Techrate1 Techrate

TechRate

AUDIT COMPANY

Pawthereum
Smart Contract Security Audit

TechRate

October, 2021

Audit Details

Audited project

Pawthereum

Deployer address

0x8EA6cC82148D92F979D34031Bdba60CCD35b0f9e

Client contacts:

Pawthereum team

Blockchain

Ethereum

Project website:

https://pawthereum.com

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

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

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

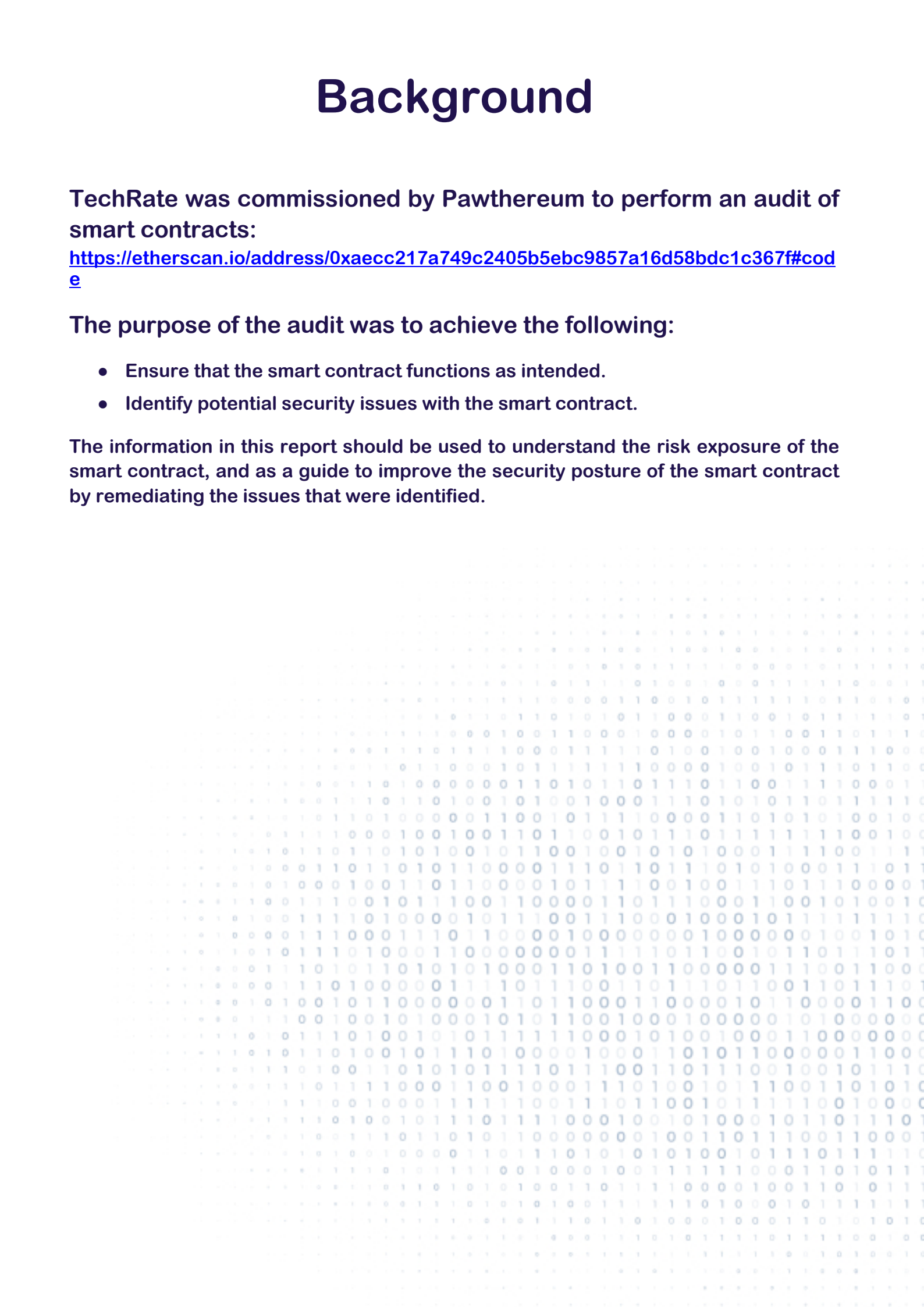
TechRate was commissioned by Pawthereum to perform an audit of smart contracts:

https://etherscan.io/address/0xaecc217a749c2405b5ebc9857a16d58bdc1c367f#code

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.



Contracts Details

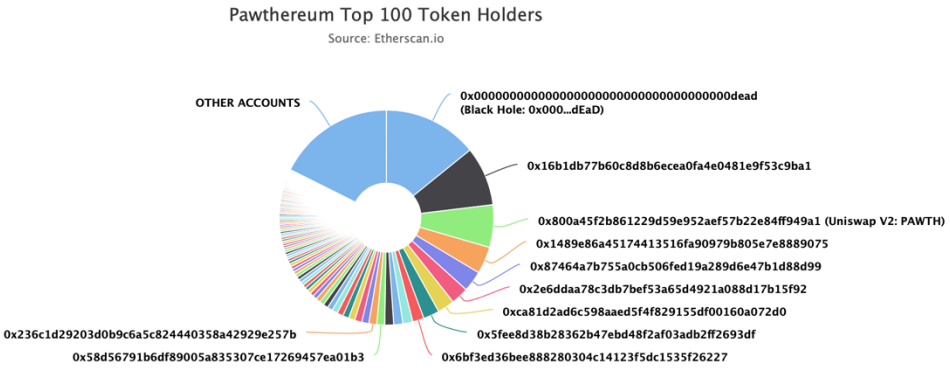
Token contract details for 24.10.2021

Contract name	Pawthereum
Contract address	0xAEcc217a749c2405b5ebC9857a16d58Bdc1c367F
Total supply	1,000,000,000
Token ticker	PAWTH
Decimals	9
Token holders	1,076
Transactions count	2,139
Top 100 holders dominance	82.36%
Tax fee	2
Charity fee	2
Marketing fee	0
Uniswap V2 pair	0x800A45f2b861229d59E952aeF57B22e84Ff949A1
Contract deployer address	0x8EA6cC82148D92F979D34031Bdba60CCD35b0f9e
Contract's current owner address	0xCd7cfD6326A7854E6FABF262C3258f5fd136CD7A

Pawthereum Token Distribution

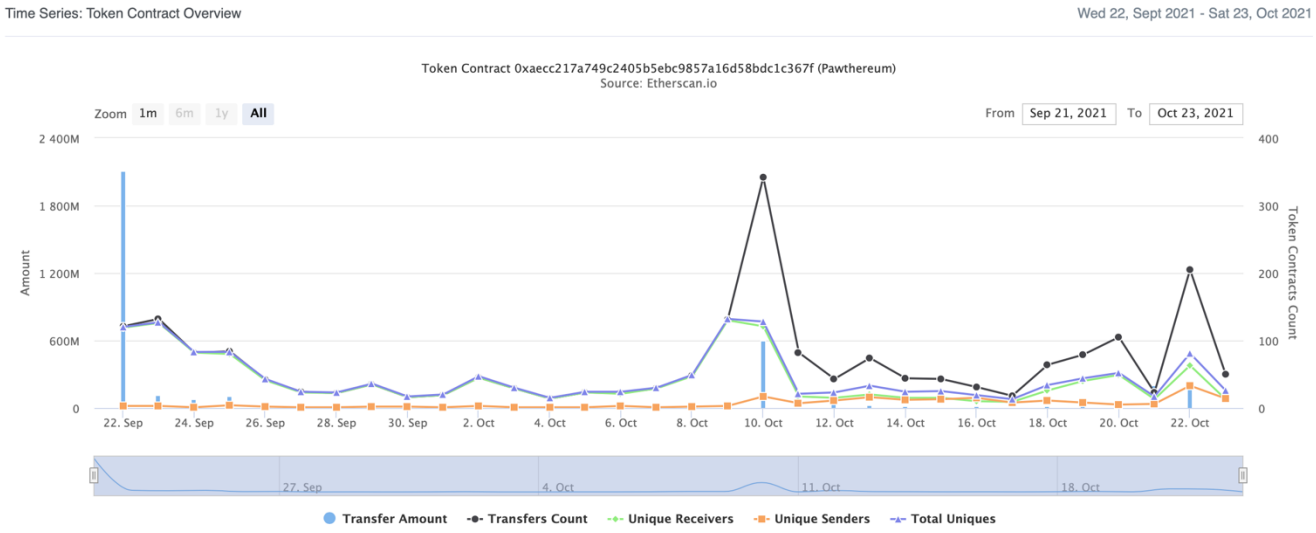
The top 100 holders collectively own 82.36% (823,595,027.51 Tokens) of Pawthereum

Token Total Supply: 1,000,000,000.00 Token | Total Token Holders: 1,076



(A total of 823,595,027.51 tokens held by the top 100 accounts from the total supply of 1,000,000,000.00 token)

Pawthereum Contract Interaction Details



Pawthereum Top 10 Token Holders

Rank	Address	Quantity	Percentage
1	Black Hole: 0x000...dEaD	142,000,397.444422449	14.2000%
2	0x16b1db77b60c8d8b6ecea0fa4e0481e9f53c9ba1	88,856,083.517632501	8.8856%
3	Uniswap V2: PAWTH	63,980,088.304046631	6.3980%
4	0x1489e86a45174413516fa90979b805e7e8889075	40,795,970.613462337	4.0796%
5	bacchus.eth	31,351,459.695692302	3.1351%
6	0x2e6ddaa78c3db7bef53a65d4921a088d17b15f92	26,366,946.920634911	2.6367%
7	0xca81d2ad6c598aaed5f4f829155df00160a072d0	25,850,786.746937485	2.5851%
8	0x5fee8d38b28362b47ebd48f2af03adb2ff2693df	24,136,698.319274315	2.4137%
9	0x6bf3ed36bee888280304c14123f5dc1535f26227	17,168,513.946536505	1.7169%
10	pawther.eth	14,848,270.618359756	1.4848%

Pawthereum LP Token Holders

Rank	Address	Quantity	Percentage
1	Unicrypt : Liquidity Lockers	2.222544487743720193	98.7094%
2	0xa7577f841d95b1331954c936d71fe45ba2f62fe5	0.022449944320643638	0.9971%
3	Pawthereum: PAWTH Token	0.006610202076080365	0.2936%
4	Black Hole: 0x000…000	0.000000000000001	0.0000%

Contract functions details

+ Context
  - [Int] _msgSender
  - [Int] _msgData

+ [Int] IERC20
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] div
  - [Int] mod
  - [Int] mod

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Prv] _functionCallWithValue #

+ Ownable (Context)
  - [Int] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner

+ [Int] IUniswapV2Factory
  - [Ext] createPair #

+ [Int] IUniswapV2Pair
  - [Ext] sync #

+ [Int] IUniswapV2Router01
  - [Ext] factory
  - [Ext] WETH
  - [Ext] addLiquidity #
  - [Ext] addLiquidityETH ($)

+ [Int] IUniswapV2Router02 (IUniswapV2Router01)

- [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
- [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #
- [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
- [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)

+ Pawthereum (Context, IERC20, Ownable)
  - [Pub] <Constructor> #
  - [Ext] init #
    - modifiers: onlyOwner
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] transfer #
  - [Pub] allowance
  - [Pub] approve #
  - [Pub] transferFrom #
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Pub] isExcluded
  - [Pub] reflectionFromToken
  - [Pub] tokenFromReflection
  - [Ext] excludeAccount #
    - modifiers: onlyOwner
  - [Ext] includeAccount #
    - modifiers: onlyOwner
  - [Prv] _approve #
  - [Prv] _transfer #
  - [Prv] collectFee #
  - [Prv] _getReflectionRate
  - [Prv] swapAndLiquify #
    - modifiers: lockTheSwap
  - [Prv] swapTokensForEth #
  - [Prv] addLiquidity #
  - [Ext] setPair #
    - modifiers: onlyOwner
  - [Ext] setMarketingWallet #
    - modifiers: onlyOwner
  - [Ext] setCharityWallet #
    - modifiers: onlyOwner
  - [Ext] setTaxless #
    - modifiers: onlyOwner
  - [Ext] setSwapAndLiquifyEnabled #
    - modifiers: onlyOwner
  - [Ext] setTaxActive #
    - modifiers: onlyOwner
  - [Ext] setTaxFee #
    - modifiers: onlyOwner
  - [Ext] setBurnFee #
    - modifiers: onlyOwner
  - [Ext] setLiquidityFee #
    - modifiers: onlyOwner
  - [Ext] setMarketingFee #
    - modifiers: onlyOwner

- [Ext] setCharityFee #
  - modifiers: onlyOwner
- [Ext] setMaxTxAmount #
  - modifiers: onlyOwner
- [Ext] setMinTokensBeforeSwap #
  - modifiers: onlyOwner
- [Ext] <Fallback> ($)

($) = payable function
= non-constant function

Issues Checking Status

	Issue description	Checking status
1.	Compiler errors.	Passed
2.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3.	Possible delays in data delivery.	Passed
4.	Oracle calls.	Passed
5.	Front running.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow.	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Low issues
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Low issues
12.	The impact of the exchange rate on the logic.	Passed
13.	Private user data leaks.	Passed
14.	Malicious Event log.	Passed
15.	Scoping and Declarations.	Passed
16.	Uninitialized storage pointers.	Passed
17.	Arithmetic accuracy.	Passed
18.	Design Logic.	Passed
19.	Cross-function race conditions.	Passed
20.	Safe Open Zeppelin contracts implementation and usage.	Passed
21.	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

1. Out of gas

Issue:

- The function `includeAccount()` uses the loop to find and remove addresses from the `_excluded` list. Function will be aborted with `OUT_OF_GAS` exception if there will be a long excluded addresses list.

```
function includeAccount(address account) external onlyOwner {
    require(_isExcluded[account], "ERC20: Account is already included");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tokenBalance[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- The function `_getReflectionRate()` also uses the loop for evaluating total supply. It also could be aborted with `OUT_OF_GAS` exception if there will be a long excluded addresses list.

```
function _getReflectionRate() private view returns (uint256) {
    uint256 reflectionSupply = _reflectionTotal;
    uint256 tokenSupply = _tokenTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (
            _reflectionBalance[_excluded[i]] > reflectionSupply ||
            _tokenBalance[_excluded[i]] > tokenSupply
        ) return _reflectionTotal.div(_tokenTotal);
        reflectionSupply = reflectionSupply.sub(
            _reflectionBalance[_excluded[i]]
        );
        tokenSupply = tokenSupply.sub(_tokenBalance[_excluded[i]]);
    }
    if (reflectionSupply < _reflectionTotal.div(_tokenTotal))
        return _reflectionTotal.div(_tokenTotal);
    return reflectionSupply.div(tokenSupply);
}
```

Recommendation:

Check that the excluded array length is not too big.

2. Wrong reflection from token calculations

Issue:

- Missing parentheses when calculating target value.

```
tokenAmount
        .sub(tokenAmount.mul(_taxFee).div(10**(_feeDecimal + 2)))
        .mul(_getReflectionRate());
```

```
function reflectionFromToken(uint256 tokenAmount, bool deductTransferFee)
    public
    view
    returns (uint256)
{
    require(tokenAmount <= _tokenTotal, "Amount must be less than supply");
    if (!deductTransferFee) {
        return tokenAmount.mul(_getReflectionRate());
    } else {
        return
            tokenAmount
                .sub(tokenAmount.mul(_taxFee).div(10**_feeDecimal + 2))
                .mul(_getReflectionRate());
    }
}
```

Owner privileges (In the period when the owner is not renounced)

- Owner can include in or exclude addresses from reflection.

```
function excludeAccount(address account) external onlyOwner {
    require(
        account != address(uniswapV2Router),
        "ERC20: We can not exclude Uniswap router."
    );
    require(!_isExcluded[account], "ERC20: Account is already excluded");
    if (_reflectionBalance[account] > 0) {
        _tokenBalance[account] = tokenFromReflection(
            _reflectionBalance[account]
        );
    }
    _isExcluded[account] = true;
    _excluded.push(account);
}

function includeAccount(address account) external onlyOwner {
    require(_isExcluded[account], "ERC20: Account is already included");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tokenBalance[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- Owner can update Uniswap pair.

```
function setPair(address pair) external onlyOwner {
    uniswapV2Pair = pair;
}
```

- Owner can change marketing and charity wallet addresses.

```
function setMarketingWallet(address account) external onlyOwner {
    marketingWallet = account;
}

function setCharityWallet(address account) external onlyOwner {
    charityWallet = account;
}
```

- Owner can include in or exclude from the taxes.

```
function setTaxless(address account, bool value) external onlyOwner {
    isTaxless[account] = value;
}
```

- Owner can enable / disable swap and liquify.

```
function setSwapAndLiquifyEnabled(bool enabled) external onlyOwner {
    swapAndLiquifyEnabled = enabled;
    SwapAndLiquifyEnabledUpdated(enabled);
}
```

- Owner can enable / disable taxes.

```
function setTaxActive(bool value) external onlyOwner {
    isTaxActive = value;
}
```

- Owner can change the tax, burn, liquidity, marketing and charity fee.

```
function setTaxFee(uint256 fee) external onlyOwner {
    require(fee <= 200, "You can't set reflections fee above 2 percent.");
    _taxFee = fee;
}

function setBurnFee(uint256 fee) external onlyOwner {
    require(fee <= 200, "You can't set burn fees above 2 percent.");
    _burnFee = fee;
}

function setLiquidityFee(uint256 fee) external onlyOwner {
    require(fee <= 200, "You can't set this fee above 2 percent.");
    _liquidityFee = fee;
}

function setMarketingFee(uint256 fee) external onlyOwner {
    require(fee <= 200, "You can't set the marketing fee above 2 percent.");
    _marketingFee = fee;
}

function setCharityFee(uint256 fee) external onlyOwner {
    require(fee <= 200, "You can't set the charity fee above 2 percent.");
    _charityFee = fee;
}
```

- Owner can change the maximum transaction amount.

```
function setMaxTxAmount(uint256 amount) external onlyOwner {
    maxTxAmount = amount;
}
```

- Owner can change minimum amount of tokens needed to swap.

```
function setMinTokensBeforeSwap(uint256 amount) external onlyOwner {
    minTokensBeforeSwap = amount;
}
```

Conclusion

Smart contracts contain low severity issues and owner privileges! Liquidity pair contract's security is not checked due to out of scope.

Liquidity locking details provided by the team:
https://app.unicrypt.network/amm/uni-v2/pair/0x800a45f2b861229d59e952aef57b22e84ff949a1

TechRate note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.