AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Figment

DATED : 08 Mar 23'

AUDIT SUMMARY

Project name – Figment

Date: 08 March, 2023

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

Audit Status: Passed with Medium Risk

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	1	0
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/0x32d19d97a85f45644e2bccb654e5aa277868feda

Token Information

Token Name : Figment

Token Symbol: Figma

Decimals: 18

Token Supply: 1,000,000,000

Token Address:
0xcEAa73aAAAE02c8A1175F7273741648B5086F5b7

Checksum:
40f8b00e12adf454bf3d074d0e915c10a291721b

Owner:
0xb971ff557D6d40ef75248257DC27382c73Dd4057

TOKEN OVERVIEW

Fees:

Buy Fees: 3%

Sell Fees: 3%

Transfer Fees: 0%

Fees Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: including and excluding from fees

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.

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**	0
◆ **Medium-Risk**	0
◆ **Low-Risk**	1
◆ **Gas Optimization / Suggestions**	0

INHERITANCE TREE
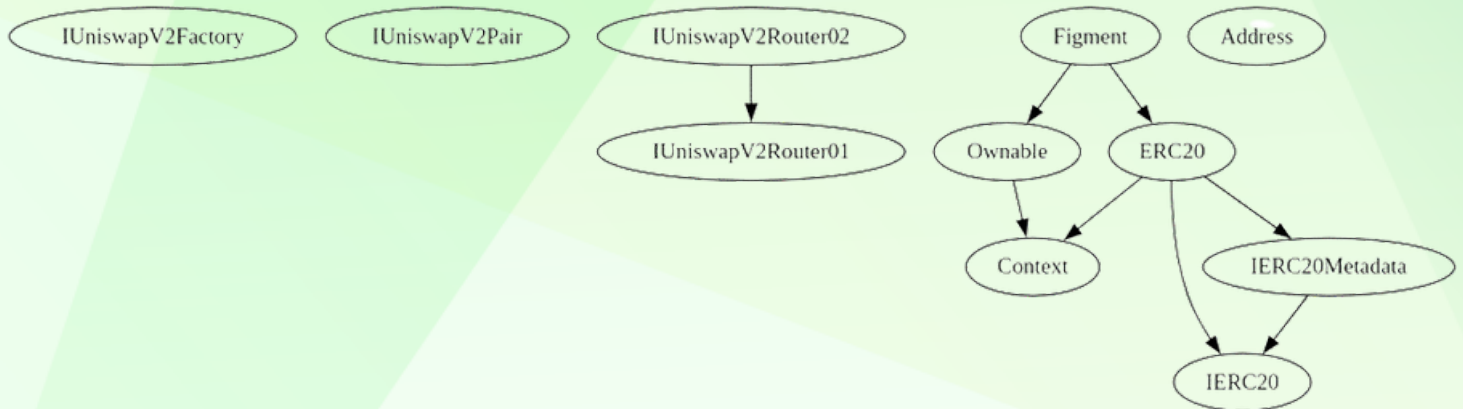
POINTS TO NOTE

- Owner is not able to change fees more than 3% for buy/sell, transfer taxes are 0
- Owner is able to set max buy/sell/transfer/holding amount, this amount has a minimum limit of 1% of total supply (safeguard)
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner must enable trading in order for investors to be able to buy/sell/transfer (enableTrading feature)
- Owner is not able to mint new tokens

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IUniswapV2Factory	Interface			
└	feeTo	External ❗		NO❗
└	feeToSetter	External ❗		NO❗
└	getPair	External ❗		NO❗
└	allPairs	External ❗		NO❗
└	allPairsLength	External ❗		NO❗
└	createPair	External ❗	🛑	NO❗
└	setFeeTo	External ❗	🛑	NO❗
└	setFeeToSetter	External ❗	🛑	NO❗
IUniswapV2Pair	Interface			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🛑	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗
└	swap	External ❗	🛑	NO❗
└	skim	External ❗	🛑	NO❗
└	sync	External ❗	🛑	NO❗
└	initialize	External ❗	🛑	NO❗

CONTRACT ASSESMENT

IUniswapV2Router01	Interface				
└	factory	External ❗			NO❗
└	WETH	External ❗			NO❗
└	addLiquidity	External ❗	🛑		NO❗
└	addLiquidityETH	External ❗	💵		NO❗
└	removeLiquidity	External ❗	🛑		NO❗
└	removeLiquidityETH	External ❗	🛑		NO❗
└	removeLiquidityWithPermit	External ❗	🛑		NO❗
└	removeLiquidityETHWithPermit	External ❗	🛑		NO❗
└	swapExactTokensForTokens	External ❗	🛑		NO❗
└	swapTokensForExactTokens	External ❗	🛑		NO❗
└	swapExactETHForTokens	External ❗	💵		NO❗
└	swapTokensForExactETH	External ❗	🛑		NO❗
└	swapExactTokensForETH	External ❗	🛑		NO❗
└	swapETHForExactTokens	External ❗	💵		NO❗
└	quote	External ❗			NO❗
└	getAmountOut	External ❗			NO❗
└	getAmountIn	External ❗			NO❗
└	getAmountsOut	External ❗			NO❗
└	getAmountsIn	External ❗			NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01			
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑		NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑		NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑		NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵		NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑		NO❗
IERC20	Interface				
└	totalSupply	External ❗			NO❗
└	balanceOf	External ❗			NO❗
└	transfer	External ❗	🛑		NO❗
└	allowance	External ❗			NO❗
└	approve	External ❗	🛑		NO❗
└	transferFrom	External ❗	🛑		NO❗
IERC20Metadata	Interface	IERC20			
└	name	External ❗			NO❗
└	symbol	External ❗			NO❗
└	decimals	External ❗			NO❗

CONTRACT ASSESMENT

Address	Library			
└ isContract	Internal 🔒			
└ sendValue	Internal 🔒	🛑		
└ functionCall	Internal 🔒	🛑		
└ functionCall	Internal 🔒	🛑		
└ functionCallWithValue	Internal 🔒	🛑		
└ functionCallWithValue	Internal 🔒	🛑		
└ functionStaticCall	Internal 🔒			
└ functionStaticCall	Internal 🔒			
└ functionDelegateCall	Internal 🔒	🛑		
└ functionDelegateCall	Internal 🔒	🛑		
└ verifyCallResultFromTarget	Internal 🔒			
└ verifyCallResult	Internal 🔒			
└ _revert	Private 🔐			
Context	Implementation			
└ _msgSender	Internal 🔒			
└ _msgData	Internal 🔒			
Ownable	Implementation	Context		
└ <Constructor>	Public ❗	🛑	NO❗	
└ owner	Public ❗		NO❗	
└ renounceOwnership	Public ❗	🛑	onlyOwner	
└ transferOwnership	Public ❗	🛑	onlyOwner	
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└ <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 🔒	🛑		
└ _mint	Internal 🔒	🛑		
└ _burn	Internal 🔒	🛑		

CONTRACT ASSESMENT

└	_approve	Internal 🔒	🛑		
└	_beforeTokenTransfer	Internal 🔒	🛑		
└	_afterTokenTransfer	Internal 🔒	🛑		
Figment	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO❗
└	isExcludedFromLimits	External ❗		NO❗
└	setExcludedFromLimits	External ❗	🛑	onlyOwner
└	lockFees	External ❗	🛑	onlyOwner
└	setFees	External ❗	🛑	onlyOwner
└	changeFIGMAburn_FBRprizepoolFunding_Wallet	External ❗	🛑	onlyOwner
└	enableTrading	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	setSwapEnabled	External ❗	🛑	onlyOwner
└	setMaxTxPercent	External ❗	🛑	onlyOwner
└	setMaxWalletSize	External ❗	🛑	onlyOwner
└	getMaxTX	External ❗		NO❗
└	getMaxWallet	External ❗		NO❗
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	swapAndSendMarketing	Private 🔐	🛑	

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Address.functionDelegateCall(address,bytes,string) (contracts/Token.sol#488-501) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/Token.sol#449-459) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/Token.sol#461-474) is never used and should be removed
Address.isContract(address) (contracts/Token.sol#375-377) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (contracts/Token.sol#521-531) is never used and should be removed
Address.verifyCallResultFromTarget(address,bool,bytes,string) (contracts/Token.sol#503-519) is never used and should be removed
Context._msgData() (contracts/Token.sol#556-559) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#746-761) 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-0.8.18 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#379-390):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#385)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/Token.sol#427-447):
        - (success,returndata) = target.call{value: value}(data) (contracts/Token.sol#437-439)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/Token.sol#461-474):
        - (success,returndata) = target.staticcall(data) (contracts/Token.sol#466)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/Token.sol#488-501):
        - (success,returndata) = target.delegatecall(data) (contracts/Token.sol#493)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#69) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#71) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#102) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/Token.sol#142) is not in mixedCase
Parameter Figment.setFees(uint16,uint16)._FIGMAtokensburnFundingFeeOnBuy (contracts/Token.sol#929) is not in mixedCase
Parameter Figment.setFees(uint16,uint16)._FBRprizepoolFundingFeeOnSell (contracts/Token.sol#930) is not in mixedCase
Function Figment.changeFIGMAburn_FBRprizepoolFunding_Wallet(address) (contracts/Token.sol#948-963) is not in mixedCase
Parameter Figment.changeFIGMAburn_FBRprizepoolFunding_Wallet(address)._FIGMAburn_FBRprizepoolFunding_Wallet (contracts/Token.sol#949) is not in mixedCase
Parameter Figment.setSwapEnabled(bool)._enabled (contracts/Token.sol#1044) is not in mixedCase
Variable Figment.FIGMAtokensburnFundingFeeOnBuy (contracts/Token.sol#800) is not in mixedCase
Variable Figment.FBRprizepoolFundingFeeOnSell (contracts/Token.sol#801) is not in mixedCase
Constant Figment.maxFIGMAtokensburnFundingFeeOnBuy (contracts/Token.sol#803) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Figment.maxFBRprizepoolFundingFeeOnSell (contracts/Token.sol#804) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Figment.BuySellFeesAreLocked (contracts/Token.sol#806) is not in mixedCase
Variable Figment.FIGMAburn_FBRprizepoolFunding_Wallet (contracts/Token.sol#810) is not in mixedCase
Variable Figment._ownerchosenDividend (contracts/Token.sol#1083-1084) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#557)" inContext (contracts/Token.sol#551-560)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#147) is too similar to IUniswapV2Router01.addLiq
uidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#148)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Figment.creator (contracts/Token.sol#798) should be immutable
Figment.pancakeswapV2Pair (contracts/Token.sol#792) should be immutable
Figment.pancakeswapV2Router (contracts/Token.sol#791) 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/0x203b56d568934d11d33e760665a5a2704c1930e594f85965efdb782b2fd88fbb

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x0b879e5d62884ceb62917125e322ed7f4e88b33c1b46e72c9e26cd5e77158922

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x3f7ab213acdf2158df997c7c4a31e6ec45e9379871635158c02e0296ec2f38cd

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xf205d5f27b69f191cf3d4b66c665bf5093bdbbd3da42447e3e040feff9990e78

5- Buying when not excluded (1% tax) (passed):
https://testnet.bscscan.com/tx/0x21894fa214107fe2c4e7aaadba12fd9980dd86dd7f12da8f6d1239aba3824d9e

6- Selling when not excluded (3% tax) (passed):
https://testnet.bscscan.com/tx/0x5b13c5e18b6cbf0969226fa4b3b4cd27fdadd1daa7d59690a215c179f927634b

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xbbd4ac61b96bbb191f47b66768f95e24126975ee7e6133ce37aa4f6ef24b292e

8-Internal swap (passed):
prize pool wallet received ETH
https://testnet.bscscan.com/address/0x7ed37dfd3167dc26d2666dceaca22eaf1f53aaea#internaltx

MANUAL TESTING

Low Risk Issue

Issue: redundant code at _transfer function

Type : **Code Smell**

Function:_transfer

Line: 731-738

Severity: **Low**

Overview: since **_totalFeesOnBuy** is equal to **FIGMAtokensburnFundingFeeOnBuy** and **_totalFeesOnSell** is equal to **FBRprizepoolFundingFeeOnSell** there is no need to calculate **marketingShare**

```
uint256 totalFee = totalFeesOnBuy + totalFeesOnSell; //@audit redundant

uint256 marketingShare = FIGMAtokensburnFundingFeeOnBuy +
    FBRprizepoolFundingFeeOnSell; //@audit redundant

//@audit marketingShare is always 100%
if (marketingShare > 0) {
    uint256 marketingTokens = (contractTokenBalance *
        marketingShare) / totalFee;
    swapAndSendMarketing(marketingTokens);
}

swapping = false;
}
```

MANUAL TESTING

Low Risk Issue

Recommendations

To address the issue identified in this audit, we recommend the following:

- use **swapAndSendMarketing** directly with contract's token balance

```
if (
    canSwap &&
    !swapping &&
    to == pancakeswapV2Pair &&
    totalFeesOnBuy + totalFeesOnSell > 0 &&
    swapEnabled
) {

    swapping = true;
    swapAndSendMarketing(balanceOf(address(this)));
    swapping = false;

}
```

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