auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

ETC WEB3

DATED : 18 JAN 23'

AUDIT SUMMARY

Project name – ETC WEB3

Date: 18 January , 2023

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

Audit Status: Passed (Contract is developed by Pinksale safu dev)

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	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- Goerli:
all tests were done on Goerli network, each test has its transaction has attached to it.

3- Slither : Static Analysis

TESTNET LINKS

All tests were done using this contract, tests are done on goerli

https://goerli.etherscan.io/token/0xc4f4a4be51c341733e2aa6447bf15ed53aae6ed1#code

Token Address:
0x0c8f1F10fFfdF8eD00903DD99ce185b4613F7eC5

Checksum:
2413fb3709b05939f04cf2e92f7d0897fc2596f9ad0b8a9ea855c7bfebaae892

Deployer:
0xE34993B3f4F8780f27D05F7F138d787A3611B82A

Owner:
0xE34993B3f4F8780f27D05F7F138d787A3611B82A

TOKEN OVERVIEW

Fees:

Buy Fees: 3%

Sell Fees: 3%

Transfer Fees: 0%

Fees Privilige: None

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount:NO

Blacklist: No

Other Priviliges: none

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

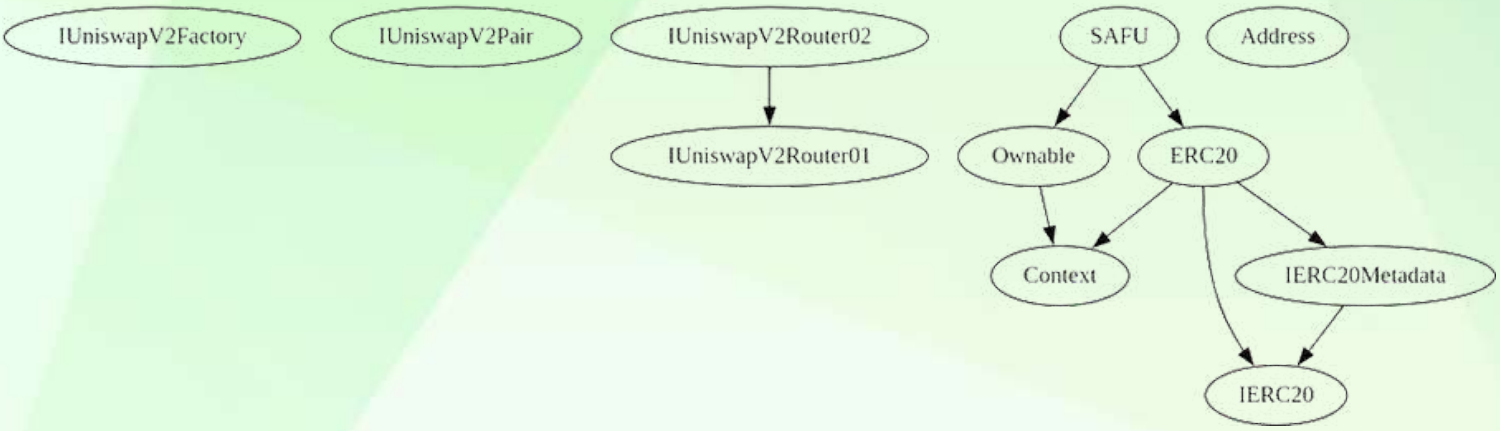
CLASSIFICATION OF RISK

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	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to change taxes (3% tax on buys/sells and 0% on transfers)
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to set max buy/sell/transfer amounts
- Owner is not able to disable trades
- 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 ❗
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 ❗

└	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 ❗	
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 ❗

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 🔒	🔴	
└| _approve	Internal 🔒	🔴	
└| _beforeTokenTransfer	Internal 🔒	🔴	
└| _afterTokenTransfer	Internal 🔒	🔴	

SAFU	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO ❗
└	changeMarketingWallet	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	swapAndLiquify	Private 🔐	🛑	
└	swapAndSendMarketing	Private 🔐	🛑	

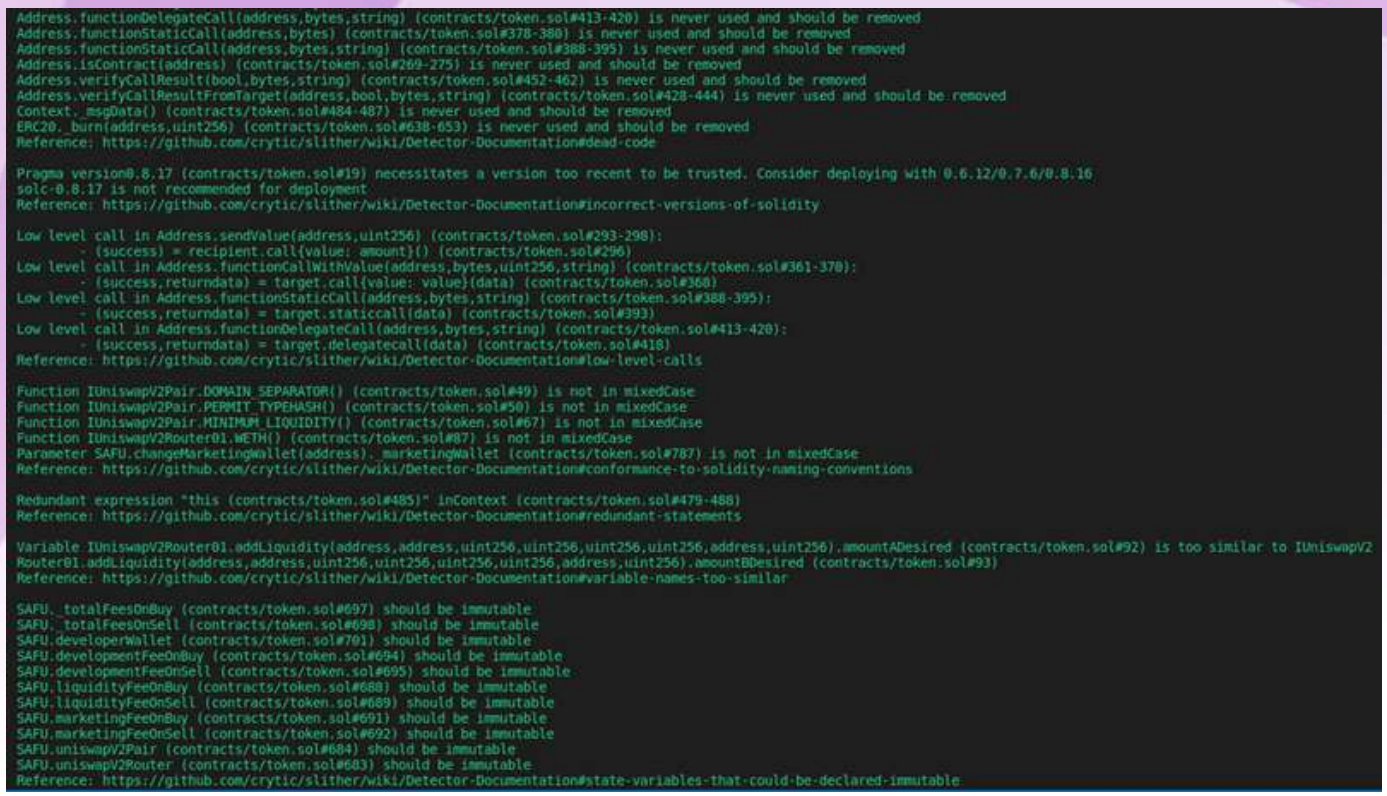
Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
SAFU.constructor().router (contracts/token.sol#714) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

SAFU.swapAndLiquify(uint256) (contracts/token.sol#865-894) ignores return value by uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xde
ad),block.timestamp) (contracts/token.sol#884-891)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Reentrancy in SAFU._transfer(address,address,uint256) (contracts/token.sol#797-854):
        External calls:
        - swapAndLiquify(liquidityTokens) (contracts/token.sol#825)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (contracts/token.sol#875-880)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (contracts/token.sol#884-891)
        - swapAndSendMarketing(marketingTokens) (contracts/token.sol#830)
                - (success) = recipient.call{value: amount}() (contracts/token.sol#296)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/token.sol#903-908)
                - address(marketingWallet).sendValue(marketingBalance) (contracts/token.sol#915)
                - address(developerWallet).sendValue(developmentBalance) (contracts/token.sol#916)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (contracts/token.sol#825)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (contracts/token.sol#884-891)
        - swapAndSendMarketing(marketingTokens) (contracts/token.sol#830)
                - (success) = recipient.call{value: amount}() (contracts/token.sol#296)
        Event emitted after the call(s):
        - SwapAndSendMarketing(tokenAmount,newBalance) (contracts/token.sol#918)
                - swapAndSendMarketing(marketingTokens) (contracts/token.sol#830)
        - Transfer(sender,recipient,amount) (contracts/token.sol#621)
                - super._transfer(from,address(this),fees) (contracts/token.sol#850)
        - Transfer(sender,recipient,amount) (contracts/token.sol#621)
                - super._transfer(from,to,amount) (contracts/token.sol#853)
Reentrancy in SAFU.swapAndLiquify(uint256) (contracts/token.sol#865-894):
        External calls:
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (contracts/token.sol#875-880)
        - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (contracts/token.sol#884-891)
        External calls sending eth:
        - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (contracts/token.sol#884-891)
        Event emitted after the call(s):
        - SwapAndLiquify(half,newBalance,otherHalf) (contracts/token.sol#893)
Reentrancy in SAFU.swapAndSendMarketing(uint256) (contracts/token.sol#896-919):
        External calls:
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/token.sol#903-908)
        - address(marketingWallet).sendValue(marketingBalance) (contracts/token.sol#915)
        - address(developerWallet).sendValue(developmentBalance) (contracts/token.sol#916)
        Event emitted after the call(s):
        - SwapAndSendMarketing(tokenAmount,newBalance) (contracts/token.sol#918)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address._revert(bytes,string) (contracts/token.sol#464-476) uses assembly
        - INLINE ASM (contracts/token.sol#469-472)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address._revert(bytes,string) (contracts/token.sol#464-476) is never used and should be removed
Address.functionCall(address,bytes) (contracts/token.sol#318-320) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/token.sol#328-334) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/token.sol#347-353) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/token.sol#361-370) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/token.sol#403-405) is never used and should be removed
```

STATIC ANALYSIS



```
Address.functionDelegateCall(address,bytes,string) (contracts/token.sol#413-420) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/token.sol#378-380) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/token.sol#388-395) is never used and should be removed
Address.isContract(address) (contracts/token.sol#269-275) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (contracts/token.sol#452-462) is never used and should be removed
Address.verifyCallResultFromTarget(address,bool,bytes,string) (contracts/token.sol#428-444) is never used and should be removed
Context._msgData() (contracts/token.sol#484-487) is never used and should be removed
ERC20._burn(address,uint256) (contracts/token.sol#638-653) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.8.17 (contracts/token.sol#19) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.17 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#293-298):
        - (success) = recipient.call{value: amount}() (contracts/token.sol#296)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/token.sol#361-370):
        - (success,returndata) = target.call{value: value}(data) (contracts/token.sol#368)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/token.sol#388-395):
        - (success,returndata) = target.staticcall(data) (contracts/token.sol#393)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/token.sol#413-420):
        - (success,returndata) = target.delegatecall(data) (contracts/token.sol#418)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/token.sol#49) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/token.sol#50) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/token.sol#67) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/token.sol#87) is not in mixedCase
Parameter SAFU.changeMarketingWallet(address)._marketingWallet (contracts/token.sol#787) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/token.sol#485)" inContext (contracts/token.sol#479-488)
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#92) is too similar to IUniswapV2
Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/token.sol#93)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

SAFU._totalFeesOnBuy (contracts/token.sol#697) should be immutable
SAFU._totalFeesOnSell (contracts/token.sol#698) should be immutable
SAFU.developerWallet (contracts/token.sol#701) should be immutable
SAFU.developmentFeeOnBuy (contracts/token.sol#694) should be immutable
SAFU.developmentFeeOnSell (contracts/token.sol#695) should be immutable
SAFU.liquidityFeeOnBuy (contracts/token.sol#688) should be immutable
SAFU.liquidityFeeOnSell (contracts/token.sol#689) should be immutable
SAFU.marketingFeeOnBuy (contracts/token.sol#691) should be immutable
SAFU.marketingFeeOnSell (contracts/token.sol#692) should be immutable
SAFU.uniswapV2Pair (contracts/token.sol#684) should be immutable
SAFU.uniswapV2Router (contracts/token.sol#683) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => No issues found

FUNCTIONAL TESTING

Functionality tests for ERC20 tokens includes:
- adding liquidity
- buying / selling /transferring (for non-excluded wallets_
- tax conversion and add liquidity

1- Adding Liquidity:

liquidity added on Uniswap v2:

https://goerli.etherscan.io/tx/0x0aa0465b675f52be78fed7b05b63bff7a3ee505677c6d0b3def1bb4e0ab60252
no issue were found on adding liquidity.

2- Buying from a non-excluded wallet:

https://goerli.etherscan.io/tx/0xebbeba7901d8683b14a614a278b5af62380ff855d6a355bee4b086b452a832ce
taxes sent to the contract (accumulated)

3- Selling from a non-excluded wallet

https://goerli.etherscan.io/tx/0xbd7ac96d8578e01685a3a2b66fcf62cc682670f6e42559319f84db99b3fd6b78

FUNCTIONAL TESTING

4- Tax conversions:

development wallet and marketing wallets both received ETH after swap and liquidfy

development wallet :
https://goerli.etherscan.io/address/0x76727C9dc6d19257e76582cc9B47a65a43e12b3d#internaltx

marketing wallet:

https://goerli.etherscan.io/address/0x9B79C54B54493153dc25f38C59884284707caf74
dead wallet received 0.003354159693099419 lp share tokens (Auto liquidity works)

MANUAL TESTING

Critical Risk Findings:

NO RISKS WERE FOUND IN THE CONTRACT

Social Media Overview

Here are the Social Media Accounts of ETC WEB3

https://t.me/ET_COIN

https://twitter.com/ETCOINWEB3

https://www.etcoinweb3.com

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