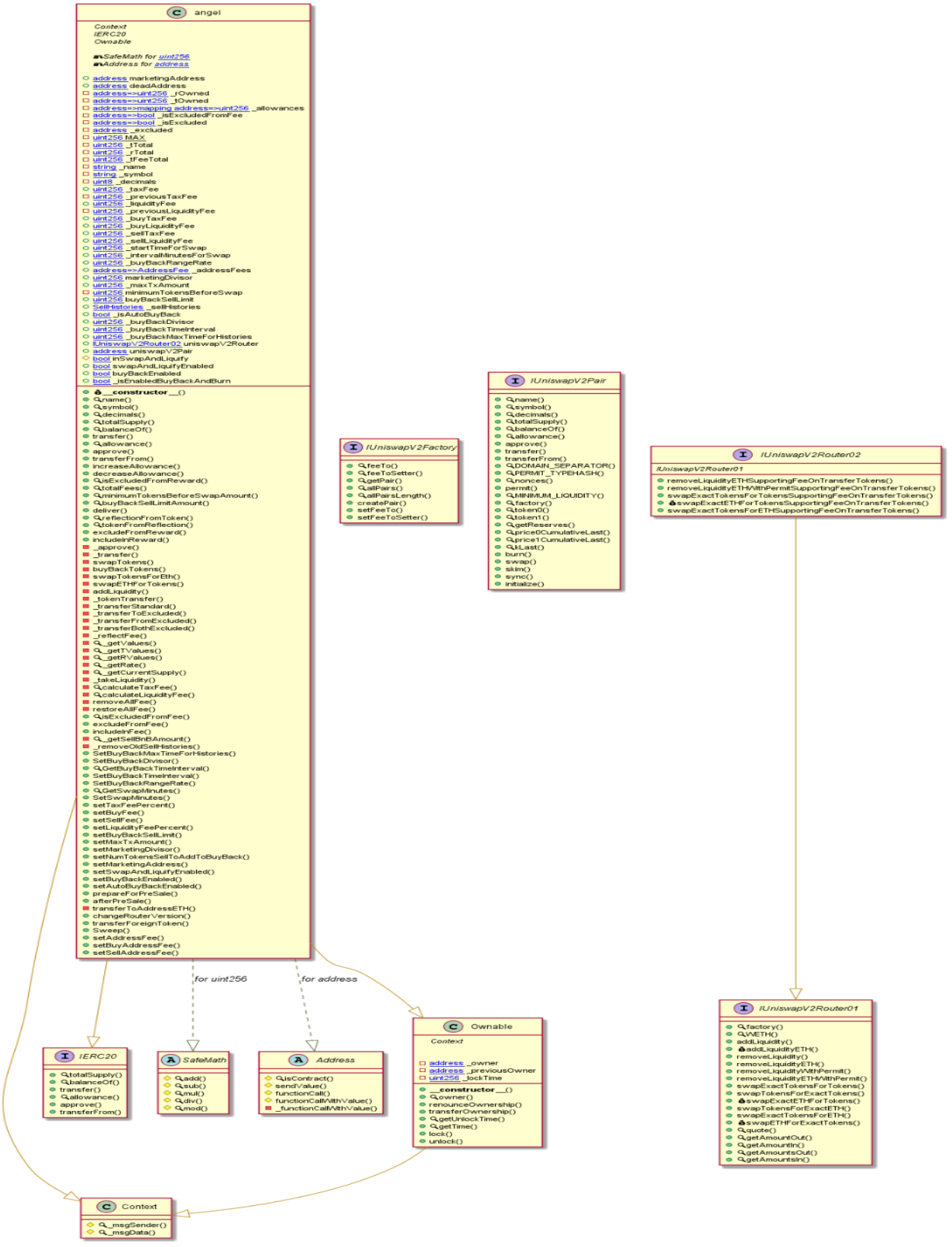
Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can’t guarantee explicit security of the audited smart contracts.

Appendix

Code Flow Diagram - Angel Token



Slither Results Log

Slither log >> Angel.sol

```
INFO:Detectors:
angel.swapETHForTokens(uint256) (angel.sol#818-833) sends eth to arbitrary user
        Dangerous calls:
        - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300))
(angel.sol#825-830)
angel.addLiquidity(uint256,uint256) (angel.sol#835-848) sends eth to arbitrary user
        Dangerous calls:
        - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (angel.sol#840-847)
angel.Sweep() (angel.sol#1159-1162) sends eth to arbitrary user
        Dangerous calls:
        - address(owner()).transfer(balance) (angel.sol#1161)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations
INFO:Detectors:
angel._transfer(address,address,uint256) (angel.sol#661-778) uses a weak PRNG: "_bBSLimit = _bBSLimitMin + uint256(keccak256(bytes)(abi.e
ncodePacked(block.timestamp,block.difficulty))) % (_bBSLimitMax - _bBSLimitMin + 1) (angel.sol#722)"
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#weak-PRNG
INFO:Detectors:
Reentrancy in angel._transfer(address,address,uint256) (angel.sol#661-778):
        External calls:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (a
ngel.sol#807-813)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
        State variables written after the call(s):
        - _removeOldSellHistories() (angel.sol#717)
                - _sellHistories[i].time = _sellHistories[j].time (angel.sol#1018)
                - _sellHistories[i].bnbAmount = _sellHistories[j].bnbAmount (angel.sol#1019)
                - _sellHistories.pop() (angel.sol#1029)
Reentrancy in angel._transfer(address,address,uint256) (angel.sol#661-778):
        External calls:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (a
ngel.sol#807-813)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
```

```
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (angel.sol#954)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#870)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#879)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#871)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#890)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#900)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#891)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#881)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#902)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _rTotal = _rTotal.sub(rFee) (angel.sol#909)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (angel.sol#956)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (angel.sol#899)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (angel.sol#889)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (angel.sol#880)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (angel.sol#901)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - inSwapAndLiquify = true (angel.sol#507)
                - inSwapAndLiquify = false (angel.sol#509)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
angel._transfer(address,address,uint256).sellHistory (angel.sol#677) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
angel.addLiquidity(uint256,uint256) (angel.sol#835-848) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}(address
(this),tokenAmount,0,0,owner(),block.timestamp) (angel.sol#840-847)
```

```
INFO:Detectors:
angel.allowance(address,address).owner (angel.sol#562) shadows:
        - Ownable.owner() (angel.sol#161-163) (function)
angel._approve(address,address,uint256).owner (angel.sol#653) shadows:
        - Ownable.owner() (angel.sol#161-163) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
angel.setMarketingAddress(address)._marketingAddress (angel.sol#1097) lacks a zero-check on :
        - marketingAddress = address(_marketingAddress) (angel.sol#1098)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in angel._transfer(address,address,uint256) (angel.sol#661-778):
        External calls:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (a
ngel.sol#807-813)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        State variables written after the call(s):
```

```
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        State variables written after the call(s):
        - removeAllFee() (angel.sol#741)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _buyLiquidityFee (angel.sol#743)
        - removeAllFee() (angel.sol#747)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _sellLiquidityFee (angel.sol#749)
        - removeAllFee() (angel.sol#754)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _addressFees[from]._liquidityFee (angel.sol#756)
        - _liquidityFee = _addressFees[from]._sellLiquidityFee (angel.sol#761)
        - removeAllFee() (angel.sol#768)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _addressFees[to]._buyLiquidityFee (angel.sol#771)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _liquidityFee = _previousLiquidityFee (angel.sol#983)
                - _liquidityFee = 0 (angel.sol#978)
        - removeAllFee() (angel.sol#741)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - removeAllFee() (angel.sol#747)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - removeAllFee() (angel.sol#754)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - removeAllFee() (angel.sol#768)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - removeAllFee() (angel.sol#741)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - removeAllFee() (angel.sol#747)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - removeAllFee() (angel.sol#754)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - removeAllFee() (angel.sol#768)
                - _previousTaxFee = _taxFee (angel.sol#974)
```

```
        - removeAllFee() (angel.sol#768)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _tFeeTotal = _tFeeTotal.add(tFee) (angel.sol#910)
        - removeAllFee() (angel.sol#741)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _buyTaxFee (angel.sol#742)
        - removeAllFee() (angel.sol#747)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _sellTaxFee (angel.sol#748)
        - removeAllFee() (angel.sol#754)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _addressFees[from]._taxFee (angel.sol#755)
        - _taxFee = _addressFees[from]._sellTaxFee (angel.sol#760)
        - removeAllFee() (angel.sol#768)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _addressFees[to]._buyTaxFee (angel.sol#770)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _taxFee = _previousTaxFee (angel.sol#982)
                - _taxFee = 0 (angel.sol#977)
Reentrancy in angel.changeRouterVersion(address) (angel.sol#1134-1147):
        External calls:
        - _pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (angel.sol#1140-1141)
        State variables written after the call(s):
        - uniswapV2Pair = _pair (angel.sol#1143)
        - uniswapV2Router = _uniswapV2Router (angel.sol#1146)
Reentrancy in angel.constructor() (angel.sol#512-534):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (angel.sol#522-
523)
        State variables written after the call(s):
        - _isExcludedFromFee[owner()] = true (angel.sol#528)
        - _isExcludedFromFee[address(this)] = true (angel.sol#529)
        - _startTimeForSwap = block.timestamp (angel.sol#531)
        - uniswapV2Router = _uniswapV2Router (angel.sol#525)
Reentrancy in angel.transferFrom(address,address,uint256) (angel.sol#571-575):
```

```
        External calls:
        - _transfer(sender,recipient,amount) (angel.sol#572)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (a
ngel.sol#807-813)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (angel.sol#572)
                - recipient.transfer(amount) (angel.sol#1131)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (angel.sol
#573)
                - _allowances[owner][spender] = amount (angel.sol#657)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in angel._transfer(address,address,uint256) (angel.sol#661-778):
        External calls:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (a
ngel.sol#807-813)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (angel.sol#832)
                - buyBackTokens(_bBSLimit) (angel.sol#725)
```

```
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#894)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#884)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#905)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
Reentrancy in angel.constructor() (angel.sol#512-534):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (angel.sol#522-
523)
        Event emitted after the call(s):
        - Transfer(address(0),_msgSender(),_tTotal) (angel.sol#533)
Reentrancy in angel.swapETHForTokens(uint256) (angel.sol#818-833):
        External calls:
        - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.add(300))
(angel.sol#825-830)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (angel.sol#832)
Reentrancy in angel.swapTokensForEth(uint256) (angel.sol#798-816):
        External calls:
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (angel.sol
#807-813)
        Event emitted after the call(s):
        - SwapTokensForETH(tokenAmount,path) (angel.sol#815)
Reentrancy in angel.transferFrom(address,address,uint256) (angel.sol#571-575):
        External calls:
        - _transfer(sender,recipient,amount) (angel.sol#572)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (a
ngel.sol#807-813)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (angel.sol#572)
                - recipient.transfer(amount) (angel.sol#1131)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        Event emitted after the call(s):
```

```
INFO:Detectors:
Ownable.unlock() (angel.sol#196-201) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp > _lockTime,Contract is locked until 7 days) (angel.sol#198)
angel._transfer(address,address,uint256) (angel.sol#661-778) uses timestamp for comparisons
        Dangerous comparisons:
        - overMinimumTokenBalance && _startTimeForSwap + _intervalMinutesForSwap <= block.timestamp (angel.sol#687)
        - _sellHistories[i].time >= startTime (angel.sol#707)
        - balance > _bBSLimit (angel.sol#724)
angel.buyBackTokens(uint256) (angel.sol#792-796) uses timestamp for comparisons
        Dangerous comparisons:
        - amount > 0 (angel.sol#793)
angel._removeOldSellHistories() (angel.sol#1010-1032) uses timestamp for comparisons
        Dangerous comparisons:
        - _sellHistories[j].time >= maxStartTimeForHistories (angel.sol#1016)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (angel.sol#91-100) uses assembly
        - INLINE ASM (angel.sol#98)
Address._functionCallWithValue(address,bytes,uint256,string) (angel.sol#128-145) uses assembly
        - INLINE ASM (angel.sol#137-140)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (angel.sol#128-145) is never used and should be removed
Address.functionCall(address,bytes) (angel.sol#111-113) is never used and should be removed
Address.functionCall(address,bytes,string) (angel.sol#115-117) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (angel.sol#119-121) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (angel.sol#123-126) is never used and should be removed
Address.isContract(address) (angel.sol#91-100) is never used and should be removed
Address.sendValue(address,uint256) (angel.sol#102-108) is never used and should be removed
Context._msgData() (angel.sol#14-17) is never used and should be removed
SafeMath.mod(uint256,uint256) (angel.sol#79-81) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (angel.sol#83-86) is never used and should be removed
angel.addLiquidity(uint256,uint256) (angel.sol#835-848) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
angel._rTotal (angel.sol#422) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
```

```
INFO:Detectors:
Pragma version^0.8.4 (angel.sol#7) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (angel.sol#102-108):
        - (success) = recipient.call{value: amount}() (angel.sol#106)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (angel.sol#128-145):
        - (success,returndata) = target.call{value: weiValue}(data) (angel.sol#131)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (angel.sol#236) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (angel.sol#237) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (angel.sol#253) is not in mixedCase
Function IUniswapV2Router01.WETH() (angel.sol#272) is not in mixedCase
Contract angel (angel.sol#405-1183) is not in CapWords
Parameter angel.calculateTaxFee(uint256)._amount (angel.sol#959) is not in mixedCase
Parameter angel.calculateLiquidityFee(uint256)._amount (angel.sol#965) is not in mixedCase
Function angel.SetBuyBackMaxTimeForHistories(uint256) (angel.sol#1034-1036) is not in mixedCase
Function angel.SetBuyBackDivisor(uint256) (angel.sol#1038-1040) is not in mixedCase
Function angel.GetBuyBackTimeInterval() (angel.sol#1042-1044) is not in mixedCase
Function angel.SetBuyBackTimeInterval(uint256) (angel.sol#1046-1048) is not in mixedCase
Function angel.SetBuyBackRangeRate(uint256) (angel.sol#1050-1053) is not in mixedCase
Function angel.GetSwapMinutes() (angel.sol#1055-1057) is not in mixedCase
Function angel.SetSwapMinutes(uint256) (angel.sol#1059-1061) is not in mixedCase
Parameter angel.setNumTokensSellToAddToBuyBack(uint256)._minimumTokensBeforeSwap (angel.sol#1093) is not in mixedCase
Parameter angel.setMarketingAddress(address)._marketingAddress (angel.sol#1097) is not in mixedCase
Parameter angel.setSwapAndLiquifyEnabled(bool)._enabled (angel.sol#1101) is not in mixedCase
Parameter angel.setBuyBackEnabled(bool)._enabled (angel.sol#1106) is not in mixedCase
Parameter angel.setAutoBuyBackEnabled(bool)._enabled (angel.sol#1111) is not in mixedCase
Parameter angel.changeRouterVersion(address)._router (angel.sol#1134) is not in mixedCase
Parameter angel.transferForeignToken(address,address)._token (angel.sol#1153) is not in mixedCase
Parameter angel.transferForeignToken(address,address)._to (angel.sol#1153) is not in mixedCase
```

```
Parameter angel.setAddressFee(address,bool,uint256,uint256)._address (angel.sol#1164) is not in mixedCase
Parameter angel.setAddressFee(address,bool,uint256,uint256)._enable (angel.sol#1164) is not in mixedCase
Parameter angel.setAddressFee(address,bool,uint256,uint256)._addressTaxFee (angel.sol#1164) is not in mixedCase
Parameter angel.setAddressFee(address,bool,uint256,uint256)._addressLiquidityFee (angel.sol#1164) is not in mixedCase
Parameter angel.setBuyAddressFee(address,bool,uint256,uint256)._address (angel.sol#1170) is not in mixedCase
Parameter angel.setBuyAddressFee(address,bool,uint256,uint256)._enable (angel.sol#1170) is not in mixedCase
Parameter angel.setBuyAddressFee(address,bool,uint256,uint256)._addressTaxFee (angel.sol#1170) is not in mixedCase
Parameter angel.setBuyAddressFee(address,bool,uint256,uint256)._addressLiquidityFee (angel.sol#1170) is not in mixedCase
Parameter angel.setSellAddressFee(address,bool,uint256,uint256)._address (angel.sol#1176) is not in mixedCase
Parameter angel.setSellAddressFee(address,bool,uint256,uint256)._enable (angel.sol#1176) is not in mixedCase
Parameter angel.setSellAddressFee(address,bool,uint256,uint256)._addressTaxFee (angel.sol#1176) is not in mixedCase
Parameter angel.setSellAddressFee(address,bool,uint256,uint256)._addressLiquidityFee (angel.sol#1176) is not in mixedCase
Variable angel._taxFee (angel.sol#444) is not in mixedCase
Variable angel._liquidityFee (angel.sol#447) is not in mixedCase
Variable angel._buyTaxFee (angel.sol#450) is not in mixedCase
Variable angel._buyLiquidityFee (angel.sol#451) is not in mixedCase
Variable angel._sellTaxFee (angel.sol#453) is not in mixedCase
Variable angel._sellLiquidityFee (angel.sol#454) is not in mixedCase
Variable angel._startTimeForSwap (angel.sol#456) is not in mixedCase
Variable angel._intervalMinutesForSwap (angel.sol#457) is not in mixedCase
Variable angel._buyBackRangeRate (angel.sol#459) is not in mixedCase
Variable angel._addressFees (angel.sol#462) is not in mixedCase
Variable angel._maxTxAmount (angel.sol#466) is not in mixedCase
Variable angel._sellHistories (angel.sol#471) is not in mixedCase
Variable angel._isAutoBuyBack (angel.sol#472) is not in mixedCase
Variable angel._buyBackDivisor (angel.sol#473) is not in mixedCase
Variable angel._buyBackTimeInterval (angel.sol#474) is not in mixedCase
Variable angel._buyBackMaxTimeForHistories (angel.sol#475) is not in mixedCase
Variable angel._isEnabledBuyBackAndBurn (angel.sol#484) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (angel.sol#15)" inContext (angel.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in angel._transfer(address,address,uint256) (angel.sol#661-778):
        External calls:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
```

```
        State variables written after the call(s):
        - _removeOldSellHistories() (angel.sol#717)
                - _sellHistories[i].time = _sellHistories[j].time (angel.sol#1018)
                - _sellHistories[i].bnbAmount = _sellHistories[j].bnbAmount (angel.sol#1019)
                - _sellHistories.pop() (angel.sol#1029)
Reentrancy in angel._transfer(address,address,uint256) (angel.sol#661-778):
        External calls:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
        External calls sending eth:
        - swapTokens(contractTokenBalance) (angel.sol#690)
                - recipient.transfer(amount) (angel.sol#1131)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        State variables written after the call(s):
        - removeAllFee() (angel.sol#741)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _buyLiquidityFee (angel.sol#743)
        - removeAllFee() (angel.sol#747)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _sellLiquidityFee (angel.sol#749)
        - removeAllFee() (angel.sol#754)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _addressFees[from]._liquidityFee (angel.sol#756)
        - _liquidityFee = _addressFees[from]._sellLiquidityFee (angel.sol#761)
        - removeAllFee() (angel.sol#768)
                - _liquidityFee = 0 (angel.sol#978)
        - _liquidityFee = _addressFees[to]._buyLiquidityFee (angel.sol#771)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _liquidityFee = _previousLiquidityFee (angel.sol#983)
                - _liquidityFee = 0 (angel.sol#978)
        - removeAllFee() (angel.sol#741)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - removeAllFee() (angel.sol#747)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
        - removeAllFee() (angel.sol#754)
                - _previousLiquidityFee = _liquidityFee (angel.sol#975)
```

```
        - removeAllFee() (angel.sol#741)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - removeAllFee() (angel.sol#747)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - removeAllFee() (angel.sol#754)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - removeAllFee() (angel.sol#768)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _previousTaxFee = _taxFee (angel.sol#974)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (angel.sol#954)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#870)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#879)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#871)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#890)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (angel.sol#900)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#891)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#881)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (angel.sol#902)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _rTotal = _rTotal.sub(rFee) (angel.sol#909)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _tFeeTotal = _tFeeTotal.add(tFee) (angel.sol#910)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (angel.sol#956)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (angel.sol#899)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (angel.sol#889)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (angel.sol#880)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (angel.sol#901)
        - removeAllFee() (angel.sol#741)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _buyTaxFee (angel.sol#742)
        - removeAllFee() (angel.sol#747)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _sellTaxFee (angel.sol#748)
        - removeAllFee() (angel.sol#754)
                - _taxFee = 0 (angel.sol#977)
```

```
        - _taxFee = _addressFees[from]._taxFee (angel.sol#755)
        - _taxFee = _addressFees[from]._sellTaxFee (angel.sol#760)
        - removeAllFee() (angel.sol#768)
                - _taxFee = 0 (angel.sol#977)
        - _taxFee = _addressFees[to]._buyTaxFee (angel.sol#770)
        - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
                - _taxFee = _previousTaxFee (angel.sol#982)
                - _taxFee = 0 (angel.sol#977)
        - buyBackTokens(_bBSLimit) (angel.sol#725)
                - inSwapAndLiquify = true (angel.sol#507)
                - inSwapAndLiquify = false (angel.sol#509)
        Event emitted after the call(s):
        - SwapETHForTokens(amount,path) (angel.sol#832)
                - buyBackTokens(_bBSLimit) (angel.sol#725)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#874)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#894)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#884)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
        - Transfer(sender,recipient,tTransferAmount) (angel.sol#905)
                - _tokenTransfer(from,to,amount,takeFee) (angel.sol#777)
Reentrancy in angel.transferFrom(address,address,uint256) (angel.sol#571-575):
        External calls:
        - _transfer(sender,recipient,amount) (angel.sol#572)
                - recipient.transfer(amount) (angel.sol#1131)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (angel.sol#572)
                - recipient.transfer(amount) (angel.sol#1131)
                - uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amount}(0,path,deadAddress,block.timestamp.ad
d(300)) (angel.sol#825-830)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (angel.sol
#573)
                - _allowances[owner][spender] = amount (angel.sol#657)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (angel.sol#658)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (a
```

```
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (angel.sol#277)
is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (angel.
sol#278)
Variable angel._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (angel.sol#930) is too similar to angel._transferStandard(add
ress,address,uint256).tTransferAmount (angel.sol#869)
Variable angel._getValues(uint256).rTransferAmount (angel.sol#915) is too similar to angel._transferFromExcluded(address,address,uint256)
.tTransferAmount (angel.sol#888)
Variable angel._transferToExcluded(address,address,uint256).rTransferAmount (angel.sol#878) is too similar to angel._transferStandard(add
ress,address,uint256).tTransferAmount (angel.sol#869)
Variable angel._transferToExcluded(address,address,uint256).rTransferAmount (angel.sol#878) is too similar to angel._transferBothExcluded
(address,address,uint256).tTransferAmount (angel.sol#898)
Variable angel._transferToExcluded(address,address,uint256).rTransferAmount (angel.sol#878) is too similar to angel._getTValues(uint256).
tTransferAmount (angel.sol#922)
Variable angel._transferToExcluded(address,address,uint256).rTransferAmount (angel.sol#878) is too similar to angel._getValues(uint256).t
TransferAmount (angel.sol#914)
Variable angel._transferToExcluded(address,address,uint256).rTransferAmount (angel.sol#878) is too similar to angel._transferToExcluded(a
ddress,address,uint256).tTransferAmount (angel.sol#878)
Variable angel._transferStandard(address,address,uint256).rTransferAmount (angel.sol#869) is too similar to angel._transferFromExcluded(a
ddress,address,uint256).tTransferAmount (angel.sol#888)
Variable angel._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (angel.sol#930) is too similar to angel._transferFromExcluded
(address,address,uint256).tTransferAmount (angel.sol#888)
Variable angel._transferToExcluded(address,address,uint256).rTransferAmount (angel.sol#878) is too similar to angel._transferFromExcluded
(address,address,uint256).tTransferAmount (angel.sol#888)
Variable angel._transferBothExcluded(address,address,uint256).rTransferAmount (angel.sol#898) is too similar to angel._transferFromExclud
ed(address,address,uint256).tTransferAmount (angel.sol#888)
Variable angel._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (angel.sol#930) is too similar to angel._getValues(uint256).t
TransferAmount (angel.sol#914)
Variable angel._transferBothExcluded(address,address,uint256).rTransferAmount (angel.sol#898) is too similar to angel._getValues(uint256)
.tTransferAmount (angel.sol#914)
Variable angel._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (angel.sol#930) is too similar to angel._transferBothExcluded
(address,address,uint256).tTransferAmount (angel.sol#898)
Variable angel._transferStandard(address,address,uint256).rTransferAmount (angel.sol#869) is too similar to angel._transferStandard(addre
ss,address,uint256).tTransferAmount (angel.sol#869)
Variable angel._transferFromExcluded(address,address,uint256).rTransferAmount (angel.sol#888) is too similar to angel._transferFromExclud
ed(address,address,uint256).tTransferAmount (angel.sol#888)
Variable angel.reflectionFromToken(uint256,bool).rTransferAmount (angel.sol#619) is too similar to angel._transferFromExcluded(address,ad
dress,uint256).tTransferAmount (angel.sol#888)
```

```
Variable angel._transferBothExcluded(address,address,uint256).rTransferAmount (angel.sol#898) is too similar to angel._transferBothExclud
ed(address,address,uint256).tTransferAmount (angel.sol#898)
Variable angel._transferBothExcluded(address,address,uint256).rTransferAmount (angel.sol#898) is too similar to angel._transferStandard(a
ddress,address,uint256).tTransferAmount (angel.sol#869)
Variable angel.reflectionFromToken(uint256,bool).rTransferAmount (angel.sol#619) is too similar to angel._getValues(uint256).tTransferAmo
unt (angel.sol#914)
Variable angel.reflectionFromToken(uint256,bool).rTransferAmount (angel.sol#619) is too similar to angel._transferBothExcluded(address,ad
dress,uint256).tTransferAmount (angel.sol#898)
Variable angel.reflectionFromToken(uint256,bool).rTransferAmount (angel.sol#619) is too similar to angel._transferStandard(address,addres
s,uint256).tTransferAmount (angel.sol#869)
Variable angel._transferStandard(address,address,uint256).rTransferAmount (angel.sol#869) is too similar to angel._transferToExcluded(add
ress,address,uint256).tTransferAmount (angel.sol#878)
Variable angel._transferStandard(address,address,uint256).rTransferAmount (angel.sol#869) is too similar to angel._transferBothExcluded(a
ddress,address,uint256).tTransferAmount (angel.sol#898)
Variable angel.reflectionFromToken(uint256,bool).rTransferAmount (angel.sol#619) is too similar to angel._transferToExcluded(address,addr
ess,uint256).tTransferAmount (angel.sol#878)
Variable angel.reflectionFromToken(uint256,bool).rTransferAmount (angel.sol#619) is too similar to angel._getTValues(uint256).tTransferAm
ount (angel.sol#922)
Variable angel._getValues(uint256).rTransferAmount (angel.sol#915) is too similar to angel._transferStandard(address,address,uint256).tTr
ansferAmount (angel.sol#869)
Variable angel._transferFromExcluded(address,address,uint256).rTransferAmount (angel.sol#888) is too similar to angel._transferStandard(a
ddress,address,uint256).tTransferAmount (angel.sol#869)
Variable angel._transferStandard(address,address,uint256).rTransferAmount (angel.sol#869) is too similar to angel._getTValues(uint256).tT
ransferAmount (angel.sol#922)
Variable angel._transferFromExcluded(address,address,uint256).rTransferAmount (angel.sol#888) is too similar to angel._transferBothExclud
ed(address,address,uint256).tTransferAmount (angel.sol#898)
Variable angel._transferStandard(address,address,uint256).rTransferAmount (angel.sol#869) is too similar to angel._getValues(uint256).tTr
ansferAmount (angel.sol#914)
Variable angel._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (angel.sol#930) is too similar to angel._transferToExcluded(a
ddress,address,uint256).tTransferAmount (angel.sol#878)
Variable angel._transferFromExcluded(address,address,uint256).rTransferAmount (angel.sol#888) is too similar to angel._getTValues(uint256
).tTransferAmount (angel.sol#922)
Variable angel._transferBothExcluded(address,address,uint256).rTransferAmount (angel.sol#898) is too similar to angel._transferToExcluded
(address,address,uint256).tTransferAmount (angel.sol#878)
Variable angel._getValues(uint256).rTransferAmount (angel.sol#915) is too similar to angel._getValues(uint256).tTransferAmount (angel.sol
#914)
Variable angel._transferFromExcluded(address,address,uint256).rTransferAmount (angel.sol#888) is too similar to angel._getValues(uint256)
.tTransferAmount (angel.sol#914)
```

```
Variable angel._getValues(uint256).rTransferAmount (angel.sol#915) is too similar to angel._transferToExcluded(address,address,uint256).t
TransferAmount (angel.sol#878)
Variable angel._getValues(uint256).rTransferAmount (angel.sol#915) is too similar to angel._getTValues(uint256).tTransferAmount (angel.so
l#922)
Variable angel._transferBothExcluded(address,address,uint256).rTransferAmount (angel.sol#898) is too similar to angel._getTValues(uint256
).tTransferAmount (angel.sol#922)
Variable angel._transferFromExcluded(address,address,uint256).rTransferAmount (angel.sol#888) is too similar to angel._transferToExcluded
(address,address,uint256).tTransferAmount (angel.sol#878)
Variable angel._getValues(uint256).rTransferAmount (angel.sol#915) is too similar to angel._transferBothExcluded(address,address,uint256)
.tTransferAmount (angel.sol#898)
Variable angel._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (angel.sol#930) is too similar to angel._getTValues(uint256).
tTransferAmount (angel.sol#922)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
angel.prepareForPreSale() (angel.sol#1116-1121) uses literals with too many digits:
        - _maxTxAmount = 1000000000 * 10 ** 6 * 10 ** 9 (angel.sol#1120)
angel.afterPreSale() (angel.sol#1123-1128) uses literals with too many digits:
        - _maxTxAmount = 3000000 * 10 ** 6 * 10 ** 9 (angel.sol#1127)
angel.slitherConstructorVariables() (angel.sol#405-1183) uses literals with too many digits:
        - deadAddress = 0x000000000000000000000000000000000000dEaD (angel.sol#410)
angel.slitherConstructorVariables() (angel.sol#405-1183) uses literals with too many digits:
        - _tTotal = 1000000 * 10 ** 9 * 10 ** 9 (angel.sol#421)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
angel._decimals (angel.sol#427) should be constant
angel._isEnabledBuyBackAndBurn (angel.sol#484) should be constant
angel._name (angel.sol#425) should be constant
angel._symbol (angel.sol#426) should be constant
angel._tTotal (angel.sol#421) should be constant
angel.deadAddress (angel.sol#410) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (angel.sol#170-173)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (angel.sol#175-179)
getUnlockTime() should be declared external:
        - Ownable.getUnlockTime() (angel.sol#181-183)
```

```
getTime() should be declared external:
        - Ownable.getTime() (angel.sol#185-187)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (angel.sol#189-194)
unlock() should be declared external:
        - Ownable.unlock() (angel.sol#196-201)
name() should be declared external:
        - angel.name() (angel.sol#536-538)
symbol() should be declared external:
        - angel.symbol() (angel.sol#540-542)
decimals() should be declared external:
        - angel.decimals() (angel.sol#544-546)
totalSupply() should be declared external:
        - angel.totalSupply() (angel.sol#548-550)
transfer(address,uint256) should be declared external:
        - angel.transfer(address,uint256) (angel.sol#557-560)
allowance(address,address) should be declared external:
        - angel.allowance(address,address) (angel.sol#562-564)
approve(address,uint256) should be declared external:
        - angel.approve(address,uint256) (angel.sol#566-569)
transferFrom(address,address,uint256) should be declared external:
        - angel.transferFrom(address,address,uint256) (angel.sol#571-575)
increaseAllowance(address,uint256) should be declared external:
        - angel.increaseAllowance(address,uint256) (angel.sol#577-580)
decreaseAllowance(address,uint256) should be declared external:
        - angel.decreaseAllowance(address,uint256) (angel.sol#582-585)
isExcludedFromReward(address) should be declared external:
        - angel.isExcludedFromReward(address) (angel.sol#587-589)
totalFees() should be declared external:
        - angel.totalFees() (angel.sol#591-593)
minimumTokensBeforeSwapAmount() should be declared external:
        - angel.minimumTokensBeforeSwapAmount() (angel.sol#595-597)
buyBackSellLimitAmount() should be declared external:
        - angel.buyBackSellLimitAmount() (angel.sol#599-601)
deliver(uint256) should be declared external:
        - angel.deliver(uint256) (angel.sol#603-610)
reflectionFromToken(uint256,bool) should be declared external:
        - angel.reflectionFromToken(uint256,bool) (angel.sol#613-622)
```

```
excludeFromReward(address) should be declared external:
        - angel.excludeFromReward(address) (angel.sol#630-638)
isExcludedFromFee(address) should be declared external:
        - angel.isExcludedFromFee(address) (angel.sol#986-988)
excludeFromFee(address) should be declared external:
        - angel.excludeFromFee(address) (angel.sol#990-992)
includeInFee(address) should be declared external:
        - angel.includeInFee(address) (angel.sol#994-996)
GetBuyBackTimeInterval() should be declared external:
        - angel.GetBuyBackTimeInterval() (angel.sol#1042-1044)
GetSwapMinutes() should be declared external:
        - angel.GetSwapMinutes() (angel.sol#1055-1057)
setBuyBackEnabled(bool) should be declared external:
        - angel.setBuyBackEnabled(bool) (angel.sol#1106-1109)
setAutoBuyBackEnabled(bool) should be declared external:
        - angel.setAutoBuyBackEnabled(bool) (angel.sol#1111-1114)
changeRouterVersion(address) should be declared external:
        - angel.changeRouterVersion(address) (angel.sol#1134-1147)
transferForeignToken(address,address) should be declared external:
        - angel.transferForeignToken(address,address) (angel.sol#1153-1157)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:angel.sol analyzed (10 contracts with 75 detectors), 185 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Solidity Static Analysis

Angel.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address._functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 128:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in angel.(): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 521:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in angel.swapTokensForEth(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 807:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in angel.swapETHForTokens(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 827:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 696:94:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 697:40:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 711:44:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 731:90:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 838:12:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 855:12:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 1021:43:

Low level calls:

Use of "call": should be avoided whenever possible.
It can lead to unexpected behavior if return value is not handled properly.
Please use Direct Calls via specifying the called contract's interface.
more
Pos: 106:27:

Gas costs:

Gas requirement of function angel.balanceOf is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 561:4:

Gas costs:

Gas requirement of function angel.transfer is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 566:4:

Gas costs:

Gas requirement of function angel.allowance is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 571:4:

Gas costs:

Gas requirement of function angel.approve is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 575:4:

Solhint Linter

Angel.sol

```
Angel.sol:7:1: Error: Compiler version ^0.8.4 does not satisfy the r
semver requirement
Angel.sol:131:51: Error: Avoid to use low level calls.
Angel.sol:137:17: Error: Avoid to use inline assembly. It is acceptable
only in rare cases
Angel.sol:155:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity >=0.7.0)
Angel.sol:186:16: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:192:21: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:198:17: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:241:5: Error: Function name must be in mixedCase
Angel.sol:242:5: Error: Function name must be in mixedCase
Angel.sol:258:5: Error: Function name must be in mixedCase
Angel.sol:279:5: Error: Function name must be in mixedCase
Angel.sol:414:1: Error: Contract has 40 states declarations but allowed
no more than 15
Angel.sol:414:1: Error: Contract name must be in CamelCase
Angel.sol:489:5: Error: Explicitly mark visibility of state
Angel.sol:521:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity >=0.7.0)
Angel.sol:540:29: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:687:32: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:696:95: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:697:41: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:711:45: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:731:91: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:821:13: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:838:13: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:855:13: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:1021:44: Error: Avoid to make time-based decisions in your
business logic
Angel.sol:1043:5: Error: Function name must be in mixedCase
Angel.sol:1047:5: Error: Function name must be in mixedCase
Angel.sol:1051:5: Error: Function name must be in mixedCase
Angel.sol:1055:5: Error: Function name must be in mixedCase
Angel.sol:1059:5: Error: Function name must be in mixedCase
Angel.sol:1064:5: Error: Function name must be in mixedCase
Angel.sol:1068:5: Error: Function name must be in mixedCase
```

```
Angel.sol:1159:32: Error: Code contains empty blocks
Angel.sol:1168:5: Error: Function name must be in mixedCase
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

Email: audit@EtherAuthority.io