RugFreeCoins Audit

Nobodies Finance Staking Smart Contract Security Audit

November 21st, 2022

Contents

Audit details 1

Disclaimer 2

Background 3

Target market and the concept 4

Potential to grow with score points 5

Total Points 5

Contract details 6

Contract code function details 7

Contract description table 9

Security issue checking status 15

Owner privileges 17

Audit conclusion 20

Audit details

Audited project
Nobodies Finance Staking Contract

Contract Address

0xA002EaB9fFfad1366A4E561349B95D410c3F4a38

Client contact

Nobodies Finance Team

Blockchain

Binance Smart chain

Project website

https://.nobodies.finance

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Rugfreecoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Rugfreecoins) owe no duty of care towards you or any other person, nor does Rugfreecoins make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Rugfreecoins hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Rugfreecoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Rugfreecoins, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

Rugfreecoins was commissioned by the Nobodies Finance Team to perform an audit of the smart contract.

https://bscscan.com/address/0xA002EaB9fFfad1366A4E561349B95D410c3F4a38

The focus of this audit is to verify that the smart contract is secure, resilient, and working according to the specifications.

The information in this report should be used to understand the risk exposure of the smart contract, project feasibility, and long-term sustainability, and as a guide to improving the smart contract's security posture by remediating the identified issues.

Target market and the concept

Target market

- Anyone who’s interested in the Crypto space with long-term investment plans.
- Anyone who’s ready to earn a passive income by holding tokens.
- Anyone who’s interested in trading tokens.
- Anyone who’s ready to staking and receive rewards.
- Anyone who’s interested in taking part in the Nobodies Finance ecosystem.
- Anyone who’s interested in taking part in the future plans of Nobodies Finance Token.
- Anyone who’s interested in making financial transactions with any other party using Nobodies Finance Token as the currency.

Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	9/10
3	Information quality	9/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	9/10
7	Impact on the business	9/10
8	Preparing for the future	10/10
9	Smart contract security	10/10
10	Smart contract functionality assessment	10/10
Total Points		**9.3/10**

Contract details

Token contract details for 21st of November 2022

Contract name	NoboNFTMine
Contract address	0x2aC201c0b1A2b9f6326fc218baB2aaCd6C24B58d
Reward token address	0xee5c28b190ba35a6880fedd2d8dd6561366f9e33
Pledge token address	0xee5c28b190ba35a6880fedd2d8dd6561366f9e33
NFT address	0xb8963da91644dc0919612017dc135e3ecfa7298c
Base account	0x6318493cbd615fd34b8da2d34409978184ea2108
SAFU dev address	0xa799effde45c5344866642f4a3cf8b5288aad390
Contract deployer address	0x9Bc9B008043a8063A2EbA7c602A196548b1FCAac
Contract’s current owner address	0x9bc9b008043a8063a2eba7c602a196548b1fcaac

Contract code function details

No	Category	Item	Result
1	Coding conventions	BRC20 Token standards	pass
		compile errors	pass
		Compiler version security	pass
		visibility specifiers	pass
		Gas consumption	pass
		SafeMath features	pass
		Fallback usage	pass
		tx.origin usage	pass
		deprecated items	pass
		Redundant code	pass
		Overriding variables	pass
2	Function call audit	Authorization of function call	pass
		Low level function (call/delegate call) security	pass
		Returned value security	pass
		Self-destruct function security	pass
3	Business security	Access control of owners	pass
		Business logics	pass
		Business implementations	pass
4	Integer overflow/underflow		pass
5	Reentrancy		pass
6	Exceptional reachable state		pass
7	Transaction ordering dependence		pass
8	Block properties dependence		pass
9	Pseudo random number generator (PRNG)		pass
10	DoS (Denial of Service)		pass
11	Token vesting implementation		pass
12	Fake deposit		pass

13	**Event security**		**pass**

Contract description table

The below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions, and implementations with their visibility and mutability.

Contract	Type	Bases		
└	Function Name	Visibility	Mutability	Modifiers
IBEP20	**Interface**			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC165	**Interface**			
└	supportsInterface	External ❗		NO❗
IERC721	**Interface**	**IERC165**		
└	balanceOf	External ❗		NO❗
└	ownerOf	External ❗		NO❗

└	safeTransferFrom	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	approve	External ❗	🛑	NO❗
└	getApproved	External ❗		NO❗
└	setApprovalForAll	External ❗	🛑	NO❗
└	isApprovedForAll	External ❗		NO❗
└	safeTransferFrom	External ❗	🛑	NO❗
IERC721Metadata	**Interface**	**IERC721**		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	tokenURI	External ❗		NO❗
IERC721 Enumerable	**Interface**	**IERC721**		
└	totalSupply	External ❗		NO❗
└	tokenOfOwnerByIndex	External ❗		NO❗
└	tokenByIndex	External ❗		NO❗
IERC721Receiver	**Interface**			
└	onERC721Received	External ❗	🛑	NO❗
NOriginNFT	**Interface**			

└	NIds	External ❗		NO❗
└	mint	External ❗	🛑	NO❗
└	getMintSpeed	External ❗		NO❗
SafeMath	**Library**			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
Address	**Library**			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	_functionCallWithValue	Private 🔐	🛑	

Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	**Implementation**	**Context**		
└		Internal 🔒	🛑	
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	geUnlockTime	Public ❗		NO❗
└	lock	Public ❗	🛑	onlyOwner
└	unlock	Public ❗	🛑	NO❗
CommonFunc	**Implementation**	**Ownable**		
└	transferSAFU	Public ❗	🛑	NO❗
└	setIsOpen	External ❗	🛑	NO❗
└	setBaseAccount	External ❗	🛑	NO❗
└	setTokenHoldInfo	External ❗	🛑	NO❗
└	setApproveToken	External ❗	🛑	NO❗
└	getTokenBack	External ❗	🛑	NO❗
└	getTokenHoldInfo	External ❗		NO❗

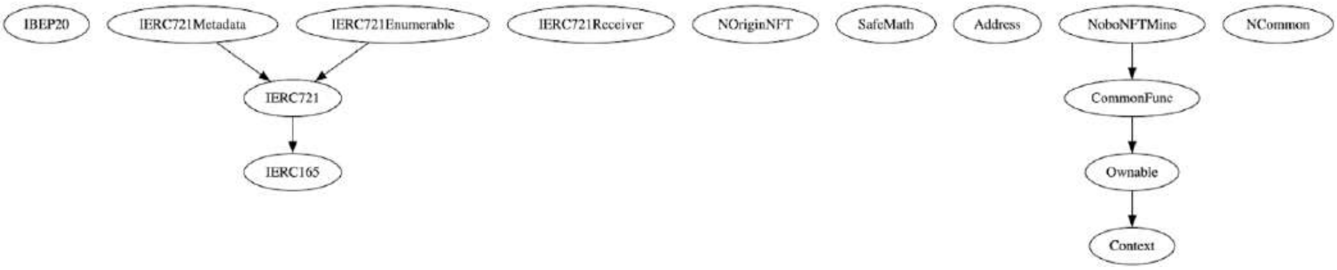
NCommon	**Library**			
└	random	Internal 🔒		
NoboNFTMine	**Implementation**	**CommonFunc**		
└	createTradeList	Public ❗️	🛑	NO❗️
└	setBaseAddr	Public ❗️	🛑	NO❗️
└	setRewardBase	Public ❗️	🛑	NO❗️
└	setNftPledgeTokenNeed	Public ❗️	🛑	NO❗️
└	setPledgePeriodLimit	Public ❗️	🛑	NO❗️
└	setPledgeAmountLimit	Public ❗️	🛑	NO❗️
└	getPledgePeriodLimit	Public ❗️		NO❗️
└	getPledgeAmountLimit	Public ❗️		NO❗️
└	update	Internal 🔒	🛑	
└	endContract	Internal 🔒	🛑	
└	getRewardNow	Internal 🔒		
└	getMineListInfo	Public ❗️		NO❗️
└	endMine	Public ❗️	🛑	NO❗️
└	renewMine	Public ❗️	🛑	NO❗️
└	addressArrayPush	Internal 🔒	🛑	
└	addressUserArrayPush	Internal 🔒	🛑	
└	addressArrayPop	Internal 🔒	🛑	

└	addressUserArrayPop	Internal 🔒	🛑	
└	getContractAddrArray	External ❗		NO❗
└	getUserContractAddrArray	External ❗		NO❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Inheritance Hierarchy



Security issue checking status

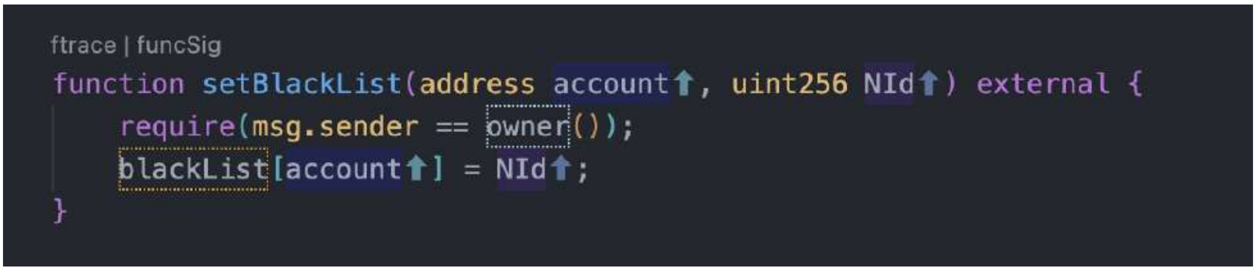
- **High severity issues**
  No High severity issues found

- **Medium severity issues**
  No medium severity issues found

- **Low severity issues**

Unused functions

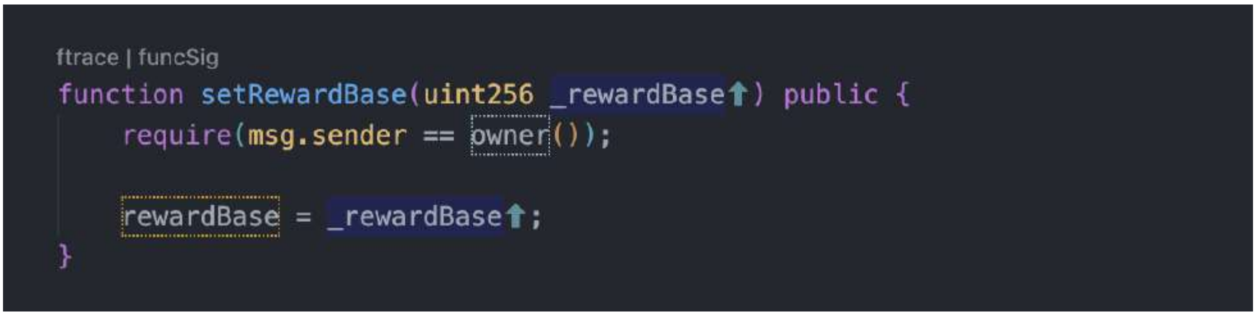
(Informed and fixed)



```
ftrace | funcSig
function setBlackList(address account, uint256 NId) external {
    require(msg.sender == owner());
    blackList[account] = NId;
}
```

- **Centralization Risk**

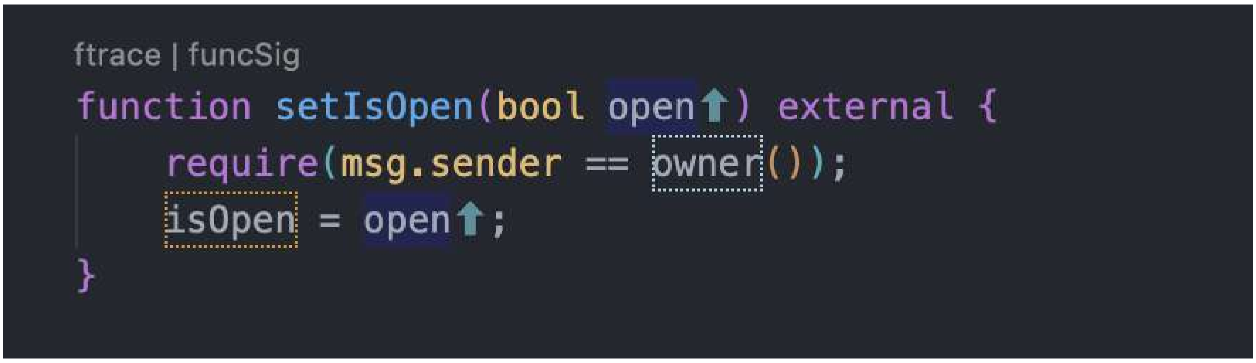
The owner can change users' reward amount by changing rewardBase value



```
ftrace | funcSig
function setRewardBase(uint256 _rewardBase) public {
    require(msg.sender == owner());

    rewardBase = _rewardBase;
}
```

The owner can enable/disable contract for trading at any time

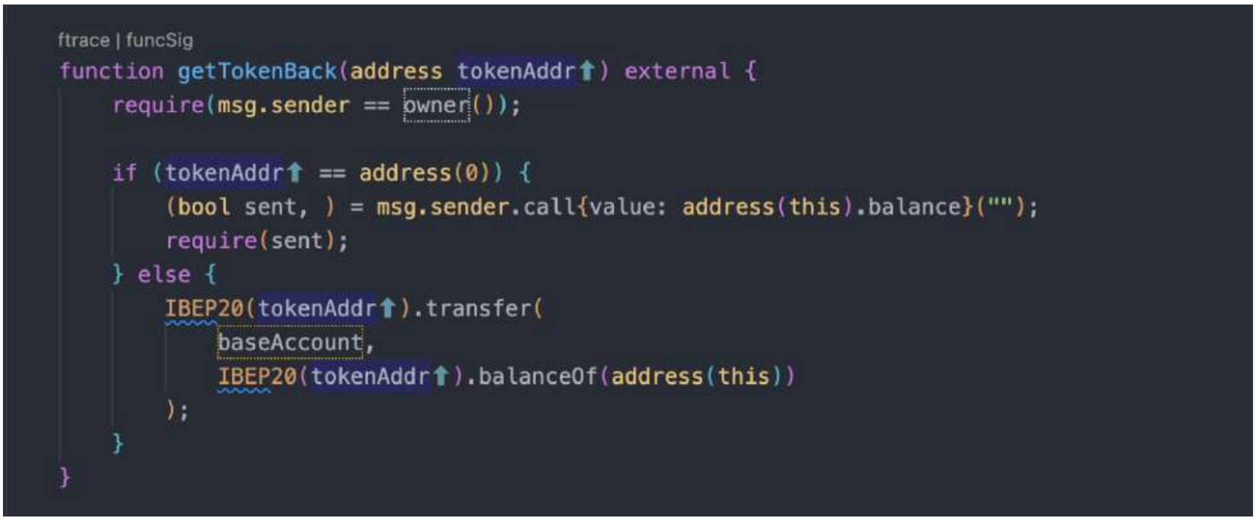
(Informed and fixed)



```
ftrace | funcSig
function setIsOpen(bool open) external {
    require(msg.sender == owner());
    isOpen = open;
}
```

The owner can get any BEP20 tokens from the contract event nobodies tokens

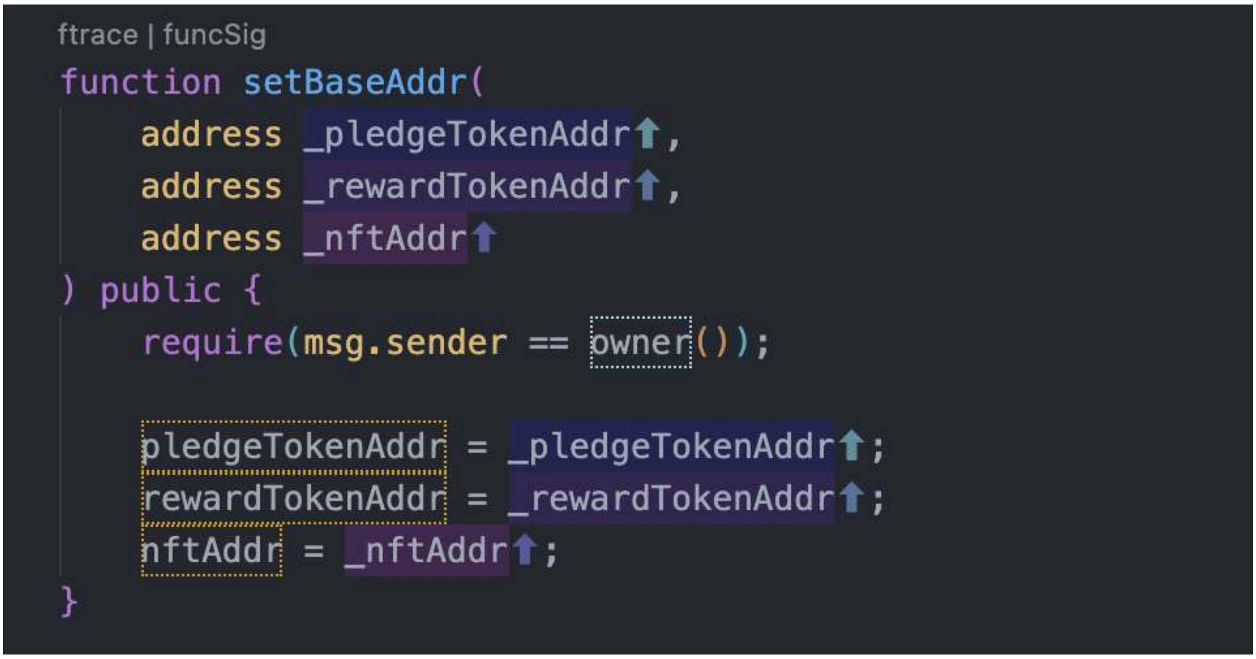
(Informed and fixed)



```
ftrace | funcSig
function getTokenBack(address tokenAddr) external {
    require(msg.sender == owner());

    if (tokenAddr == address(0)) {
        (bool sent, ) = msg.sender.call{value: address(this).balance}("");
        require(sent);
    } else {
        IBEP20(tokenAddr).transfer(
            baseAccount,
            IBEP20(tokenAddr).balanceOf(address(this))
        );
    }
}
```

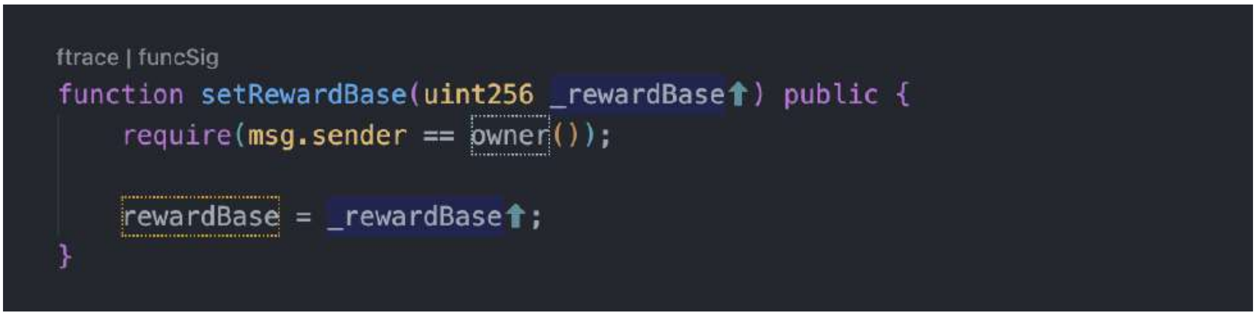
Owner privileges

- The owner can change all base addresses.



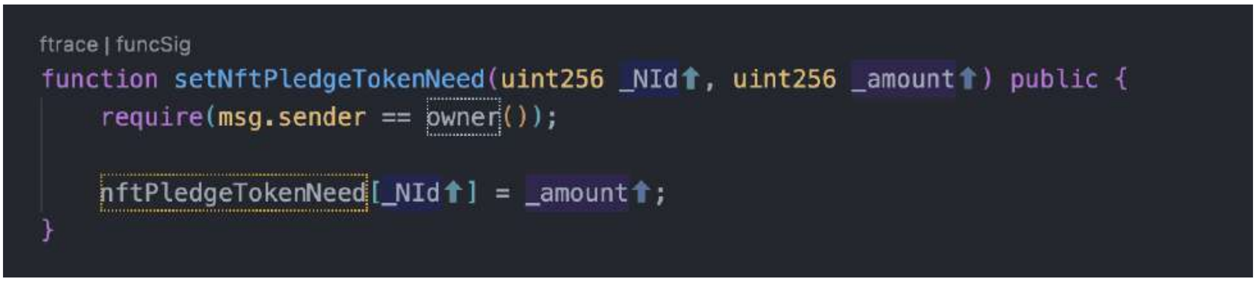
```
ftrace | funcSig
function setBaseAddr(
    address _pledgeTokenAddr,
    address _rewardTokenAddr,
    address _nftAddr
) public {
    require(msg.sender == owner());

    pledgeTokenAddr = _pledgeTokenAddr;
    rewardTokenAddr = _rewardTokenAddr;
    nftAddr = _nftAddr;
}
```

- The owner can change the reward base value



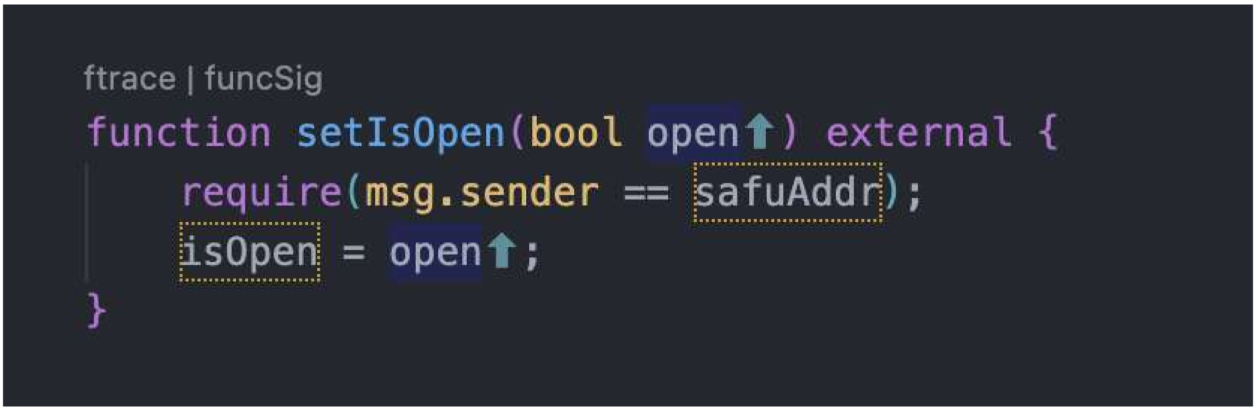
```
ftrace | funcSig
function setRewardBase(uint256 _rewardBase) public {
    require(msg.sender == owner());

    rewardBase = _rewardBase;
}
```

- The owner can change the required balance to create a trade list



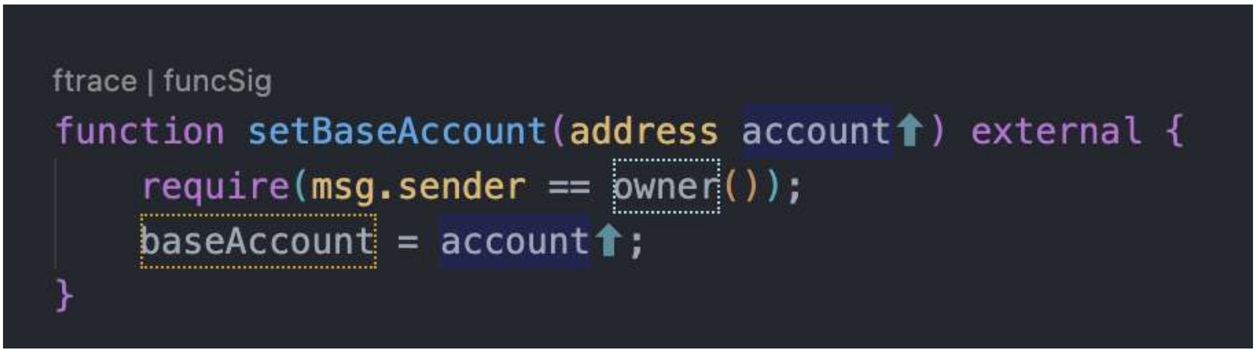
```
ftrace | funcSig
function setNftPledgeTokenNeed(uint256 _NId, uint256 _amount) public {
    require(msg.sender == owner());

    nftPledgeTokenNeed[_NId] = _amount;
}
```

- The safe dev can enable/disable the contract for trading at any time



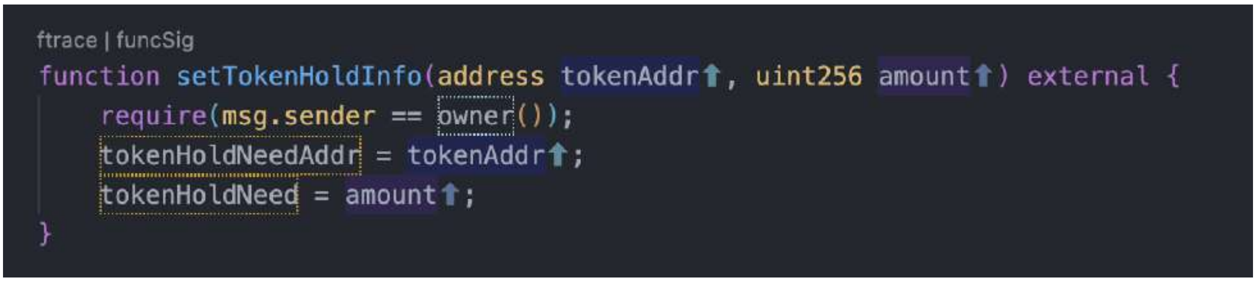
```
ftrace | funcSig
function setIsOpen(bool open) external {
    require(msg.sender == safuAddr);
    isOpen = open;
}
```

- The owner can change the base account, and the base account can get stuck tokens from the contract



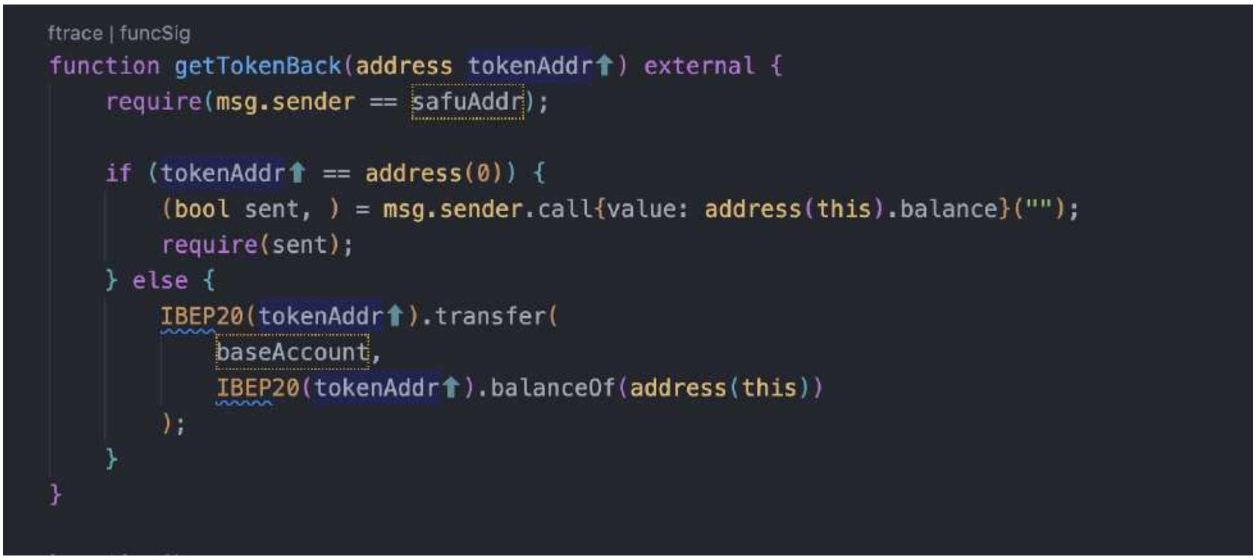
```
ftrace | funcSig
function setBaseAccount(address account) external {
    require(msg.sender == owner());
    baseAccount = account;
}
```

- The owner can change minimum token amount to use Dapp but the user can always call functions directly from the contract without this limitation.



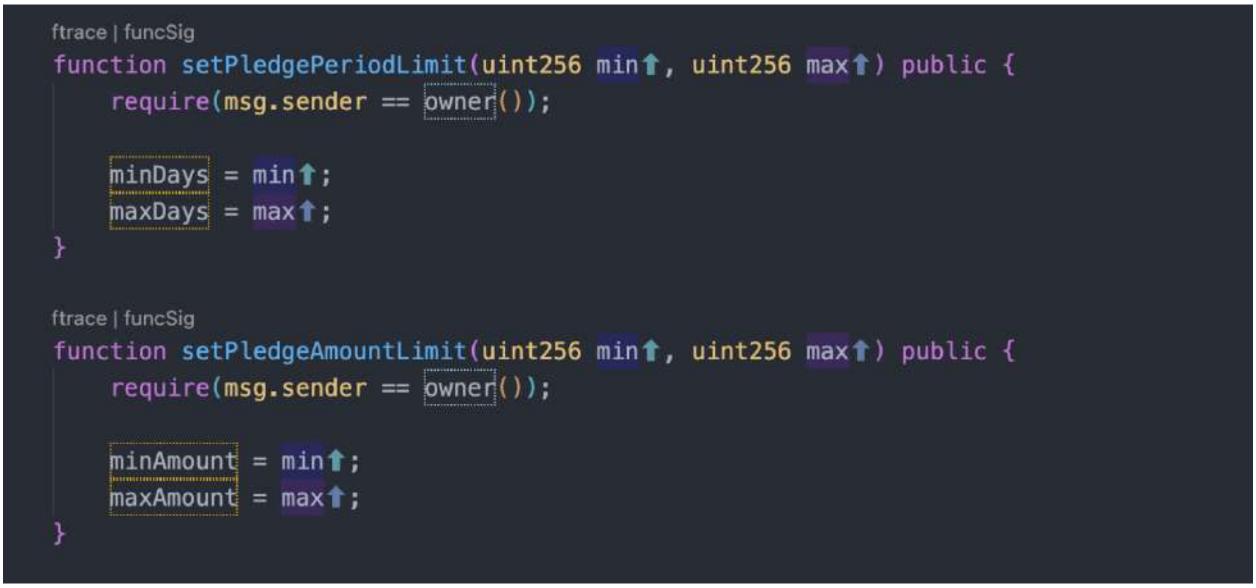
```
ftrace | funcSig
function setTokenHoldInfo(address tokenAddr, uint256 amount) external {
    require(msg.sender == owner());
    tokenHoldNeedAddr = tokenAddr;
    tokenHoldNeed = amount;
}
```

- The SAFU dev can get any BEP20 tokens from the contract



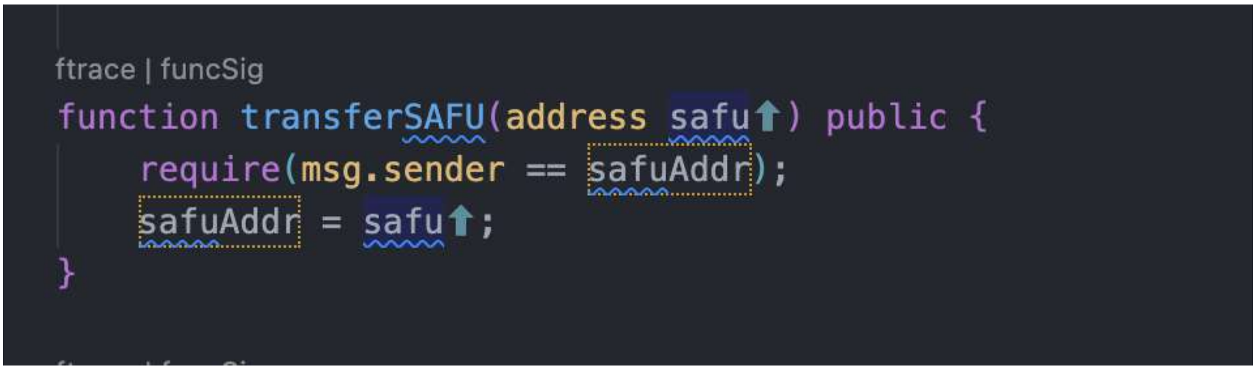
```
ftrace | funcSig
function getTokenBack(address tokenAddr) external {
    require(msg.sender == safuAddr);

    if (tokenAddr == address(0)) {
        (bool sent, ) = msg.sender.call{value: address(this).balance}("");
        require(sent);
    } else {
        IBEP20(tokenAddr).transfer(
            baseAccount,
            IBEP20(tokenAddr).balanceOf(address(this))
        );
    }
}
```

- The owner can change the minimum and maximum amounts of staking days and limit



```
ftrace | funcSig
function setPledgePeriodLimit(uint256 min, uint256 max) public {
    require(msg.sender == owner());

    minDays = min;
    maxDays = max;
}

ftrace | funcSig
function setPledgeAmountLimit(uint256 min, uint256 max) public {
    require(msg.sender == owner());

    minAmount = min;
    maxAmount = max;
}
```

- The safu dev can transfer the ownership



```
ftrace | funcSig
function transferSAFU(address safu) public {
    require(msg.sender == safuAddr);
    safuAddr = safu;
}
```

Audit conclusion

RugFreeCoins team has performed in-depth tests, line-by-line manual code review, and automated audit of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, manipulations, and hacks. According to the smart contract audit.

Smart contract functional Status: **PASS**

Number of risk issues: **1**

Solidity code functional issue level: **PASS**

Number of owner privileges: **10**

Centralization risk correlated to the active owner: **YES**

Smart contract active ownership: **ACTIVE**