



Smart Contract Audit

FOR

President Optimus

DATED : 20 June 23'



AUDIT SUMMARY

Project name - Prsident Optimus

Date: 20 June, 2023

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

Audit Status: **Passed**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	0	0	3
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0



USED TOOLS

Tools:

1- Manual Review:

a line by line code review has been performed by audit ace team.

2- BSC Test Network:

all tests were done on BSC Test network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

<https://testnet.bscscan.com/token/0xd46bbb5776176d4b047b7230c0b9c062ccec190>



Token Information

Token Name : President Optimus

Token Symbol: P-OPTIMUS

Decimals: 18

Token Supply:1,000,000,000

Token Address:

0x3EBE82fCFAfd5d9E5f74297483195a1Fa9E45a62

Checksum:

ff298f4f3aa0cc991d3842bf26fad8ea36e22f7a

Owner:

0xc51Cd60D0822e42d3604E9254aD4dC75bf1ED555



TOKEN OVERVIEW

Fees:

Buy Fees: 3%

Sell Fees: 3%

Transfer Fees: 3%

Fees Privilege: Static Fees

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: none

Blacklist: No

Other Privileges:

- Initial distribution of the tokens



AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
 - Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
 - Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
 - Test coverage analysis determines whether the test cases are covering the code and how much code is exercised when we run the test cases.
 - Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
 - Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.
-

VULNERABILITY CHECKLIST

- | | |
|------------------------------------|-------------------------------|
| ✓ Return values of low-level calls | ✓ Gasless Send |
| ✓ Private modifier | ✓ Using block.timestamp |
| ✓ Multiple Sends | ✓ Re-entrancy |
| ✓ Using Suicide | ✓ Tautology or contradiction |
| ✓ Gas Limitand Loops | ✓ Timestamp Dependence |
| ✓ Address hardcoded | ✓ Revert/require functions |
| ✓ Exception Disorder | ✓ Use of tx.origin |
| ✓ Using inline assembly | ✓ Integer overflow/underflow |
| ✓ Divide before multiply | ✓ Dangerous strict equalities |
| ✓ Missing Zero Address Validation | ✓ Using SHA3 |
| ✓ Compiler version not fixed | ✓ Using throw |
-



CLASSIFICATION OF RISK

Severity

Description

◆ Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
◆ High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
◆ Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
◆ Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
◆ Gas Optimization / Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

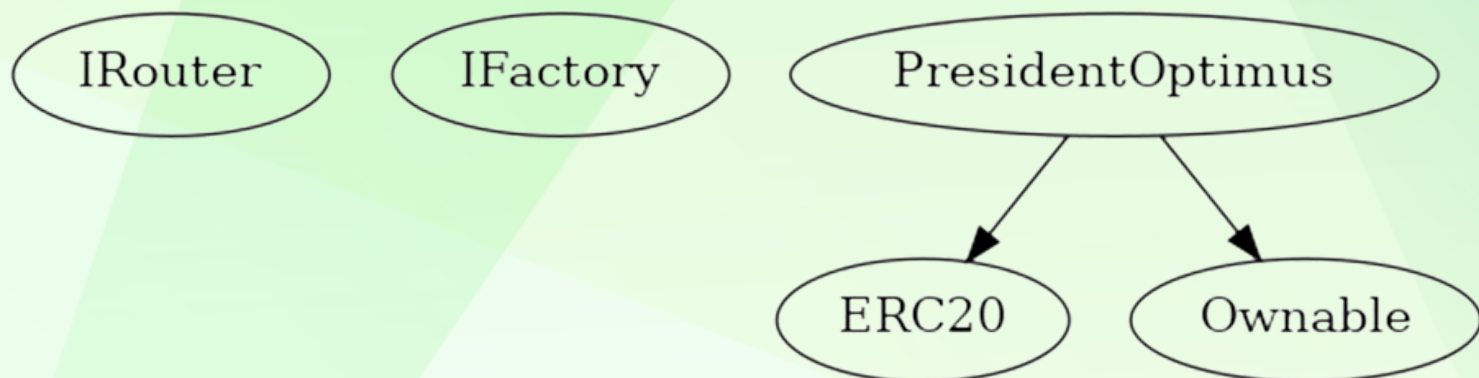
Findings

Severity

Found

◆ Critical	0
◆ High-Risk	1
◆ Medium-Risk	0
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	3

INHERITANCE TREE





POINTS TO NOTE

- owner is able to change buy/sell/transfer tax (3%)
 - there is 5% tax for 1 minute after launch which will be reduced to 3% (forever)
 - owner is not able to set max buy/sell/transfer/wallet limits
 - owner is not able to blacklist an arbitrary wallet
 - owner is not able to mint new tokens
 - owner is not able to disable trades
-

CONTRACT ASSESMENT

Contract	Type	Bases			
:-----: :-----: :-----: :-----: :-----:					
↳	**Function Name**	**Visibility**	**Mutability**	**Modifiers**	
	IRouter	Interface			
↳	WETH	External	!	NO!	
↳	factory	External	!	NO!	
↳	swapExactTokensForETHSupportingFeeOnTransferTokens	External	!	 NO!	
↳	swapExactTokensForETH	External	!	 NO!	
	IFactory	Interface			
↳	getPair	External	!	NO!	
	PresidentOptimus	Implementation	ERC20, Ownable		
↳	<Constructor>	Public	!	 ERC20	
↳	<Receive Ether>	External	!	 NO!	
↳	decimals	Public	!	NO!	
↳	updateClaimWait	External	!	 onlyOwner	
↳	getClaimWait	External	!	NO!	
↳	getTotalDividendsDistributed	External	!	NO!	
↳	withdrawableDividendOf	Public	!	NO!	
↳	dividendTokenBalanceOf	Public	!	NO!	
↳	excludeFromDividends	External	!	 onlyOwner	
↳	getAccountDividendsInfo	External	!	NO!	
↳	getAccountDividendsInfoAtIndex	External	!	NO!	
↳	processDividendTracker	External	!	 NO!	
↳	claim	External	!	 NO!	
↳	getLastProcessedIndex	External	!	NO!	
↳	getNumberOfDividendTokenHolders	External	!	NO!	
↳	excludeFromFee	Public	!	 onlyOwner	
↳	isSwapPair	Private	 		
↳	_transfer	Internal	 		
↳	_swapAndTransferFee	Private	 		
↳	_swapForETH	Private	 		

Legend

Symbol	Meaning
:-----: -----	
	Function can modify state
	Function is payable



STATIC ANALYSIS

```
Low level call in Address.sendValue(address,uint256) (contracts/PresidentOptimus/PresidentOptimus.sol#1493-1498):
- (success) = recipient.call{value: amount}() (contracts/PresidentOptimus/PresidentOptimus.sol#1496)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/PresidentOptimus/PresidentOptimus.sol#1556-1563):
- (success,returndata) = target.call{value: value}(data) (contracts/PresidentOptimus/PresidentOptimus.sol#1561)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/PresidentOptimus/PresidentOptimus.sol#1581-1588):
- (success,returndata) = target.staticcall(data) (contracts/PresidentOptimus/PresidentOptimus.sol#1586)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/PresidentOptimus/PresidentOptimus.sol#1606-1612):
- (success,returndata) = target.delegatecall(data) (contracts/PresidentOptimus/PresidentOptimus.sol#1610)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter DividendPayingToken.dividendOf(address). owner (contracts/PresidentOptimus/PresidentOptimus.sol#1070) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address). owner (contracts/PresidentOptimus/PresidentOptimus.sol#1077) is not in mixedCase
Parameter DividendPayingToken.withdrawDividendOf(address). owner (contracts/PresidentOptimus/PresidentOptimus.sol#1084) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address). owner (contracts/PresidentOptimus/PresidentOptimus.sol#1093) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/PresidentOptimus/PresidentOptimus.sol#1006) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter DividendTracker.getAccount(address). account (contracts/PresidentOptimus/PresidentOptimus.sol#1292) is not in mixedCase
Function IRouter.WETH() (contracts/PresidentOptimus/PresidentOptimus.sol#1682) is not in mixedCase
Constant PresidentOptimus.zeroAddr (contracts/PresidentOptimus/PresidentOptimus.sol#1708) is not in UPPER_CASE_WITH_UNDERSCORES
Constant PresidentOptimus.decimals (contracts/PresidentOptimus/PresidentOptimus.sol#1719) is not in UPPER_CASE_WITH_UNDERSCORES
Constant PresidentOptimus.reward (contracts/PresidentOptimus/PresidentOptimus.sol#1726) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable DividendPayingToken.constructor(address). rewardToken (contracts/PresidentOptimus/PresidentOptimus.sol#1026) is too similar to DividendTracker.constructor(address,uint256).rewardToken
(contracts/PresidentOptimus/PresidentOptimus.sol#1246)
Variable DividendPayingToken._withdrawDividendOfUser(address). withdrawableDividend (contracts/PresidentOptimus/PresidentOptimus.sol#1050) is too similar to DividendTracker.getAccount(address).
withdrawableDividends (contracts/PresidentOptimus/PresidentOptimus.sol#1299)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

PresidentOptimus.slitherConstructorVariables() (contracts/PresidentOptimus/PresidentOptimus.sol#1704-1941) uses literals with too many digits:
- transferFeeAt = supply * 5 / 1000000 (contracts/PresidentOptimus/PresidentOptimus.sol#1727)
PresidentOptimus.slitherConstructorVariables() (contracts/PresidentOptimus/PresidentOptimus.sol#1704-1941) uses literals with too many digits:
- gasForProcessing = 300000 (contracts/PresidentOptimus/PresidentOptimus.sol#1742)
PresidentOptimus.slitherConstructorVariables() (contracts/PresidentOptimus/PresidentOptimus.sol#1704-1941) uses literals with too many digits:
- miniumForDividend = supply / 10000000 (contracts/PresidentOptimus/PresidentOptimus.sol#1743)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/PresidentOptimus/PresidentOptimus.sol#238) is never used in SafeMathInt (contracts/PresidentOptimus/PresidentOptimus.sol#236-293)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DividendPayingToken.weth (contracts/PresidentOptimus/PresidentOptimus.sol#1001) should be constant
PresidentOptimus.gasForProcessing (contracts/PresidentOptimus/PresidentOptimus.sol#1742) should be constant
PresidentOptimus.marketing (contracts/PresidentOptimus/PresidentOptimus.sol#1717) should be constant
PresidentOptimus.supply (contracts/PresidentOptimus/PresidentOptimus.sol#1720) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DividendPayingToken.rewardToken (contracts/PresidentOptimus/PresidentOptimus.sol#1000) should be immutable
DividendTracker.minimumTokenBalanceForDividends (contracts/PresidentOptimus/PresidentOptimus.sol#1239) should be immutable
PresidentOptimus.dividendTracker (contracts/PresidentOptimus/PresidentOptimus.sol#1731) should be immutable
PresidentOptimus.miniumForDividend (contracts/PresidentOptimus/PresidentOptimus.sol#1743) should be immutable
PresidentOptimus.transferFeeAt (contracts/PresidentOptimus/PresidentOptimus.sol#1727) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

**Result => A static analysis of contract's source code has been performed using slither,
No major issues were found in the output**



FUNCTIONAL TESTING

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):

<https://testnet.bscscan.com/tx/0x36b93116c69833a83fbb1a4236afb585c6a21be3201d73731a7d20abbc95a477>

2- Buying when excluded (0% tax) (passed):

<https://testnet.bscscan.com/tx/0x0e303bc394067c9e098ed0626c06c7dc05761fe4c44b4d6ca873e3238f9a1443>

3- Selling when excluded (0% tax) (passed):

<https://testnet.bscscan.com/tx/0xb96ace7e7d9fd314a8511986154992632f7a427e9421be5a1421838a50b6bbff>

4- Transferring when excluded (0% tax) (passed):

<https://testnet.bscscan.com/tx/0x3b3c253b59df1f6143ff7dea2776ea08dbd03e759e5776acbd55354f7f7590c3>

5- Buying when not excluded from fees (3% tax) (passed):

<https://testnet.bscscan.com/tx/0x3b338ff72c4ea3b1ebe5aa1a668168df052d80c5475368e634b4f85c73024ea0>

6- Selling when not excluded from fees (3% tax) (passed):

<https://testnet.bscscan.com/tx/0xd0e78682358dce6b30a93371fec275b464b7b9b0af4db898a83d9309286beb2>



FUNCTIONAL TESTING

7- Transferring when not excluded from fees (3% tax) (passed):

<https://testnet.bscscan.com/tx/0x39e1bc42149f9127c8eea1ed4d4416c973d1e298b3a970afce7f871b3f1c3905>

8-Internal swap (passed):

- BNB fee sent to marketing wallet
- Rewards distributed

<https://testnet.bscscan.com/tx/0x39e1bc42149f9127c8eea1ed4d4416c973d1e298b3a970afce7f871b3f1c3905>

ISSUES FOUND

Centralization – recursion at process function and internal swap

Severity: **High**

Status: **Open**

Overview:

Since dividend tracker is distributing native tokens (P-OPTIMUS), `_transfer` function will be called by dividend tracker on each reward distribution call hence calling `process` and `_swapAndTransferFee` functions again

```
if (!swapping && from != address(dividendTracker)) {  
    uint256 gas = gasForProcessing;  
  
    try dividendTracker.process(gas) returns (uint256 iterations, uint256 claims, uint256  
lastProcessedIndex) {  
        emit ProcessedDividendTracker(iterations, claims, lastProcessedIndex, true, gas, tx.origin);  
    } catch {}  
}  
  
-----  
if (canSwap && !isSwapPair(from) && !swapping && !isExcludedFromFee[from]  
&& !isExcludedFromFee[to]) {  
    swapping = true;  
    _swapAndTransferFee(feeInContract);  
    swapping = false;  
}
```

Suggestion

its suggested to ensure that dividend tracker is not calling “proecess” and “_swapAndTransferFee” functions

```
if (msg.sender != address(dividendTracker) && from != address(dividendTracker)) {  
    // rest of the code  
}  
  
if (canSwap && !isSwapPair(from) && !swapping && !isExcludedFromFee[from]  
&& !isExcludedFromFee[to] && from != address(dividendTracker) && to !=  
address(dividendTracker)) {  
    swapping = true;  
    _swapAndTransferFee(feeInContract);  
    swapping = false;  
}
```

ISSUES FOUND

Missing Logic – Stuck ETH and Tokens

Severity: **Informational**

Status: **Open**

Overview:

Contract has no function to withdraw stuck ETH or ERC20 tokens. If ETH or ERC20 sent to contract by mistake, there won't be any ways to withdraw those funds.

Suggestion

Implement a function to be able to withdraw stuck ETH and ERC20 tokens from the contract (by owner)

Missing Logic – Immutable tax

Severity: **Informational**

Status: **Open**

Overview:

Fees are immutable and owner is not able to change fees later.

Suggestion

It's suggested to implement a function for updating fees depending on different market conditions in a safe range.

$0 \leq \text{total buy fees} \leq 10$

$0 \leq \text{total sell fees} \leq 10$

$0 \leq \text{total transfer fees} \leq 10$

<https://docs.pinksale.finance/important/safu-contract>

Missing Logic – Immutable internal swap threshold

Severity: **Informational**

Status: **Open**

Overview:

Internal swap threshold (**transferFeeAt**) is immutable meaning that owner is not able to adjust this value depending on liquidity pool size or different market conditions.

Suggestion

It's suggested to implement a function for updating **transferFeeAt** depending on liquidity pool size or different market conditions.



DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.



ABOUT AUDITACE

We specialize in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.



<https://auditace.tech/>



https://t.me/Audit_Ace



https://twitter.com/auditace_



<https://github.com/Audit-Ace>
