Cyberscope

Audit Report

Nevis Investments

June 2022

SHA256 f5593fd4d7cb69734d13c226595f99e27e10947ddb79370cdb32bd29ee45c563

Audited by © cyberscope

Table of Contents

Contract Review

Contract Name	Nevis
Testing Deploy	https://testnet.bscscan.com/token/0x15b6D8F6a32A8D48AFE5c9572D7e92D61BdF34de
Symbol	nevis
Decimals	9
Total Supply	100,000,000,000
Domain	https://nevis.investments/

Source Files

Filename	**SHA256**
contract.sol	f5593fd4d7cb69734d13c226595f99e27e10947ddb79370cdb32bd29ee45c563

Audit Updates

Initial Audit	28th June 2022
Corrected	

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L1143,L1146

Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the `_NevisbankFee` to 100% and stop buying transactions.

```
if(recipient!=uniswapV2Pair){
        OwnerAmt = tAmount.mul(_NevisbankFee).div(100);
        if(OwnerAmt > 0){emit Transfer(sender, _NevisbankWallet, OwnerAmt);}
}else{
        OwnerAmt = tAmount.mul(_NevisswapSellFee).div(100);
        if(OwnerAmt > 0){emit Transfer(sender, _NevisbankWallet, OwnerAmt);}
}
```

The owner can also turn the contract into a honeypot and prevent users from selling by setting `_NevisswapSellFee` to 100%.

```
if(recipient!=uniswapV2Pair){
        OwnerAmt = tAmount.mul(_NevisbankFee).div(100);
        if(OwnerAmt > 0){emit Transfer(sender, _NevisbankWallet, OwnerAmt);}
}else{
        OwnerAmt = tAmount.mul(_NevisswapSellFee).div(100);
        if(OwnerAmt > 0){emit Transfer(sender, _NevisbankWallet, OwnerAmt);}
}
```

Recommendation

The contract could embody a check for not allowing setting the _NevisbankFee and _NevisswapSellFee less than a reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L1220,L1224,L1228

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setTaxFeePercent`, `setNevisbankFeePercent`, `setNeviswapFeePercent` function with a high percentage value.

```
function setTaxFeePercent(uint256 redistributionFee) external onlyOwner() {
        _redistributionFee = redistributionFee;
}

function setNevisbankFeePercent(uint256 NevisbankFee) external onlyOwner() {
        _NevisbankFee = NevisbankFee;
}

function setNeviswapFeePercent(uint256 NevisswapFee) external onlyOwner() {
        _NevisswapSellFee = NevisswapFee;
}
```

Recommendation

The contract could embody a check for the maximum acceptable value.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	FSA	Fixed Swap Address
	CO	Code Optimization
	MC	Missing Check
	L01	Public Function could be Declared External
	L02	State Variables could be Declared Constant
	L04	Conformance to Solidity Naming Conventions
	L05	Unused State Variable
	L07	Missing Events Arithmetic
	L09	Dead Code Elimination
	L11	Unnecessary Boolean equality
	L13	Divide before Multiply Operation

FSA - Fixed Swap Address

Criticality	minor
Location	contract.sol#L844

Description

The swap address is assigned once in the constructor and it can not be changed. The decentralized swaps sometimes create a new swap version or abandon the current. A contract that cannot change the swap address may not be able to catch-up the upgrade.

```
IUniswapV2Router02 _uniswapV2Router =
IUniswapV2Router02(0x9Ac64Cc6e4415144C455BD8E4837Fea55603e5c3);
// Create a uniswap pair for this new token
uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(busd,address(this));
```

Recommendation

It could be better to allow the swap address mutation in case of future swap updates.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L1123

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.
The variable `totalHoldings` is updated inside the function `_tokenTransfer` which increases the gas price of each transaction. Moving calculation of `totalHoldings` to `distributerewards` could optimize this code segment.

```
totalHoldings=0;
for(uint256 i = 0; i < investors.length; i++){
      if (!_isExcluded[investors[i]]) {
            totalHoldings+=balanceOf(investors[i]);
      }
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

MC - Missing Check

Criticality	minor
Location	contract.sol#L1137, L1123

Description

The contract is processing variables that have not properly sanitized and checked that they form the proper shape. The SafeMath library must be used to avoid possible variable overflow. Due to the old solidity version.

```
user.rewards+=userreward;

totalHoldings+=balanceOf(investors[i]);
```

Recommendation

The contract should properly check the variables according to the required specifications

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L933,912,1212,121,532,1206,126,930,99,137,1061,95,922,940,132,946,926,541,879,917,547,91,112,875,103,560,1001,871,117,552,1202

Description

Public functions that are never called by the contract should be declared external to save gas.

```
excludeFromFee
lock
allowance
name
_getCurrentSupply
unlock
totalSupply
symbol
transfer
...
```

Recommendation

Use the external attribute for functions never called from the contract.

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L817,88,781,777,815,87,85,810,89,780,77,782,829,813,816

Description

Constant state variables should be declared constant to save gas.

```
_symbol
totalWithdraw
_NevisRewardWallet
DEAD_NON_CHECKSUM
busd
DEAD
_decimals
_tTotal
_limitSupply
...
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L1025,782,826,1224,85,88,600,616,781,780,1012,835,599,834,87,1001,933,79,1228,1086,638,822,829,823,789,784,89,819

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_redistributionFee
_decimals
_tOwned
_balances
_NevisbankWallet
_NevisRewardWallet
_NevisbankFee
WETH
_Amount
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L782,85,78,79,813,827,77,811

Description

There are segments that contain unused state variables.

```
_rTotal
busd
_previousNevisswapsellFee
totalWithdraw
nevos_token
token
_limitSupply
DEAD_NON_CHECKSUM
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1224,1228,1220

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
_redistributionFee = redistributionFee
_NevisswapSellFee = NevisswapFee
_NevisbankFee = NevisbankFee
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L1174,389,440,153,450,1086,455,1056,362,425,1048,415

Description

Functions that are not used in the contract, and make the code's size bigger.

```
functionCall
removeAllFee
isContract
restoreAllFee
_functionCallWithValue
swaptoken
functionCallWithValue
_burn
sendValue
...
```

Recommendation

Remove unused functions.

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L1106

Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
includeInInvestor(recipient) != true && ! _isExcludedFromFee[recipient]
```

Recommendation

Remove the equality to the boolean constant.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L1025

Description

Performing divisions before multiplications may cause lose of prediction.

```
dividentShare = nevisinbusd.mul(100).div(holdingbusd)
```

Recommendation

The multiplications should be prior to the divisions.

Contract Functions

Contract	Type	Bases		
Contract	**Type**	**Bases**		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
ERC20	Implementation	IERC20		
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
SafeMath	Library			
	add	Internal		
	sub	Internal		

	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	_functionCallWithValue	Private	✓	
Ownable	Implementation	Context		
	<Constructor>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	geUnlockTime	Public		-
	lock	Public	✓	onlyOwner
	unlock	Public	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-

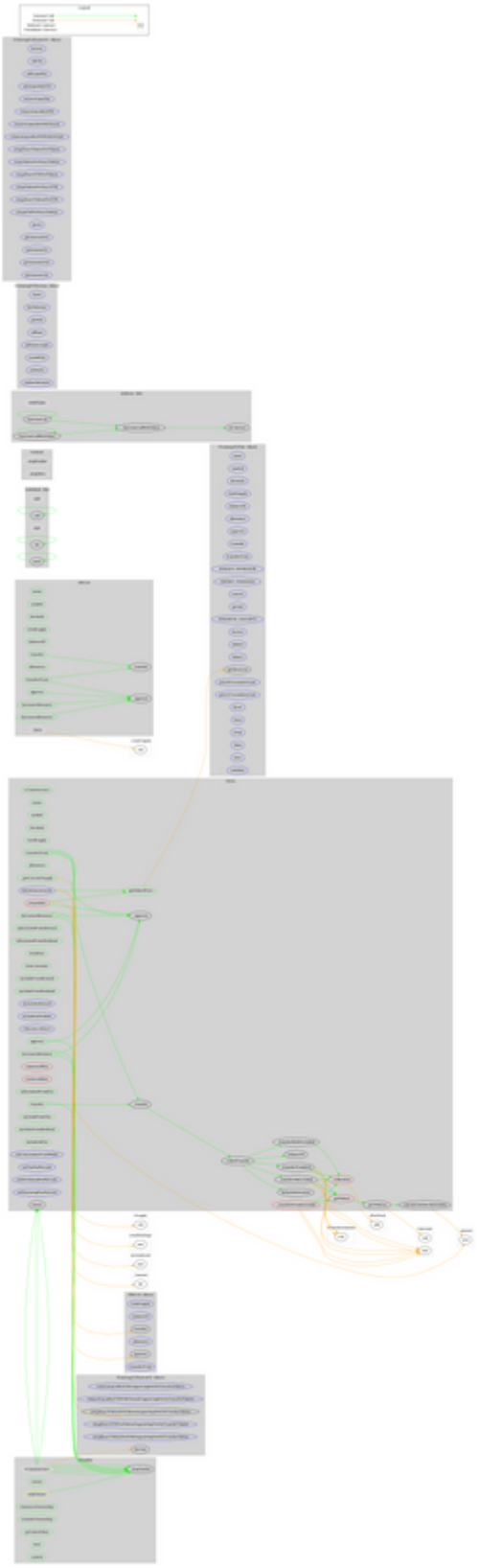
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-

IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
Nevis	Implementation	Context,		

		IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	isExcludedFromReward	Public		-
	isExcludedFromDividend	Public		-
	totalFees	Public		-
	total_investor	Public		-
	excludeFromReward	Public	✓	onlyOwner
	excludeFromDividend	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	includeInDivident	External	✓	onlyOwner
	<Receive Ether>	External	Payable	-
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getCurrentSupply	Public		-
	calculateredistributionFee	Private		
	getTokenPrice	Public		-
	distributerewards	External	Payable	onlyOwner
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	
	isExcludedFromFee	Public		-
	_approve	Private	✓	
	_transfer	Private	✓	

	swaptoken	Private	✓	
	_tokenTransfer	Private	✓	
	includeInInvestor	Private		
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	
	_transferBothExcluded	Private	✓	
	excludeFromFee	Public	✓	onlyOwner
	excludeFromdividend	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	setCommunityTrustWallet	External	✓	onlyOwner
	setTaxFeePercent	External	✓	onlyOwner
	setNevisbankFeePercent	External	✓	onlyOwner
	setNeviswapFeePercent	External	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	nevis.investments
Registry Domain ID	96638f912b3b43dea5c1a1d6d10aee9e-DONUTS
Creation Date	2022-02-02T10:11:43Z
Updated Date	2022-05-05T11:44:18Z
Registry Expiry Date	2027-02-02T10:11:43Z
Registrar WHOIS Server	www.bigrock.com.in/whois-lookup.php
Registrar URL	http://bigrock.com
Registrar	BigRock Solutions Ltd.
Registrar IANA ID	1495

The domain has been created 5 months before the creation of the audit. It will expire in over 4 years.

There is no public billing information, the creator is protected by the privacy settings.

Summary

The Smart Contract analysis reported no compiler error. The contract Owner can access some admin functions that can be used in a malicious way to disturb the users' transactions.

There are some functions that can be abused by the owner like stopping transactions and manipulating fees. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions.

A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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.

Cyberscope 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 Cyberscope 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 Cyberscope 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 Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io