SMART CONTRACT

SECURITY AUDIT

JULY, 2021

AUDIT DETAILS

Audited Project
WaterWellProject

Deployer Address
0x27F0fDF3f86D96afaC697E7cd5F1Bd1C709e6093

Client Contacts
WaterWellProject Team

BlockChain
Binance Smart Chain

Project Website
https://waterwell-74f989.ingress-erytho.easywp.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 SyloShield and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SyloShield) owe no duty of care towards you or any other person, nor does SyloShield 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 SyloShield 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, SyloShield hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SyloShield, 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

SyloShield was commissioned by WaterWellProject to perform an audit of smart contracts:

https://bscscan.com/address/0x27F0fDF3f86D96afaC697E7cd5F1Bd1C709e6093#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.**

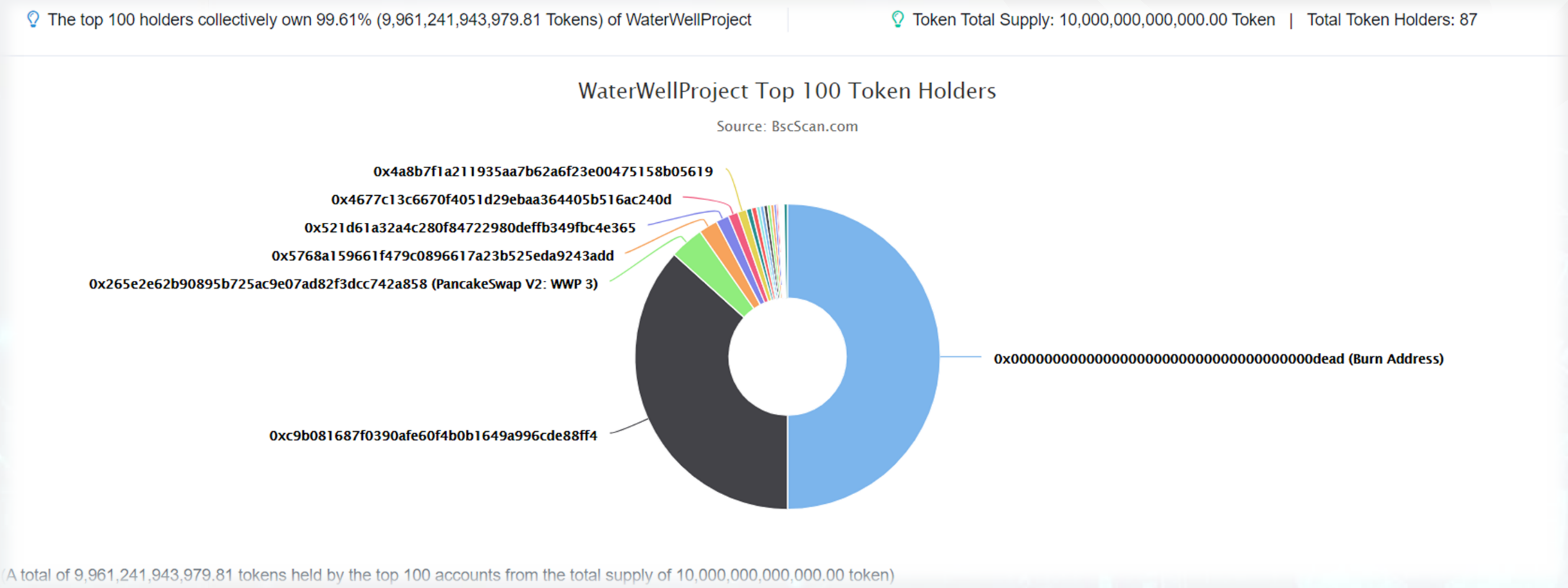
The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

CONTRACTS DETAILS

Token contracts details for 02.07.2021

Contract name	**WaterWellProject**
Contract address	0x27F0fDF3f86D96afaC697E7cd5F1Bd1C709e6093
Total supply	10,000,000,000,000
Token ticker	WWP
Decimals	9
Token holders	87
Transaction count	1,501
Top 100 holders dominance	99.61%
Donation fee	2
Liquidity fee	4
Tax fee	4
Contract deployer address	0x9E9CFA33E9fBEeD3a83d17C58dDa1Fc01D1AcD87
Contract’s current owner address	0xc9b081687f0390AfE60f4B0B1649A996cde88FF4

WaterWellProject Token Distribution


The top 100 holders collectively own 99.61% (9,961,241,943,979.81 Tokens) of WaterWellProject

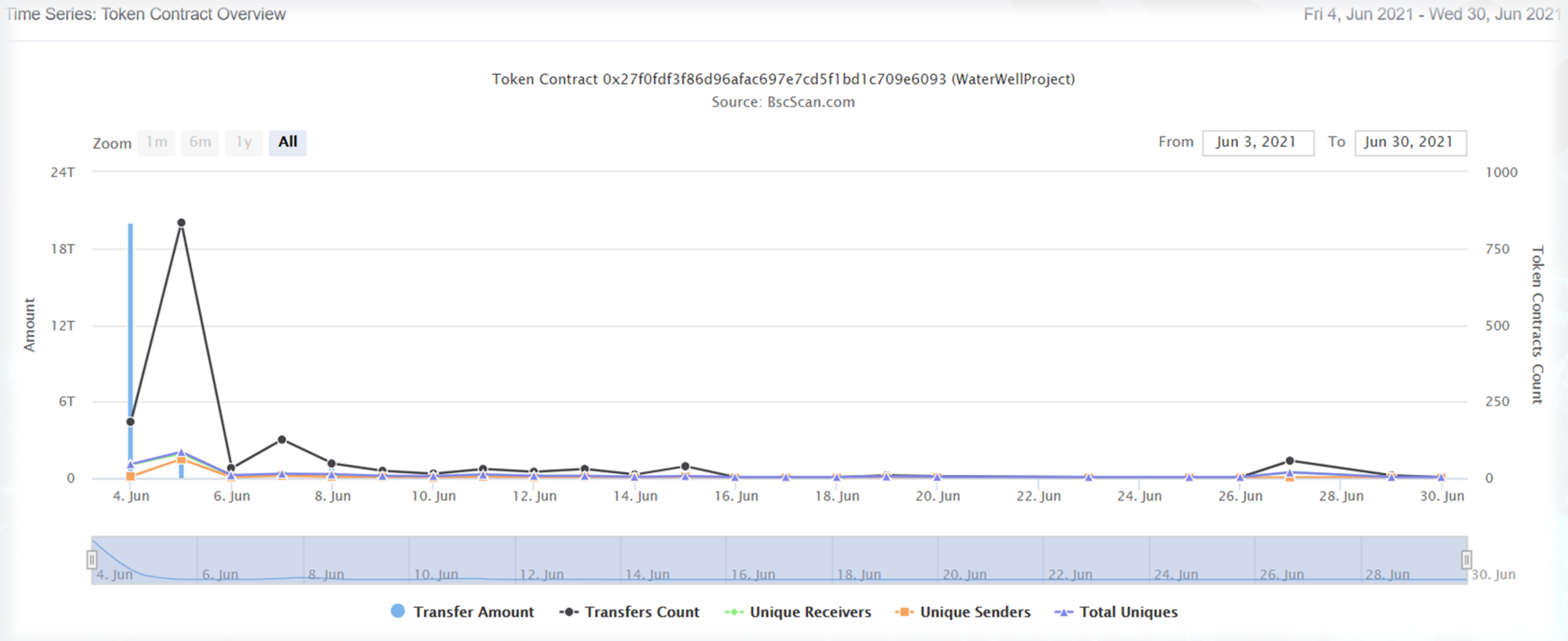
Token Total Supply: 10,000,000,000,000.00 Token | Total Token Holders: 87

WaterWellProject Top 100 Token Holders

Source: BscScan.com

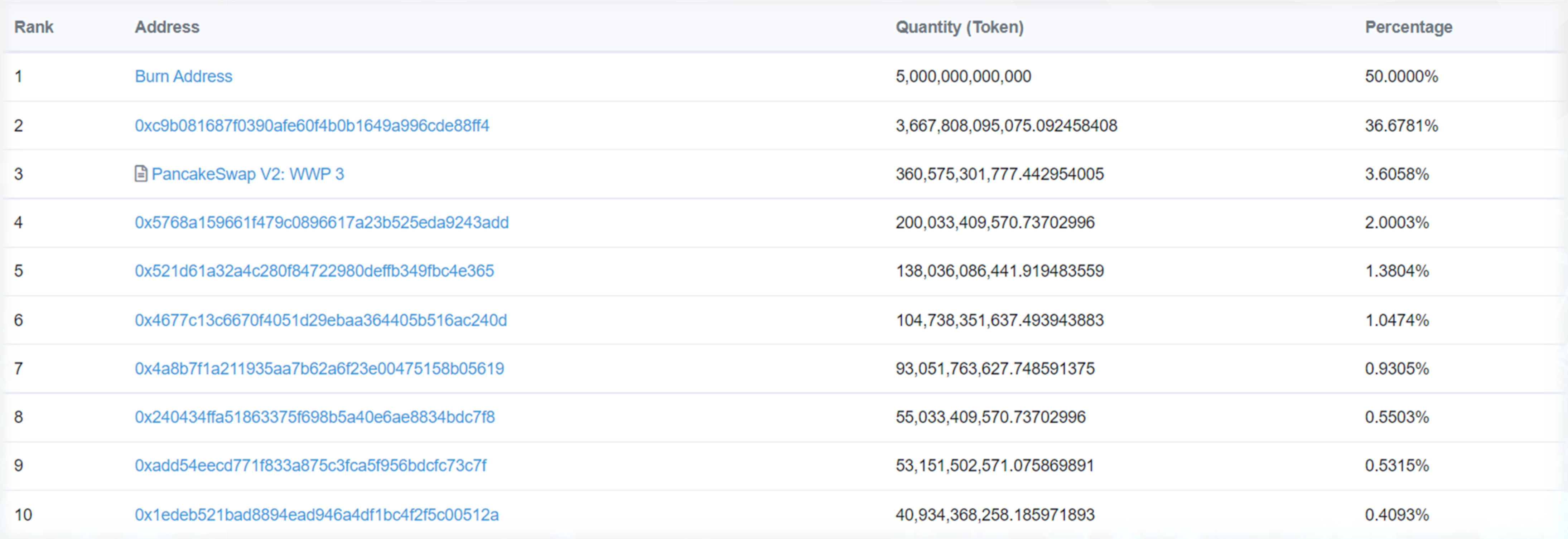
(A total of 9,961,241,943,979.81 tokens held by the top 100 accounts from the total supply of 10,000,000,000,000.00 token)

WaterWellProject Contract Interaction Details



SYLOSHIELD

WaterWellProject Top 10 Token Holders



Rank	Address	Quantity (Token)	Percentage
1	Burn Address	5,000,000,000,000	50.0000%
2	0xc9b081687f0390afe60f4b0b1649a996cde88ff4	3,667,808,095,075.092458408	36.6781%
3	PancakeSwap V2: WWP 3	360,575,301,777.442954005	3.6058%
4	0x5768a159661f479c0896617a23b525eda9243add	200,033,409,570.73702996	2.0003%
5	0x521d61a32a4c280f84722980deffb349fbc4e365	138,036,086,441.919483559	1.3804%
6	0x4677c13c6670f4051d29ebaa364405b516ac240d	104,738,351,637.493943883	1.0474%
7	0x4a8b7f1a211935aa7b62a6f23e00475158b05619	93,051,763,627.748591375	0.9305%
8	0x240434ffa51863375f698b5a40e6ae8834bdc7f8	55,033,409,570.73702996	0.5503%
9	0xadd54eecd771f833a875c3fca5f956bdcfc73c7f	53,151,502,571.075869891	0.5315%
10	0x1edeb521bad8894ead946a4df1bc4f2f5c00512a	40,934,368,258.185971893	0.4093%

Contract Functions Details

```
+ [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

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

+ [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
  - [Pub] geUnlockTime
  - [Pub] lock #
  - modifiers: onlyOwner
  - [Pub] unlock #
```

Contract Functions Details

+ [Int] IUniswapV2Router01
  - [Ext] factory
  - [Ext] WETH
  - [Ext] addLiquidity #
  - [Ext] addLiquidityETH ($)
  - [Ext] removeLiquidity #
  - [Ext] removeLiquidityETH #
  - [Ext] removeLiquidityWithPermit #
  - [Ext] removeLiquidityETHWithPermit #
  - [Ext] swapExactTokensForTokens #
  - [Ext] swapTokensForExactTokens #
  - [Ext] swapExactETHForTokens ($)
  - [Ext] swapTokensForExactETH #
  - [Ext] swapExactTokensForETH #
  - [Ext] swapETHForExactTokens ($)
  - [Ext] quote
  - [Ext] getAmountOut
  - [Ext] getAmountIn
  - [Ext] getAmountsOut
  - [Ext] getAmountsIn

Contract Functions Details

```
+ [Int] IUniswapV2Factory
   - [Ext] feeTo
   - [Ext] feeToSetter
   - [Ext] getPair
   - [Ext] allPairs
   - [Ext] allPairsLength
   - [Ext] createPair #
   - [Ext] setFeeTo #
  - [Ext] setFeeToSetter #

+ [Int] IUniswapV2Router02 (IUniswapV2Router01)
   - [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
   - [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #
   - [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
   - [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
   - [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #
```

Contract Functions Details

```
+ [Int] IUniswapV2Pair
   - [Ext] name
   - [Ext] symbol
   - [Ext] decimals
   - [Ext] totalSupply
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transfer #
   - [Ext] transferFrom #
   - [Ext] DOMAIN_SEPARATOR
   - [Ext] PERMIT_TYPEHASH
   - [Ext] nonces
   - [Ext] permit
   - [Ext] MINIMUM_LIQUIDITY
   - [Ext] factory
   - [Ext] token0
   - [Ext] token1
   - [Ext] getReserves #
   - [Ext] price0CumulativeLast #
   - [Ext] price1CumulativeLast #
   - [Ext] kLast #
   - [Ext] mint #
   - [Ext] burn #
   - [Ext] swap
   - [Ext] skim
   - [Ext] sync
   - [Ext] initialize
```

Contract Functions Details

```
+ WaterWellProject (Context, IERC20, Ownable)
  - [Pub] <Constructor> #
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] transfer #
  - [Pub] allowance
  - [Pub] approve #
  - [Pub] transferFrom #
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Pub] isExcludedFromReward
  - [Pub] totalFees
  - [Pub] totalDonationFees
  - [Pub] blacklistAddress
  - [Pub] deliver #
  - [Pub] reflectionFromToken
  - [Pub] tokenFromReflection
  - [Pub] excludeFromReward #
  - modifiers: onlyOwner
  - [Ext] includeInReward #
  - modifiers: onlyOwner
  - [Ext] controlLiquidityBalanceAfterVault
  - modifiers: onlyOwner
  - [Prv] _transferBothExcluded #
  - [Pub] excludeFromFee #
  - modifiers: onlyOwner
  - [Pub] includeInFee #
  - modifiers: onlyOwner
  - [Pub] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner
  - [Pub] isExcludedFromFee
```

Contract Functions Details

- [Ext] setTaxFeePercent #
- modifiers: onlyOwner
- [Ext] setLiquidityFeePercent #
- modifiers: onlyOwner
- [Ext] setDonationFeePercent #
- modifiers: onlyOwner
- [Ext] setMaxTxPercent #
- modifiers: onlyOwner
- [Ext] setAsDonationAddress #
- modifiers: onlyOwner
- [Ext] <Fallback> ($)
- [Prv] _reflectFee #
- [Prv] _getValues
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate

($) = payable function
# = non-constant function

- [Prv] _getCurrentSupply
- [Prv] _takeLiquidity #
- [Prv] calculateTaxFee
- [Prv] calculateLiquidityFee
- [Prv] calculateDonationFee
- [Prv] removeAllFee #
- [Prv] restoreAllFee #
- [Prv] _approve #
- [Prv] _transfer #
- [Prv] swapAndLiquify #
- modifiers: lockTheSwap
- [Prv] swapTokensForEth #
- [Prv] addLiquidity #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #
- [Prv] _transferToExcluded #
- [Prv] _transferFromExcluded #

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	Passed

ISSUES CHECKING STATUS

	Issue Description	Checking status
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.

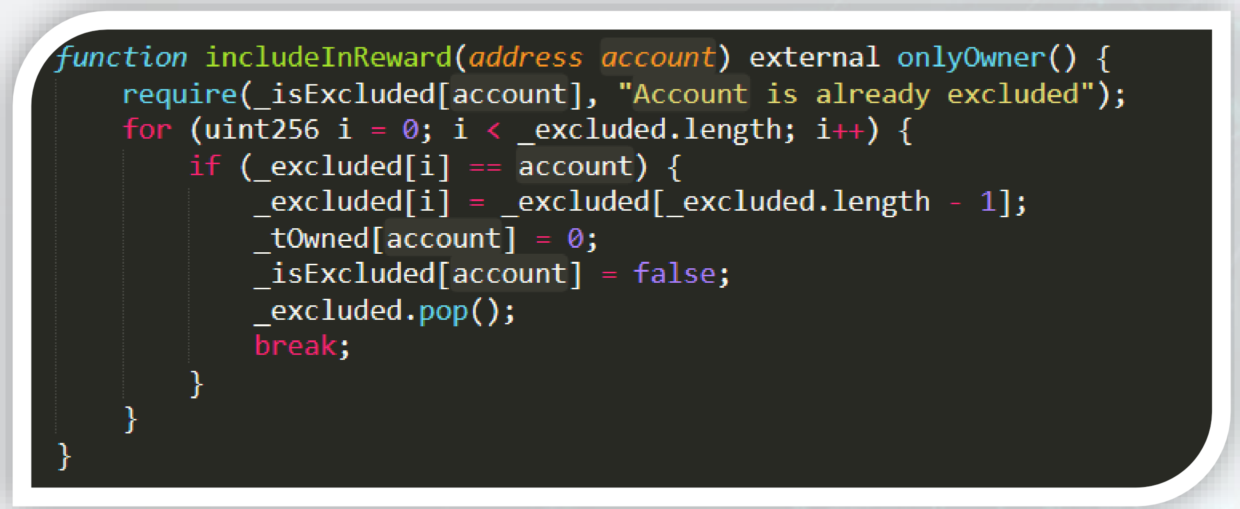
Security Issues

Low Severity Issues

1. Out of gas

Issue:

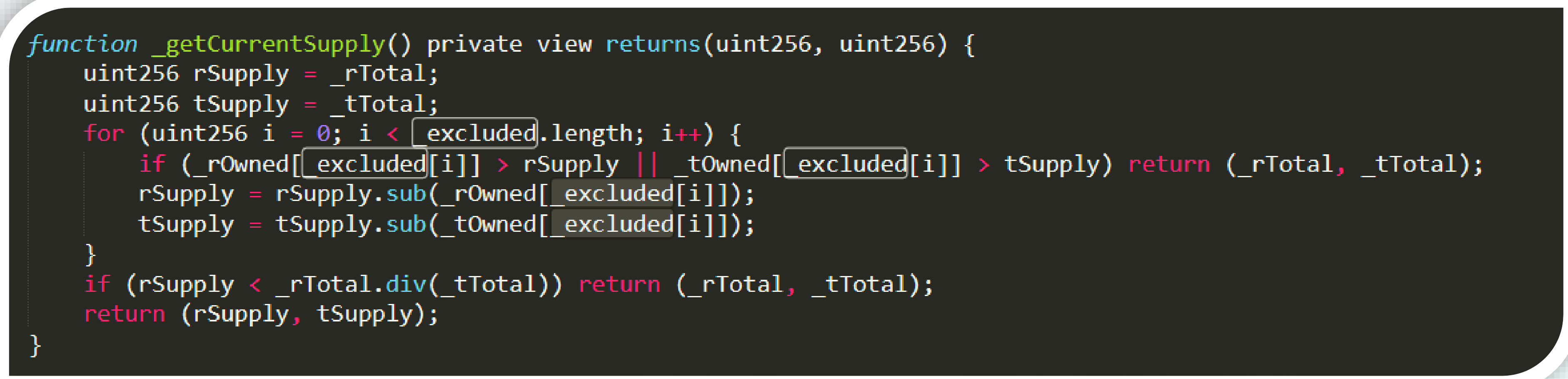
- **The function includeInReward() uses the loop to find and remove addresses from the _excluded list. Function will be aborted with OUT_OF_GAS exception if there will be a long excluded addresses list.**



```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already 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;
        }
    }
}
```

Security Issues

- The function _getCurrentSupply 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 _getCurrentSupply() private view returns(uint256, uint256) {
    uint256 rSupply = _rTotal;
    uint256 tSupply = _tTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return (_rTotal, _tTotal);
        rSupply = rSupply.sub(_rOwned[_excluded[i]]);
        tSupply = tSupply.sub(_tOwned[_excluded[i]]);
    }
    if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
    return (rSupply, tSupply);
}
```

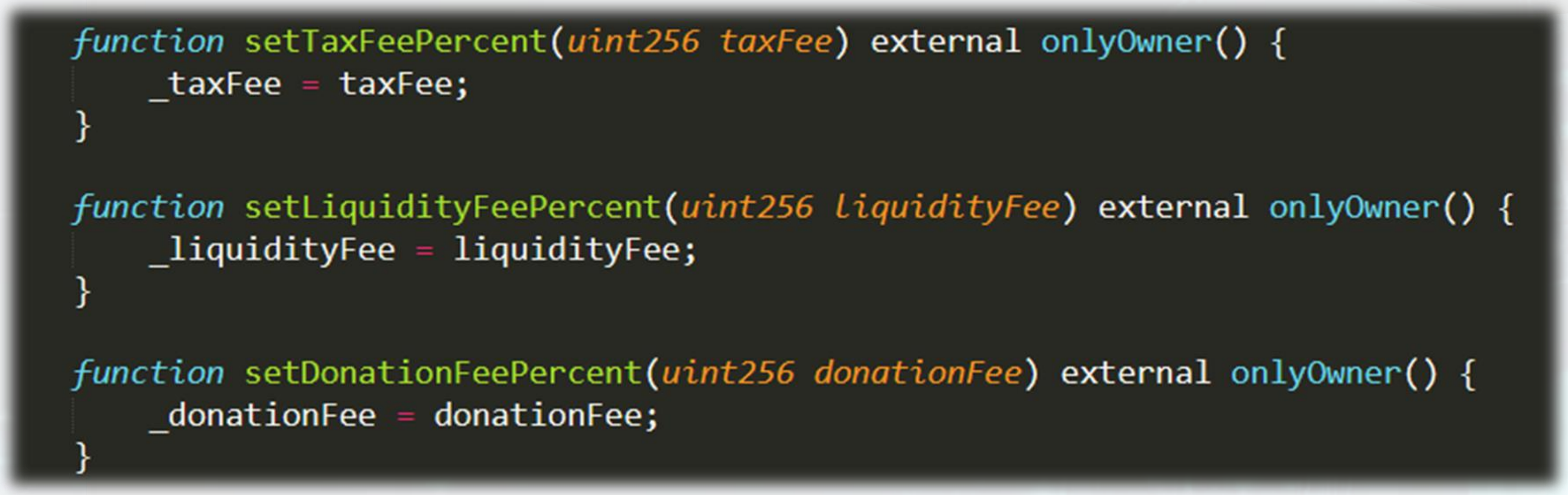
Recommendation:

Check that the excluded array length is not too big.

Security Issues

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

- Owner can change the tax, liquidity and donation fee.



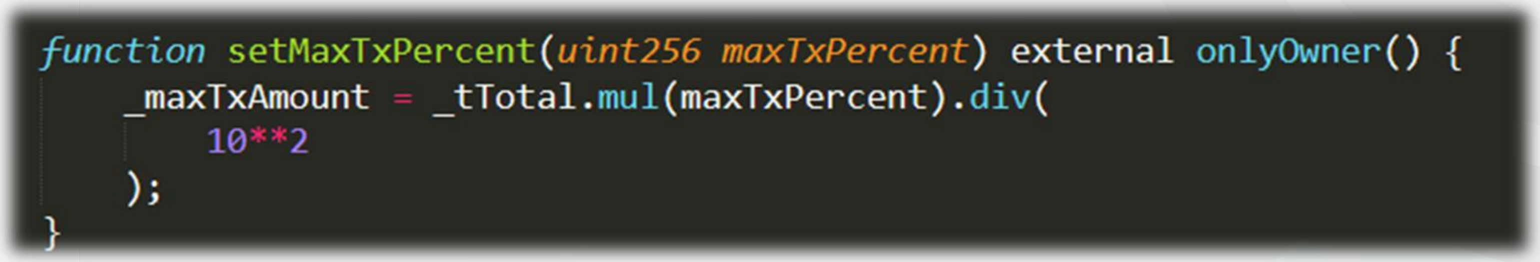
```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
    _taxFee = taxFee;
}

function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
    _liquidityFee = liquidityFee;
}

function setDonationFeePercent(uint256 donationFee) external onlyOwner() {
    _donationFee = donationFee;
}
```

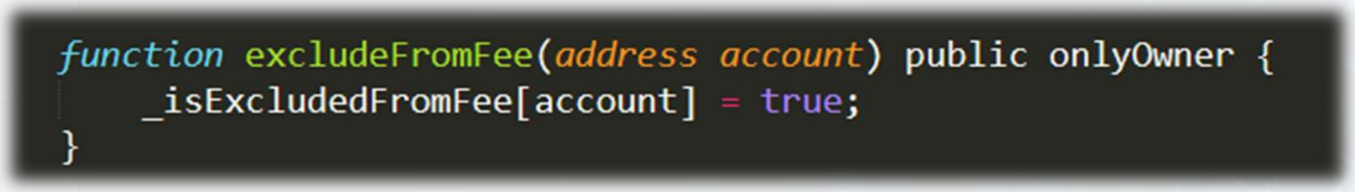
Security Issues

- Owner can change the maximum transaction amount.



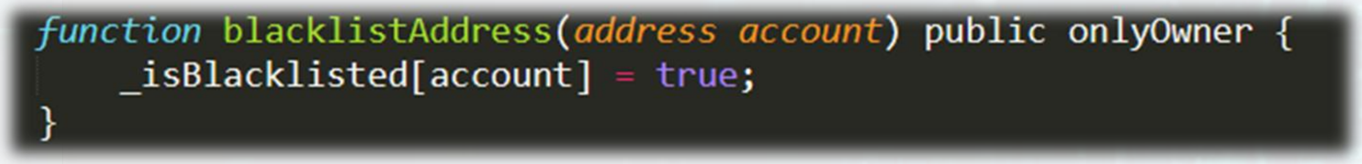
```
function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner() {
    _maxTxAmount = _tTotal.mul(maxTxPercent).div(
        10**2
    );
}
```

- Owner can exclude from the fee.



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

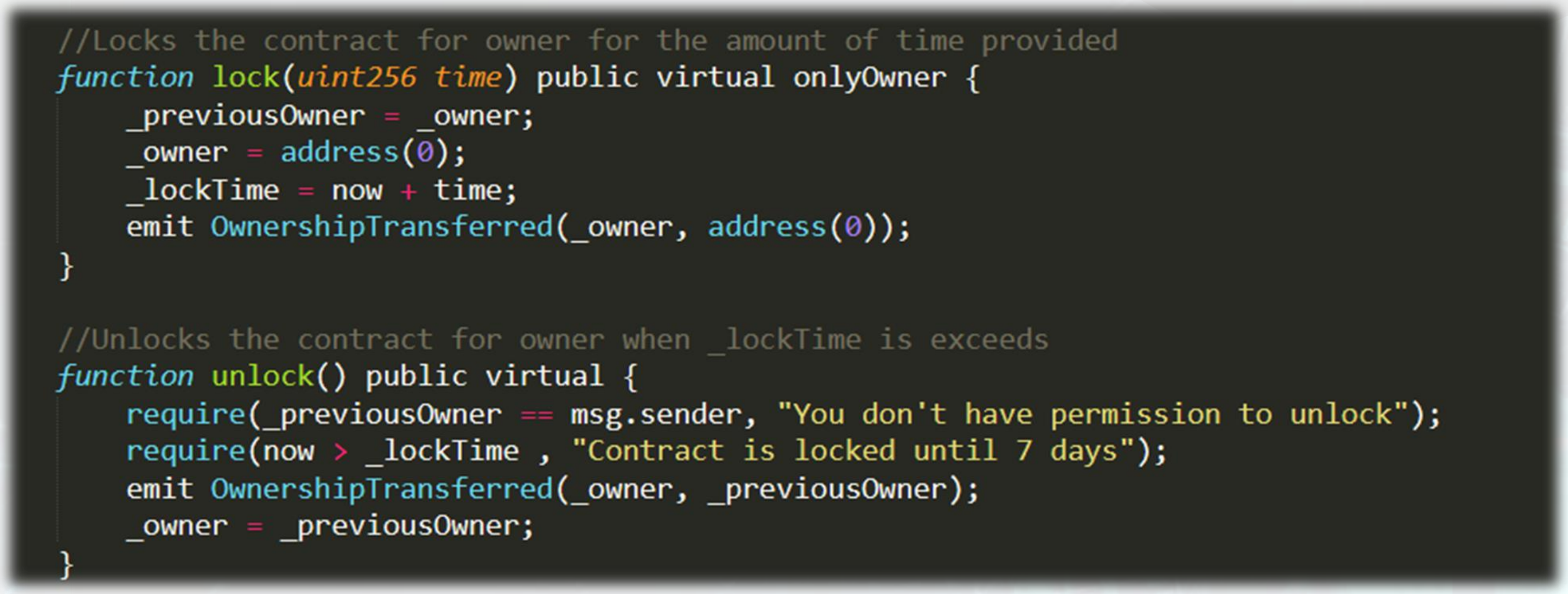
- Owner can add any address to blacklist



```
function blacklistAddress(address account) public onlyOwner {
    _isBlacklisted[account] = true;
}
```

Security Issues

- Owner can lock and unlock. By the way, using these functions the owner could retake privileges even after the ownership was renounced.



```
//Locks the contract for owner for the amount of time provided
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = now + time;
    emit OwnershipTransferred(_owner, address(0));
}

//Unlocks the contract for owner when _lockTime is exceeds
function unlock() public virtual {
    require(_previousOwner == msg.sender, "You don't have permission to unlock");
    require(now > _lockTime , "Contract is locked until 7 days");
    emit OwnershipTransferred(_owner, _previousOwner);
    _owner = _previousOwner;
}
```

Conclusion

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

Liquidity locking details NOT provided by the team.

SyloShield 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

Syloshield Syloshield Syloshield_audits