auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Owl Finance

DATED : 17 June 23'

AUDIT SUMMARY

Project name – Owl Finance

Date: 17 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	1
Acknowledged	0	0	0	0	0
Resolved	0	1	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/address/0x38aC93fEA4B96F6a1F147884E6D13728143F3CAD

Token Information

Token Name : Owl Finance

Token Symbol: OwlFi

Decimals: 18

Token Supply:1,000,000,000,000

Token Address:
0x3dcf110Fd9D62FB908764a949e08979d20fFdCE8

Checksum:
a323566741554a568d95eafc42fa69140cdde6da

Owner:
0xA28a0665a8dcd2ad58aF4ceE6BA0A95644B4d3dB

TOKEN OVERVIEW

Fees:

Buy Fees: 0-10%

Sell Fees: 0-10%

Transfer Fees: 0%

Fees Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: none

Blacklist: No

Other Priviliges:

- Initial distribution of the tokens
- updating fees
- updating max wallet

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 isexercised 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.

auditace.tech

VULNERABILITY CHECKLIST

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

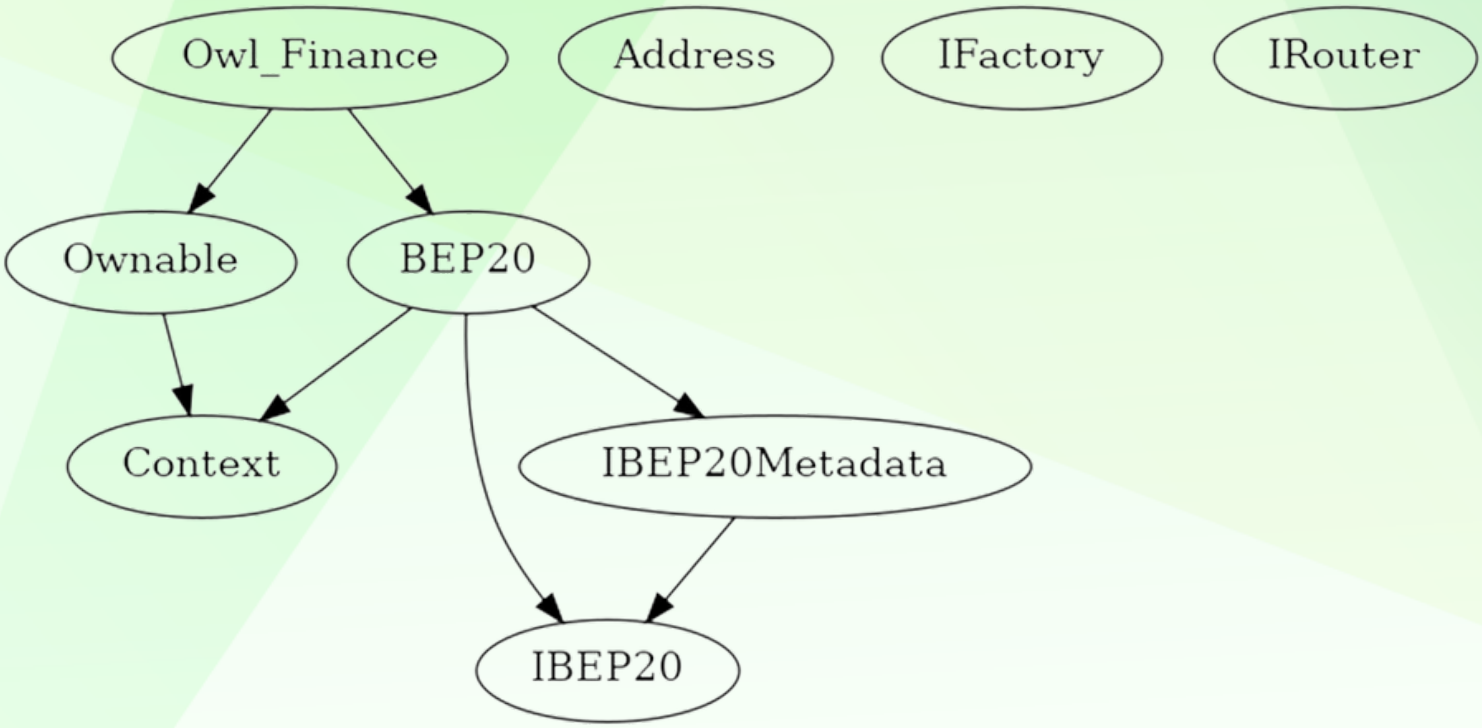
CLASSIFICATION OF RISK

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(Resolved)
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	1

INHERITANCE TREE



POINTS TO NOTE

- owner is able to change buy/sell tax in range 0-10%
- transfer fees are always zero
- there is a max wallet which can be between 0.1% - 100% of total supply
- owner is not able to blacklist an arbitrary wallet
- owner is not able to mint new tokens
- owner is not able to disable trades
- owner must enable trades manually for investors

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IBEP20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IBEP20Metadata	Interface	IBEP20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
BEP20	Implementation	Context, IBEP20, IBEP20Metadata		
└	<Constructor>	Public ❗	🛑	NO❗
└	name	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	decimals	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	transfer	Public ❗	🛑	NO❗
└	allowance	Public ❗		NO❗
└	approve	Public ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	_transfer	Internal 🔒	🛑	
└	_tokengeneration	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
Address	Library			
└	sendValue	Internal 🔒	🛑	
Ownable	Implementation	Context		

CONTRACT ASSESMENT

└	<Constructor>	Public 🔴	🟢	NO🔴
└	owner	Public 🔴		NO🔴
└	renounceOwnership	Public 🔴	🟢	onlyOwner
└	transferOwnership	Public 🔴	🟢	onlyOwner
└	_setOwner	Private 🔐	🟢	
IFactory	Interface			
└	createPair	External 🔴	🟢	NO🔴
IRouter	Interface			
└	factory	External 🔴		NO🔴
└	WETH	External 🔴		NO🔴
└	addLiquidityETH	External 🔴	💵	NO🔴
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External 🔴	🟢	NO🔴
Owl_Finance	Implementation	BEP20, Ownable		
└	<Constructor>	Public 🔴	🟢	BEP20
└	approve	Public 🔴	🟢	NO🔴
└	transferFrom	Public 🔴	🟢	NO🔴
└	increaseAllowance	Public 🔴	🟢	NO🔴
└	decreaseAllowance	Public 🔴	🟢	NO🔴
└	transfer	Public 🔴	🟢	NO🔴
└	_transfer	Internal 🔒	🟢	
└	Liquify	Private 🔐	🟢	lockTheSwap
└	swapTokensForETH	Private 🔐	🟢	
└	addLiquidity	Private 🔐	🟢	
└	updateLiquidityProvide	External 🔴	🟢	onlyOwner
└	updateLiquidityTreshhold	External 🔴	🟢	onlyOwner
└	EnableTrading	External 🔴	🟢	onlyOwner
└	updatedeadline	External 🔴	🟢	onlyOwner
└	updateMarketingWallet	External 🔴	🟢	onlyOwner
└	SetMaxTxLimit	External 🔴	🟢	onlyOwner
└	updateBuyTaxes	Public 🔴	🟢	onlyOwner

CONTRACT ASSESMENT

└	updateSellTaxes	Public ❗	🛑	onlyOwner
└	updateExemptFee	External ❗	🛑	onlyOwner
└	bulkExemptFee	External ❗	🛑	onlyOwner
└	rescueBNB	External ❗	🛑	onlyOwner
└	rescueBSC20	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Context._msgData() (contracts/Token.sol#14-17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/Token.sol#7) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#350-361):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#356)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Variable BEP20._balances (contracts/Token.sol#69) is not in mixedCase
Variable BEP20._allowances (contracts/Token.sol#71) is not in mixedCase
Function IRouter.WETH() (contracts/Token.sol#413) is not in mixedCase
Contract Owl_Finance (contracts/Token.sol#440-776) is not in CapWords
Function Owl_Finance.Liquify(uint256,Owl_Finance.Taxes) (contracts/Token.sol#632-665) is not in mixedCase
Parameter Owl_Finance.updateLiquidityTreshhold(uint256).new_amount (contracts/Token.sol#704) is not in mixedCase
Function Owl_Finance.EnableTrading() (contracts/Token.sol#709-714) is not in mixedCase
Parameter Owl_Finance.updatedeadline(uint256)._deadline (contracts/Token.sol#716) is not in mixedCase
Function Owl_Finance.SetMaxTxLimit(uint256) (contracts/Token.sol#728-731) is not in mixedCase
Parameter Owl_Finance.updateBuyTaxes(uint256,uint256,uint256)._development (contracts/Token.sol#734) is not in mixedCase
Parameter Owl_Finance.updateBuyTaxes(uint256,uint256,uint256)._marketing (contracts/Token.sol#735) is not in mixedCase
Parameter Owl_Finance.updateBuyTaxes(uint256,uint256,uint256)._liquidity (contracts/Token.sol#736) is not in mixedCase
Parameter Owl_Finance.updateSellTaxes(uint256,uint256,uint256)._development (contracts/Token.sol#744) is not in mixedCase
Parameter Owl_Finance.updateSellTaxes(uint256,uint256,uint256)._marketing (contracts/Token.sol#745) is not in mixedCase
Parameter Owl_Finance.updateSellTaxes(uint256,uint256,uint256)._liquidity (contracts/Token.sol#746) is not in mixedCase
Parameter Owl_Finance.updateExemptFee(address,bool)._address (contracts/Token.sol#753) is not in mixedCase
Variable Owl_Finance.genesis_block (contracts/Token.sol#453) is not in mixedCase
Constant Owl_Finance.deadWallet (contracts/Token.sol#458) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#15)" inContext (contracts/Token.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Owl_Finance.constructor() (contracts/Token.sol#480-495) uses literals with too many digits:
        - _tokengeneration(msg.sender,1000000000000 * 10 ** decimals()) (contracts/Token.sol#481)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

Owl_Finance.launchtax (contracts/Token.sol#455) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

Owl_Finance.pair (contracts/Token.sol#444) should be immutable
Owl_Finance.router (contracts/Token.sol#443) 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/0xb0fa71dd1a814a1150f7e88a585b549286760482bc78a838ff9ed0de50059af7

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xe4d4803fd1f41af0ecb129385f9b812840b9bd02ed5f6d93433d79a3b755965c

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x65271d02120cbab1e63c8aa1833684d0def69fcb7e335e28988aaee8b8c7b929

4- Transferring when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x38480cca306b446ee189744a1d080091f46fff6693f25c782379b0b23c4591e2

5- Buying when not excluded from fees (0-10% tax) (passed):
https://testnet.bscscan.com/tx/0x58623ed0ff3ddf2ecc06266790a157f796c9b564b3537d4821f82beaa58a2bcf

6- Selling when not excluded from fees (0-10% tax) (passed):
https://testnet.bscscan.com/tx/0xa3a222ebd6b37eb79fe355900f09ddf59fb5bb01380ec7293bd75ffdb9a53855

FUNCTIONAL TESTING

7- Transferring from a regular wallet (0% tax) (passed):
https://testnet.bscscan.com/tx/0x6b54c82176aa44ad3de7a171255d49af04d2f26e19913e3464f70fc5c11b31ad

8-Internal swap (passed):
- BNB fee sent to marketing wallet
- Auto liquidity (LP tokens sent to dead wallet)

https://testnet.bscscan.com/tx/0xae32409f9699ef9e45ff98a921aae27bee3417c24898670860969cdb5374433c

ISSUES FOUND

Centralization – Trades must be enabled

Severity: High

function: EnableTrading

Status: Resovled (owned by safu developer)

Overview:

The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    providingLiquidity = true;
    genesis_block = block.number;
}
```

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.
3. Transfer ownership to a trusted and valid 3rd party in order to guarantee enabling of the trades **(Applied)**

since contract is owned by a certified pinksale safu developer, this issue is mitigated.

ISSUES FOUND

Centralization – maximum wallet

Severity: Informational

function: SetMaxTxLimit

Status: Resovled (Max wallet is in accordance Pinksale safu criteria)

Overview:

The smart contract owner can put a maximum wallet limit on trades. This limit can be a number in range 0.1% - 100% of total supply.
Traders wont be able to buy or receive tokens (throught transfers) if their current balance plus "transferring" or "buying" amount is greater than allowed maximum wallet size. It worth noting that "selling" tokens wont be affected by this limit

```
function SetMaxTxLimit(uint256 maxWallet) external onlyOwner {
  require(maxWallet >= 1e9, "Cannot set max wallet amount lower than 0.1%");
  maxWalletLimit = maxWallet * 10**decimals();
}
```

Suggestion

Current maximum wallet limit is in accordance with pinksale safu criteria. This means maximum wallet size can not be upgraded to a very low value (e.g. 0) which may negatively affect investors.

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

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 specializes 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