AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

SAFUU 2.0

DATED : 07 Aug 24'

MANUAL TESTING

Centralization – The owner can set fees of more than 100%.
Severity: High
Function: setFees
Status: Open

Overview:
In some smart contracts, the owner or creator of the contract can set fees for certain actions or operations within the contract. These fees can be used to cover the cost of running the contract, such as paying for gas fees or compensating the contract's owner for their time and effort in developing and maintaining the contract.

```
function setFees(uint256 _BUYTax, uint256 _SELLTax) external onlyOwner {
  require(
   _BUYTax >= MIN_FEE_RATE && _BUYTax <= MAX_FEE_RATE &&
   _SELLTax >= MIN_FEE_RATE && _SELLTax <= MAX_FEE_RATE,
   "MAX_FEE_RATE or MIN_FEE_RATE not matched"
  );

  BUYTax    = _BUYTax;
  SELLTax   = _SELLTax;
 }
```

Suggestion: The owner can set buy and sell tax of more than 100% in the contract which is not recommended. The fees in the contract should not be more than 25% so that the user does not lose his token values.

AUDIT SUMMARY

Project name – SAFUU 2.0

Date: 07 Aug, 2024

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

Audit Status: **High risk major flag**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	2	3	1
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 conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:

https://testnet.bscscan.com/address/0xfcf8e661354c81615f5ceaaadcc8dbfcab41eaa5#writeContract

Token Information

Token Address:
0xC53Eb9eCc247465e2E1fb16B2EdE3A6446053170

Name: SAFUU 2.0

Symbol: SAFUU

Decimals: 18

Network: Ethereum Network

Token Type: ERC-20

Owner: 0x5fA28b1227432FA1BC4bd88073f37063381c04dC

Deployer: 0x5fA28b1227432FA1BC4bd88073f37063381c04dC

Token Supply: 25,890,650

Checksum: 76d9051a5c26a4a92f7159096df9df6d

Testnet:
https://testnet.bscscan.com/address/0xfcf8e661354c81615f5ceaaadcc8dbfcab41eaa5#writeContract

TOKEN OVERVIEW

Buy Fee: 100%

Sell Fee: 100%

Transfer Fee: 0-0%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

AUDIT METHODOLOGY

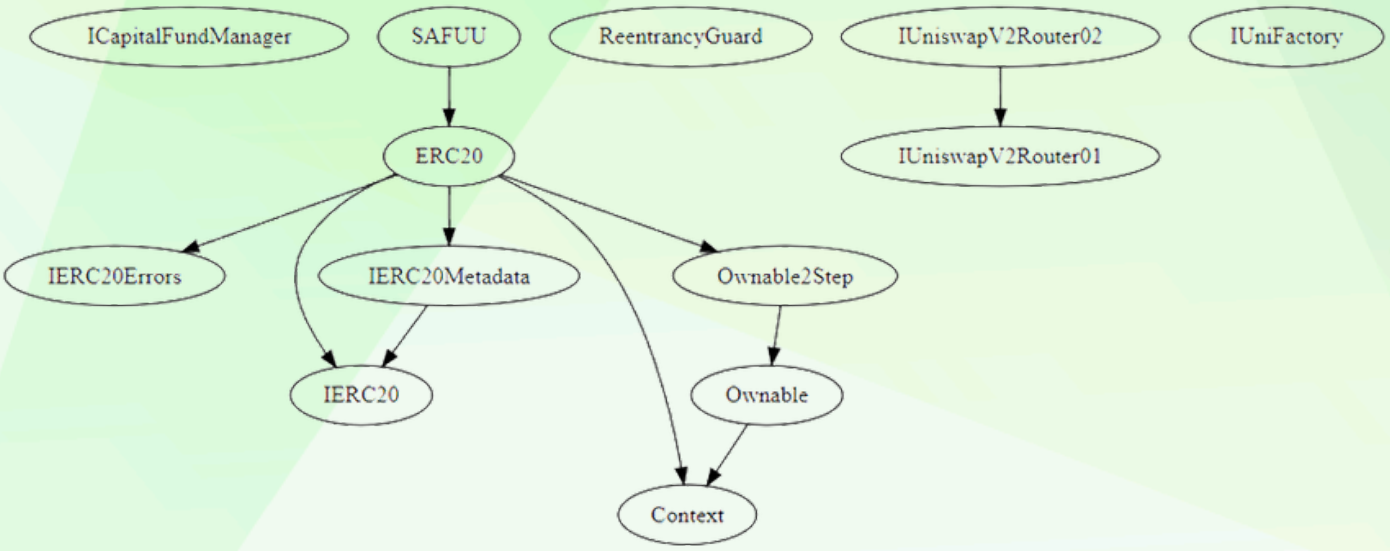
The auditing process will follow a routine as special considerations by Auditace:

- 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

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

INHERITANCE TREE



POINTS TO NOTE

- The owner can transfer/renounce the ownership.
- The owner can set treasury/staking/capital fund address.
- The owner can set MinTokens for Swap greater than 0.
- The owner can set a buy/sell tax of more than 100%
- The owner can setFeeDistribution.
- The owner can rescue ETH.
- The owner can remove/add a pair address.
- The owner can setFeeExempt.
- The owner can setTimeExpire.

STATIC ANALYSIS

```
INFO:Detectors:
ERC20.transfer(address,uint256).owner (SAFUU.sol#470) shadows:
        - Ownable.owner() (SAFUU.sol#94-96) (function)
ERC20.allowance(address,address).owner (SAFUU.sol#476) shadows:
        - Ownable.owner() (SAFUU.sol#94-96) (function)
ERC20.approve(address,uint256).owner (SAFUU.sol#482) shadows:
        - Ownable.owner() (SAFUU.sol#94-96) (function)
ERC20._approve(address,address,uint256).owner (SAFUU.sol#675) shadows:
        - Ownable.owner() (SAFUU.sol#94-96) (function)
ERC20._approve(address,address,uint256,bool).owner (SAFUU.sol#680) shadows:
        - Ownable.owner() (SAFUU.sol#94-96) (function)
ERC20._spendAllowance(address,address,uint256).owner (SAFUU.sol#694) shadows:
        - Ownable.owner() (SAFUU.sol#94-96) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
```

```
INFO:Detectors:
Context._msgData() (SAFUU.sol#62-64) is never used and should be removed
ERC20._burn(address,uint256) (SAFUU.sol#667-672) is never used and should be removed
ReentrancyGuard._nonReentrantAfter() (SAFUU.sol#188-192) is never used and should be removed
ReentrancyGuard._nonReentrantBefore() (SAFUU.sol#178-186) is never used and should be removed
ReentrancyGuard._reentrancyGuardEntered() (SAFUU.sol#195-197) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
INFO:Detectors:
Function IUniswapV2Router01.WETH() (SAFUU.sol#202) is not in mixedCase
Parameter ERC20.setTreasuryAddress(address)._newTreasuryWallet (SAFUU.sol#413) is not in mixedCase
Parameter ERC20.setMinTokensForSwap(uint256)._newMinTokens (SAFUU.sol#418) is not in mixedCase
Parameter ERC20.setFees(uint256,uint256)._BUYTax (SAFUU.sol#423) is not in mixedCase
Parameter ERC20.setFees(uint256,uint256)._SELLTax (SAFUU.sol#423) is not in mixedCase
Parameter ERC20.setFeeDistribution(uint256,uint256,uint256)._TREASURYPart (SAFUU.sol#434) is not in mixedCase
Parameter ERC20.setFeeDistribution(uint256,uint256,uint256)._SHERIFFPart (SAFUU.sol#434) is not in mixedCase
Parameter ERC20.setFeeDistribution(uint256,uint256,uint256)._BurnPart (SAFUU.sol#434) is not in mixedCase
Function ERC20.SWAPBack() (SAFUU.sol#568-627) is not in mixedCase
Function ERC20.SWAPTOKENS(uint256) (SAFUU.sol#629-641) is not in mixedCase
Parameter ERC20.checkFeeExempt(address)._addr (SAFUU.sol#733) is not in mixedCase
Parameter ERC20.setFeeExempt(address,bool)._addr (SAFUU.sol#739) is not in mixedCase
Parameter ERC20.setFeeExempt(address,bool)._value (SAFUU.sol#739) is not in mixedCase
Parameter ERC20.setTimeExpire(uint256)._value (SAFUU.sol#745) is not in mixedCase
Variable ERC20.UniswapV2Router (SAFUU.sol#343) is not in mixedCase
Variable ERC20.UniswapV2Pair (SAFUU.sol#346) is not in mixedCase
Variable ERC20.InSWAP (SAFUU.sol#347) is not in mixedCase
Variable ERC20.MIN_TOKENS_FOR_SWAP (SAFUU.sol#348) is not in mixedCase
Variable ERC20.TOTAL_ETH_VALUE_CAPITAL_FUND (SAFUU.sol#351) is not in mixedCase
Variable ERC20.TREASURY (SAFUU.sol#353) is not in mixedCase
Variable ERC20.TREASURYPart (SAFUU.sol#355) is not in mixedCase
Variable ERC20.BUYTax (SAFUU.sol#357) is not in mixedCase
Variable ERC20.SELLTax (SAFUU.sol#358) is not in mixedCase
Variable ERC20.STAKING (SAFUU.sol#367) is not in mixedCase
Variable ERC20.CAPITAL_FUND_ADDRESS (SAFUU.sol#368) is not in mixedCase
Variable ERC20.CAPITAL_FUND (SAFUU.sol#369) is not in mixedCase
Variable ERC20.SHERIFFPart (SAFUU.sol#371) is not in mixedCase
Variable ERC20.BurnPart (SAFUU.sol#372) is not in mixedCase
Variable ERC20.TOTAL_ETH_AVAILABLE_BNB (SAFUU.sol#377) is not in mixedCase
Variable ERC20.TOTAL_ETH_Treas (SAFUU.sol#378) is not in mixedCase
Variable ERC20.TOTAL_ETH_Flash (SAFUU.sol#379) is not in mixedCase
Variable ERC20.DEAD (SAFUU.sol#381) is not in mixedCase
Variable ERC20.SWAPPING (SAFUU.sol#385) is not in mixedCase
Modifier ERC20.InSWAPPING() (SAFUU.sol#388-392) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

1- Set Fees (passed):

https://testnet.bscscan.com/tx/0x44f38c527fdbaabe44689aa2fc07fddc506158ea0bdc18a19720ef8685f3951b

2- Set Staking (passed):

https://testnet.bscscan.com/tx/0xb6b6c5595ca242dac9e296de2ae99ef32887001593d24171d2b0d18a2054cc33

3- Set Treasury Address (passed):

https://testnet.bscscan.com/tx/0x5eacbd5dc4fedadd6c98ce88731d9a33149b831c2e746b506bbe87ccfda9313d

4- Set Capital Fund (passed):

https://testnet.bscscan.com/tx/0x870691de2d8617ce9d7c20afcb2b664ef26defcd364a9273c246b4c365d47ef9

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	2
Low-Risk	3
Gas Optimization / Suggestions	1

MANUAL TESTING

Centralization – The owner can set fees of more than 100%.
Severity: High
Function: setFees
Status: Open

Overview:
In some smart contracts, the owner or creator of the contract can set fees for certain actions or operations within the contract. These fees can be used to cover the cost of running the contract, such as paying for gas fees or compensating the contract's owner for their time and effort in developing and maintaining the contract.

```
function setFees(uint256 _BUYTax, uint256 _SELLTax) external onlyOwner {
  require(
   _BUYTax >= MIN_FEE_RATE && _BUYTax <= MAX_FEE_RATE &&
   _SELLTax >= MIN_FEE_RATE && _SELLTax <= MAX_FEE_RATE,
   "MAX_FEE_RATE or MIN_FEE_RATE not matched"
  );

  BUYTax    = _BUYTax;
  SELLTax   = _SELLTax;
 }
```

Suggestion: The owner can set buy and sell tax of more than 100% in the contract which is not recommended. The fees in the contract should not be more than 25% so that the user does not lose his token values.

MANUAL TESTING

Centralization – The owner can regain ownership.

Severity: Medium

Function: pendingOwner

Status: Open

Overview:

The owner can regain ownership after transferring it with the following steps:

1. Call lock function to set previous owner to the own address
2. Call unlock function to get ownership back
3. Transfer/renounce ownership
4. Call unlock function to get ownership back

```
abstract contract Ownable2Step is Ownable {
  address private _pendingOwner;

  event OwnershipTransferStarted(address indexed previousOwner, address indexed newOwner);


  function pendingOwner() public view virtual returns (address) {
    return _pendingOwner;
  }


  function transferOwnership(address newOwner) public virtual override onlyOwner {
    _pendingOwner = newOwner;
    emit OwnershipTransferStarted(owner(), newOwner);
  }
```

MANUAL TESTING

```
  function _transferOwnership(address newOwner) internal virtual
override {
    delete _pendingOwner;
    super._transferOwnership(newOwner);
  }


  function acceptOwnership() public virtual {
    address sender = _msgSender();
    if (pendingOwner() != sender) {
      revert OwnableUnauthorizedAccount(sender);
    }
    _transferOwnership(sender);
  }
}
```

Suggestion: Make sure to set the previous ownership back to address zero after using the unlock function.

MANUAL TESTING

Centralization – Missing Require Check.

Severity: Medium

Function: setStaking,setCapitalFund, setTreasuryAddress.

Status: Open

Overview:

The owner can set any arbitrary address excluding zero address as this is not recommended because if the owner will set the address to the contract address, then the Eth will not be sent to that address and the transaction will fail and this will lead to a potential honeypot in the contract.

```
function setTreasuryAddress(address _newTreasuryWallet) external onlyOwner {
    require(_newTreasuryWallet != address(0), "Zero Wallets");
    TREASURY = _newTreasuryWallet;
  }
function setStaking(address pAddress) external onlyOwner {
    STAKING = pAddress;//can be set to zero to deactivate sending to Staking
  }

  function setCapitalFund(address pAddress) external onlyOwner {
    CAPITAL_FUND_ADDRESS = pAddress;
    if(CAPITAL_FUND_ADDRESS != address(0)) CAPITAL_FUND =
ICapitalFundManager(CAPITAL_FUND_ADDRESS);//we allow the address to be
address(0) so you can deactivate the CAPITAL_FUND deposit
  }
```

Suggestion: It is recommended that the address should not be able to set as a contract address.

MANUAL TESTING

Centralization – Missing Events

Severity: Low

Subject: Missing Events

Status: Open

Overview:

They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function setTreasuryAddress(address _newTreasuryWallet) external onlyOwner {
    require(_newTreasuryWallet != address(0), "Zero Wallets");
    TREASURY = _newTreasuryWallet;
  }
function setMinTokensForSwap(uint256 _newMinTokens) external onlyOwner {
    require(_newMinTokens > 0, "New minimum tokens must be greater than 0");
    MIN_TOKENS_FOR_SWAP = _newMinTokens;
  }
function setFees(uint256 _BUYTax, uint256 _SELLTax) external onlyOwner {
    require(
      _BUYTax >= MIN_FEE_RATE && _BUYTax <= MAX_FEE_RATE &&
      _SELLTax >= MIN_FEE_RATE && _SELLTax <= MAX_FEE_RATE,
      "MAX_FEE_RATE or MIN_FEE_RATE not matched"
    );

    BUYTax    = _BUYTax;
    SELLTax   = _SELLTax;
  }
function setFeeDistribution(uint256 _TREASURYPart, uint256 _SHERIFFPart, uint256
_BurnPart) external onlyOwner {
    require(
      (_TREASURYPart + _SHERIFFPart + _BurnPart) == 1000, "3 sums not matched"
    );

    TREASURYPart  = _TREASURYPart;
    SHERIFFPart  = _SHERIFFPart;
    BurnPart    = _BurnPart;
  }
```

Suggestion:

Emit an event for critical changes.

MANUAL TESTING

Centralization – Local variable Shadowing

Severity: Low

Subject: Variable Shadowing

Status: Open

Overview:

```
function transfer(address to, uint256 value) public virtual returns (bool) {
    address owner = _msgSender();
    _transfer(owner, to, value);
    return true;
  }
function allowance(address owner, address spender) public view virtual
returns (uint256) {
    return _allowances[owner][spender];
  }


  function approve(address spender, uint256 value) public virtual returns (bool)
{
    address owner = _msgSender();
    _approve(owner, spender, value);
    return true;
  }
```

Suggestion:

Rename the local variables that shadow another component.

MANUAL TESTING

Centralization – Missing Zero Address

Severity: Low

Subject: Zero Check

Status: Open

Overview:

functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.

```
function setStaking(address pAddress) external onlyOwner {
    STAKING = pAddress;//can be set to zero to deactivate sending to Staking
 }
```

Suggestion:

It is suggested that the address should not be zero or dead.

MANUAL TESTING

Optimization

Severity: Optimization

Subject: Remove unused code.

Status: Open

Overview:

Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them.

```
function _burn(address account, uint256 value) internal {
      if (account == address(0)) {
         revert ERC20InvalidSender(address(0));
      }
      _simpleUpdate(account, address(0), value);
   }
```

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