0x Guard

Smart contracts security assessment

Final report

Tariff: Top

Platzy

April 2024

0xguard.com

hello@0xguard.com

Contents

Introduction

The report has been prepared for **Platzy**.

The PLATZY project is represented by the ERC-20 tax token PLATZY and the claim-based airdrop contract PLATZYAirdrop with vesting features.

The md5 sum of the files under investigation:

c821af48d51039ecdbd622daafbf8d52 - PLATZY.sol

1a4f869deba6ec242d708c75b2289513 - PLATZYAirdrop.sol

Report Update

The contracts code was updated according to this report.

The md5 sum of the updated files:

5bbe2effee96fa5fbe9d331593285d30 - PLATZY.sol

14c5ed509e2773a21fee510cb6705f9e - PLATZYAirdrop.sol

Name	Platzy
Audit date	2024-04-19 - 2024-04-20
Language	Solidity
Platform	Base Chain

Contracts checked

Name	Address
PLATZY	
PLATZYAirdrop	

Procedure

We perform our audit according to the following procedure:

Automated analysis

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
- Manual verification (reject or confirm) all the issues found by the tools

Manual audit

- Manually analyze smart contracts for security vulnerabilities
- Smart contracts' logic check

Known vulnerabilities checked

Title	Check result
Unencrypted Private Data On-Chain	passed
Code With No Effects	passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	passed
Presence of unused variables	passed
Incorrect Inheritance Order	passed
Requirement Violation	passed
Weak Sources of Randomness from Chain Attributes	passed
Shadowing State Variables	passed

Incorrect Constructor Name	passed
Block values as a proxy for time	passed
Authorization through tx.origin	passed
DoS with Failed Call	passed
Delegatecall to Untrusted Callee	passed
Use of Deprecated Solidity Functions	passed
Assert Violation	passed
State Variable Default Visibility	passed
Reentrancy	passed
Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	passed
Outdated Compiler Version	passed
Integer Overflow and Underflow	passed
Function Default Visibility	passed

Classification of issue severity

High severity	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
Medium severity	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.

Low severity	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

Issues

High severity issues

No issues were found

Medium severity issues

1. Zero initial claim amount (PLATZYAirdrop)

Status: Fixed

Due to the division being performed before multiplication on L130, the value of `_initial` variable will always be early zero. This is because the result of dividing numbers from 0 to 99 by 100 will be equal to zero.

As a result, this will lead to the contract's behavior not corresponding to the intended logic, where the user is supposed to be credited with the initial amount of tokens when calling the `claim()` function.

```
uint256 _initial = _amount * (_vested / 100);
```

Recommendation: Perform multiplication before division.

Low severity issues

1. Lack of events (PLATZY)

Status: Fixed

We recommend adding events for the `removeLimits()` function to easily track changes off-chain.

2. Lack of validation of constructor parameters (PLATZY)
Status: Fixed

1. In the contract constructor, variables are assigned that will not be changed later on (`buyTax`, `sellTax`, `burnPercent`, `lpPercent`).

We recommend adding validation of these constructor parameters or declaring such variables as constants. This will help avoid incorrect initialization of contract parameters during deployment.

2. Additionally, consider adding validation for the values of `_shares`: `_shares[i] < 100`, as well as validation for the lengths of arrays for `_payees` and `_shares`.

3. Order of mathematical operations (PLATZY)
Status: Fixed

In the `_transfer()` function and the contract `constructor`, division is performed before multiplication in many parts of the code. This can lead to loss of precision or zeroing of calculation results.

We strongly recommend checking the logic of calculations on L70, L71, L111, L114, L121, L123, L133, and ensuring that the calculations are performed as intended.

4. Gas optimization (PLATZY)
Status: Fixed

1. The variables `buyTax`, `sellTax`, `burnPercent`, `lpPercent`, `maxSupply`, `taxSwapThreshold`, `maxTaxSwap`, `supplyWallet` can be declared as `immutable`.

2. Consider adding check `if (toSend > 0) {perform transfer to peyees}` into the `_transfer()` function to avoid empty transfers.

3. We recommend declaring global variable `WETH` with initialization in the contract constructor or in the `createPair()` function. Using of such global variable on L172 (for `path[1]`) instead of external call allows to decrease cost of each swap.

Conclusion

Platzy PLATZY, PLATZYAirdrop contracts were audited. 1 medium, 4 low severity issues were found.
1 medium, 4 low severity issues have been fixed in the update.

We strongly recommend writing unit tests to have extensive coverage of the codebase minimize the possibility of bugs and ensure that everything works as expected.

The contract owner of the PLATZYAirdrop contract can disable the creation of new vestings at any time (by setting a new merkle-root or disabling the `_claimIsActive` value). Users interacting with the contract have to trust the owner.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability)set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without 0xGuard prior written consent.

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts 0xGuard to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Slither output

```
INFO:Detectors:
PLATZYAirdrop.claim(address,uint256,bytes32[]) (contracts/PLATZYAirdrop.sol#114-135)
ignores return value by IERC20(_token).transfer(_address,_initial) (contracts/
PLATZYAirdrop.sol#132)
PLATZYAirdrop.release(address) (contracts/PLATZYAirdrop.sol#160-171) ignores return
value by IERC20(_token).transfer(_address,vestedAmount) (contracts/
PLATZYAirdrop.sol#169)
PLATZYAirdrop.withdraw(uint256) (contracts/PLATZYAirdrop.sol#216-219) ignores return
value by IERC20(_token).transfer(msg.sender,amount) (contracts/PLATZYAirdrop.sol#218)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer
INFO:Detectors:
PLATZY.constructor(string,string,PLATZY.Fees,uint256,address[],uint16[],address)
(contracts/PLATZY.sol#54-82) performs a multiplication on the result of a division:
        - maxWalletSize = (_maxSupply / 100) * 3 (contracts/PLATZY.sol#70)
PLATZY.constructor(string,string,PLATZY.Fees,uint256,address[],uint16[],address)
(contracts/PLATZY.sol#54-82) performs a multiplication on the result of a division:
        - maxTxAmount = (_maxSupply / 100) * 2 (contracts/PLATZY.sol#71)
PLATZY.constructor(string,string,PLATZY.Fees,uint256,address[],uint16[],address)
(contracts/PLATZY.sol#54-82) performs a multiplication on the result of a division:
        - maxTaxSwap = (maxSupply / 1000) * 5 (contracts/PLATZY.sol#73)
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) performs a
multiplication on the result of a division:
        - taxAmount = (amount / 100) * buyTax (contracts/PLATZY.sol#111)
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) performs a
multiplication on the result of a division:
        - taxAmount = (amount / 100) * sellTax (contracts/PLATZY.sol#114)
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) performs a
multiplication on the result of a division:
        - lpAmount = (tokenBalance / 100) * lpPercent (contracts/PLATZY.sol#121)
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) performs a
multiplication on the result of a division:
        - burnAmount = (tokenBalance / 100) * burnPercent (contracts/PLATZY.sol#123)
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) performs a
multiplication on the result of a division:
        - lpTokens = lpAmount / 2 (contracts/PLATZY.sol#122)
        - lpETH = (balance * lpTokens) / tokensToSwap (contracts/PLATZY.sol#128)
```

```
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) performs a
multiplication on the result of a division:
        - toSend = min((balance / 100) * shares[i],address(this).balance) (contracts/
PLATZY.sol#133)
PLATZYAirdrop.claim(address,uint256,bytes32[]) (contracts/PLATZYAirdrop.sol#114-135)
performs a multiplication on the result of a division:
        - _initial = _amount * (_vested / 100) (contracts/PLATZYAirdrop.sol#130)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply
INFO:Detectors:
PLATZY.addLiquidity(uint256,uint256) (contracts/PLATZY.sol#144-154) ignores return
value by swapRouter.addLiquidityETH{value: _ethAmount}
(address(this),_tokenAmount,0,0,address(0xdead),block.timestamp) (contracts/
PLATZY.sol#146-153)
PLATZY.createPair() (contracts/PLATZY.sol#156-162) ignores return value by
IERC20(swapPair).approve(address(swapRouter),type()(uint256).max) (contracts/
PLATZY.sol#161)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
PLATZY.constructor(string,string,PLATZY.Fees,uint256,address[],uint16[],address)._supply
Wallet (contracts/PLATZY.sol#61) lacks a zero-check on :
                - supplyWallet = _supplyWallet (contracts/PLATZY.sol#80)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation
INFO:Detectors:
PLATZYAirdrop.createVestingSchedule(address,uint256) (contracts/
PLATZYAirdrop.sol#142-154) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(getWithdrawableAmount() >= _amount,Insufficient tokens)
(contracts/PLATZYAirdrop.sol#145)
PLATZYAirdrop.release(address) (contracts/PLATZYAirdrop.sol#160-171) uses timestamp for
comparisons
        Dangerous comparisons:
        - require(bool)(vestingSchedules[_address].initialized) (contracts/
PLATZYAirdrop.sol#163)
        - require(bool,string)(vestedAmount > 0,None releasable) (contracts/
PLATZYAirdrop.sol#166)
PLATZYAirdrop._computeReleasableAmount(PLATZYAirdrop.VestingSchedule) (contracts/
PLATZYAirdrop.sol#191-210) uses timestamp for comparisons
        Dangerous comparisons:
        - currentTime >= vestingSchedule.start + _duration (contracts/
PLATZYAirdrop.sol#197)
```

```
PLATZYAirdrop.withdraw(uint256) (contracts/PLATZYAirdrop.sol#216-219) uses timestamp
for comparisons
        Dangerous comparisons:
        - require(bool,string)(getWithdrawableAmount() >= amount,Insufficient funds)
(contracts/PLATZYAirdrop.sol#217)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp
INFO:Detectors:
PLATZY._transfer(address,address,uint256) (contracts/PLATZY.sol#93-142) has a high
cyclomatic complexity (14).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-
complexity
INFO:Detectors:
Pragma version^0.8.20 (contracts/PLATZY.sol#20) necessitates a version too recent to be
trusted. Consider deploying with 0.8.18.
Pragma version^0.8.19 (contracts/PLATZYAirdrop.sol#20) necessitates a version too
recent to be trusted. Consider deploying with 0.8.18.
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Low level call in PLATZY._transfer(address,address,uint256) (contracts/
PLATZY.sol#93-142):
        - (success,None) = address(payees[i]).call{value: toSend}() (contracts/
PLATZY.sol#134)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-
calls
INFO:Detectors:
Parameter PLATZY.addLiquidity(uint256,uint256)._tokenAmount (contracts/PLATZY.sol#144)
is not in mixedCase
Parameter PLATZY.addLiquidity(uint256,uint256)._ethAmount (contracts/PLATZY.sol#144) is
not in mixedCase
Parameter PLATZY.swapTokensForEth(uint256)._tokenAmount (contracts/PLATZY.sol#168) is
not in mixedCase
Parameter PLATZYAirdrop.claim(address,uint256,bytes32[])._address (contracts/
PLATZYAirdrop.sol#115) is not in mixedCase
Parameter PLATZYAirdrop.claim(address,uint256,bytes32[])._amount (contracts/
PLATZYAirdrop.sol#116) is not in mixedCase
Parameter PLATZYAirdrop.claim(address,uint256,bytes32[])._proof (contracts/
PLATZYAirdrop.sol#117) is not in mixedCase
Parameter PLATZYAirdrop.createVestingSchedule(address,uint256)._beneficiary (contracts/
```

```
PLATZYAirdrop.sol#142) is not in mixedCase
Parameter PLATZYAirdrop.createVestingSchedule(address,uint256)._amount (contracts/
PLATZYAirdrop.sol#142) is not in mixedCase
Parameter PLATZYAirdrop.release(address)._address (contracts/PLATZYAirdrop.sol#160) is
not in mixedCase
Parameter PLATZYAirdrop.computeReleasableAmount(address)._beneficiary (contracts/
PLATZYAirdrop.sol#178) is not in mixedCase
Variable PLATZYAirdrop._token (contracts/PLATZYAirdrop.sol#58) is not in mixedCase
Variable PLATZYAirdrop._duration (contracts/PLATZYAirdrop.sol#60) is not in mixedCase
Variable PLATZYAirdrop._vested (contracts/PLATZYAirdrop.sol#62) is not in mixedCase
Variable PLATZYAirdrop._merkleRoot (contracts/PLATZYAirdrop.sol#64) is not in mixedCase
Variable PLATZYAirdrop._claimIsActive (contracts/PLATZYAirdrop.sol#66) is not in
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions
INFO:Detectors:
Variable PLATZYAirdrop._merkleRoot (contracts/PLATZYAirdrop.sol#64) is too similar to
PLATZYAirdrop.setMerkleRoot(bytes32).merkleRoot_ (contracts/PLATZYAirdrop.sol#103)
Variable PLATZYAirdrop._claimIsActive (contracts/PLATZYAirdrop.sol#66) is too similar
to PLATZYAirdrop.setClaimStatus(bool).claimIsActive_ (contracts/PLATZYAirdrop.sol#98)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
too-similar
INFO:Detectors:
Loop condition `i < payees.length` (contracts/PLATZY.sol#77) should use cached array
length instead of referencing `length` member of the storage array.
 Loop condition `i < payees.length` (contracts/PLATZY.sol#132) should use cached array
length instead of referencing `length` member of the storage array.
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cache-array-
length
INFO:Detectors:
PLATZY.burnPercent (contracts/PLATZY.sol#33) should be immutable
PLATZY.buyTax (contracts/PLATZY.sol#31) should be immutable
PLATZY.lpPercent (contracts/PLATZY.sol#34) should be immutable
PLATZY.maxSupply (contracts/PLATZY.sol#35) should be immutable
PLATZY.maxTaxSwap (contracts/PLATZY.sol#37) should be immutable
PLATZY.sellTax (contracts/PLATZY.sol#32) should be immutable
PLATZY.supplyWallet (contracts/PLATZY.sol#43) should be immutable
PLATZY.taxSwapThreshold (contracts/PLATZY.sol#36) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-
variables-that-could-be-declared-immutable
INFO:Slither:. analyzed (16 contracts with 88 detectors), 53 result(s) found
```

0x Guard