



Cyberscope

Audit Report

Nevis Investments

July 2022

Type BEP20

Network BSC

Address 0x24DEF7C64B8a8C3Af2aE0a74A69999733502AB98

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Contract Analysis	4
Contract Diagnostics	5
MC - Missing Check	6
Description	6
Recommendation	6
L01 - Public Function could be Declared External	7
Description	7
Recommendation	7
L02 - State Variables could be Declared Constant	8
Description	8
Recommendation	8
L04 - Conformance to Solidity Naming Conventions	9
Description	9
Recommendation	9
L05 - Unused State Variable	10
Description	10
Recommendation	10
L13 - Divide before Multiply Operation	11
Description	11
Recommendation	11
Contract Functions	12
Contract Flow	17

Domain Info	18
Summary	19
Disclaimer	20
About Cyberscope	21

Contract Review

Contract Name	Nevis
Compiler Version	v0.6.12+commit.27d51765
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0x24DEF7C64B8a8C3Af2aE0a74A69999733502AB98
Symbol	NEVIS
Decimals	9
Total Supply	100,000,000,000
Domain	https://nevis.investments/

Source Files

Filename	SHA256
contract.sol	3a36341d9bb0f50fc53472f1b1cfed3ff8a98e74766f3c612e738ae8529e279a

Audit Updates

Initial Audit	12th July 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

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	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
●	L13	Divide before Multiply Operation

MC - Missing Check

Criticality

minor

Location

contract.sol#L786,L800

Description

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

```
totalHoldings+=balanceOf(investors[i]);  
user.rewards+=userreward;
```

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#L687,637,651,301,684,956,806,676,655,680,310,633,671,316,629,750,625,660,646

Description

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

```
transfer
transferFrom
name
_getCurrentSupply
symbol
geUnlockTime
decreaseAllowance
decimals
transferOwnership
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality

minor

Location

contract.sol#L572,549,570,567,548,571

Description

Constant state variables should be declared constant to save gas.

```
_symbol  
DEAD  
_tTotal  
_name  
ZERO  
_decimals
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L547,936,581,369,385,549,548,761,590,368,589,750,687,942,591,407,577,584,578,556,551,574

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  
_tOwned  
_balances  
_NevisbankWallet  
_NevisRewardWallet  
_NevisbankFee  
WETH  
setfees  
NevisswapFee  
...
```

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#L566,579,582,575

Description

There are segments that contain unused state variables.

```
_previousredistributionFee  
_previousNeviswapSellFee  
_previousNevisbankFee  
MAX
```

Recommendation

Remove unused state variables.

L13 - Divide before Multiply Operation

Criticality

minor

Location

contract.sol#L775

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		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
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			
Ownable	Implementation	Context		
	<Constructor>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner

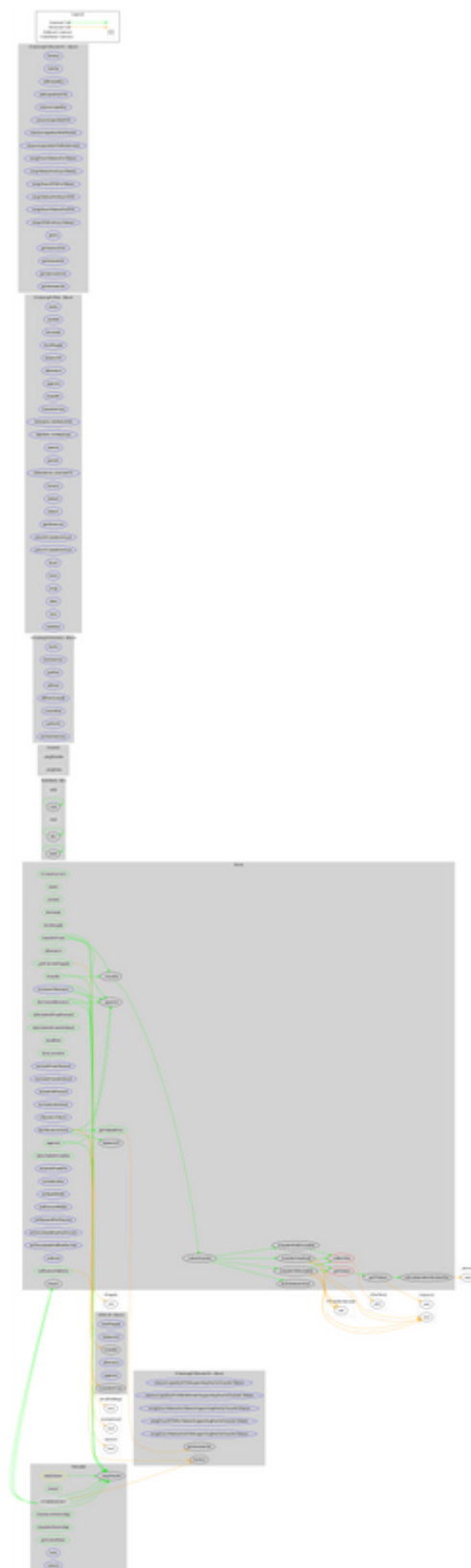
	geUnlockTime	Public		-
	lock	External	✓	onlyOwner
	unlock	External	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	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	IUniswapV2 Router01		
	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	External	✓	-
	decreaseAllowance	Public	✓	-
	isExcludedFromReward	Public		-
	isExcludedFromDividend	Public		-
	totalFees	Public		-
	total_investor	Public		-
	excludeFromReward	External	✓	onlyOwner
	excludeFromDividend	External	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	includeInDivident	External	✓	onlyOwner
	<Receive Ether>	External	Payable	-
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getCurrentSupply	Public		-
	calculatedredistributionFee	Private		
	getTokenPrice	Public		-

	distributerewards	External	Payable	onlyOwner
	isExcludedFromFee	Public		-
	_approve	Private	✓	
	_transfer	Private	✓	
	_tokenTransfer	Private	✓	
	includeInInvestor	Private		
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferBothExcluded	Private	✓	
	excludeFromFee	External	✓	onlyOwner
	includeInFee	External	✓	onlyOwner
	setBankWallet	External	✓	onlyOwner
	setRewardWallet	External	✓	onlyOwner
	setRewardFeePercent	External	✓	onlyOwner
	setNevisbankBuyFeePercent	External	✓	onlyOwner
	setNevisbankSellFeePercent	External	✓	onlyOwner
	setbusd	External	✓	onlyOwner
	setRouterAddress	Public	✓	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 in over 4 years before the creation of the audit.

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

Summary

Nevis Investments Token is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions. There is also a limit of max 10% on buy transactions and 25% on sell transactions

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>



Cyberscope

Audit Report

Nevis Investments

July 2022

Type BEP20

Network BSC

Address 0xCbe59C5Cb0F6C35918c320774bc11c46252F70a3

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Introduction	4
Source Files	5
Audit Updates	5
Contract Analysis	6
Contract Diagnostics	7
STC - Succeeded Transfer Check	8
Description	8
Recommendation	8
DSM - Data Structure Misuse	9
Description	9
Recommendation	9
L01 - Public Function could be Declared External	10
Description	10
Recommendation	10
L02 - State Variables could be Declared Constant	11
Description	11
Recommendation	11
L04 - Conformance to Solidity Naming Conventions	12
Description	12
Recommendation	12
L05 - Unused State Variable	13
Description	13

Recommendation	13
L09 - Dead Code Elimination	14
Description	14
Recommendation	14
L13 - Divide before Multiply Operation	15
Description	15
Recommendation	15
L14 - Uninitialized Variables in Local Scope	16
Description	16
Recommendation	16
Contract Functions	17
Contract Flow	22
Domain Info	23
Summary	24
Disclaimer	25
About Cyberscope	26

Contract Review

Contract Name	NEVISBANK
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xCbe59C5Cb0F6C35918c320774bc11c46252F70a3
Domain	https://nevis.investments

Source Files

Filename	SHA256
contract.sol	99b00f4c0e023ca16dc241979bdb7154a4fbcc19ffe1c7011b28f7af53cf72b4

Audit Updates

Initial Audit	13th July 2022
Corrected	

Introduction

The audited address `0xcbe59c5cb0f6c35918c320774bc11c46252f70a3` implements the contract of the Nevis Investments bank. The bank contract contains an investment platform where users can become referrers to other users to earn bonus, invest and withdraw dividends. The auditing of the bank contract is out of the scope of this audit. This audit is dedicated to the nevis bank implementation.

During the auditing period the contract:

<https://bscscan.com/address/0xcbe59c5cb0f6c35918c320774bc11c46252f70a3#contracts>

is pointed by the proxy address:

<https://bscscan.com/address/0xb37bf7f66c8187b9a23f386fed8c8bcf40f58ad#code>

Source Files

Filename	SHA256
contract.sol	3a36341d9bb0f50fc53472f1b1cfed3ff8a98e74766f3c6 12e738ae8529e279a

Audit Updates

Initial Audit	12th July 2022
Corrected	

Contract Analysis

The users have the ability to invest and withdraw tokens from the Nevis Trade Bank. In addition the users have the ability to become referrers to other users.

Nevis Trade Bank has one plan which locks your tokens for 365 days minimum and returns 0.6027% daily interest.

- On every investment on the contract the owner keeps a 10% fee on the contract. If a referrer is set, 1% bonus goes to the referrer's bonus. In addition, a minimum amount of investment is required to be greater or equal to $1 * 10^{18}$
- The users can withdraw tokens when the plan time is active.
- The users can reinvest all their earnings bonus and dividends if they have any.
- A referrer can withdraw his bonus if he has dividends in the Nevis Trade Bank.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	STC	Succeeded Transfer Check
●	DSM	Data Structure Misuse
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L05	Unused State Variable
●	L09	Dead Code Elimination
●	L13	Divide before Multiply Operation
●	L14	Uninitialized Variables in Local Scope

STC - Succeeded Transfer Check

Criticality

minor

Location

contract.sol#L705,L840

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
token.transfer(msg.sender,famount);

function rewards(address userAddress, uint256 amount) public onlyOwner() {
    require(getContractBUSDBalance() > amount,"Low Balance");
    token.transfer(userAddress,amount);
}
```

Recommendation

The contract should check if the result of the transfer methods is successful.

DSM - Data Structure Misuse

Criticality

minor

Location

contract.sol#L511

Description

The contract uses the valuable `_confirmedSnipers` as an array. The business logic of the contract does not require to iterate this structure sequentially. Thus, unnecessary loops are produced that increase the required gas.

Since the contract plan is always one, it could be a variable rather than an array.

```
Plan[] internal plans;
```

Recommendation

The contract could use a data structure that provides instant access. For instance, a Set or a Map would fit better to the business logic of the contract. This way the time complexity will be reduced from $O(n)$ to $O(1)$.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L116,208,840,683,807,91,846,552,127,87,98,102,189,122,811,865,217,223,833,736,83,823,95,111,859,850,795,803,107,814

Description

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

```
getUserAvailable  
allowance  
getContractTokenBalance  
getUserAmountOfRoll  
getinfo  
setPairaddress  
approve  
totalSupply  
getUserDepositInfo  
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality

minor

Location

contract.sol#L80,79,77,81,499

Description

Constant state variables should be declared constant to save gas.

```
totalRefBonus  
_decimals  
_limitSupply  
_name  
_symbol
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality

minor

Locationcontract.sol#L552,713,564,545,77,80,313,329,488,312,71,79,547,549,548,49,86
5,859,350,620,81

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.

```
_decimals  
buy_token  
Busd  
WETH  
pair_address  
Nevis_token  
router_address  
_____gap  
BUSD  
...
```

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#L530,49,77

Description

There are segments that contain unused state variables.

```
_limitSupply  
____gap  
referralAddress
```

Recommendation

Remove unused state variables.

L09 - Dead Code Elimination

Criticality

minor

Location

contract.sol#L143

Description

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

```
_burn
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality

minor

Location

contract.sol#L762

Description

Performing divisions before multiplications may cause lose of prediction.

```
share = user.deposits[i].amount.mul(plans[user.deposits[i].plan].percent).div(PANPER_DIVIDER)
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality

minor

Location

contract.sol#L765

Description

There are variables that are defined in the local scope and are not initialized.

```
totalAmount
```

Recommendation

All the local scoped variables should be initialized.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Initializable	Implementation			
	isConstructor	Private		
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	✓	
Context	Implementation			

	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context, Initializable		
	owninitialize	Public	✓	initializer
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	geUnlockTime	Public		-
	lock	External	✓	onlyOwner
	unlock	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
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	✓	-
NEVISBANK	Implementation	ERC20, Ownable		
	initialize	Public	✓	initializer
	invest	External	Payable	-
	buy_token	Private	✓	
	swapAndLiquify	Private	✓	
	addLiquidity	Private	✓	
	withdraw	Public	✓	-
	getTokenPrice	Public		-
	swaptoken	Private	✓	
	reinvest	Public	✓	-
	getUserDividends	Public		-
	includeInInvestor	Private		
	getUserReferralBonus	Public		-

	getUserAmountOfRoll	Public		-
	getContractBUSDBalance	Public		-
	getContractTokenBalance	Public		-
	getUserReferrer	Public		-
	getUserAmountOfDeposits	Public		-
	getUserAvailable	Public		-
	getUserTotalDeposits	Public		-
	getUserDepositInfo	Public		-
	rewards	Public	✓	onlyOwner
	getSiteInfo	Public		-
	getUserTotalWithdrawn	Public		-
	getUserInfo	Public		-
	getinfo	Public		-
	setPairaddress	Public	✓	onlyOwner
	setRouteraddress	Public	✓	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 in over 4 years before the creation of the audit.

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

Summary

Nevis Investment Token is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

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>



Cyberscope

Audit Report

Nevis Investments

June 2022

SHA256 f5593fd4d7cb69734d13c226595f99e27e10947ddb79370cdb32bd29ee45c563

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Contract Analysis	4
ST - Stop Transactions	5
Description	5
Recommendation	5
ELFM - Exceed Limit Fees Manipulation	7
Description	7
Recommendation	7
Contract Diagnostics	8
FSA - Fixed Swap Address	9
Description	9
Recommendation	9
CO - Code Optimization	10
Description	10
Recommendation	10
MC - Missing Check	11
Description	11
Recommendation	11
L01 - Public Function could be Declared External	12
Description	12
Recommendation	12
L02 - State Variables could be Declared Constant	13
Description	13

Recommendation	13
L04 - Conformance to Solidity Naming Conventions	14
Description	14
Recommendation	14
L05 - Unused State Variable	15
Description	15
Recommendation	15
L07 - Missing Events Arithmetic	16
Description	16
Recommendation	16
L09 - Dead Code Elimination	17
Description	17
Recommendation	17
L11 - Unnecessary Boolean equality	18
Description	18
Recommendation	18
L13 - Divide before Multiply Operation	19
Description	19
Recommendation	19
Contract Functions	20
Contract Flow	26
Domain Info	27
Summary	28
Disclaimer	29
About Cyberscope	30

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  
_previousNeviswapSellFee  
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		-
	calculatedredistributionFee	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>



Cyberscope

Audit Report

Nevis Investments

June 2022

SHA256 ee5c8791cf9e29a4955e20890a266bdba300105aebc0d4c16d294af03ff90b5a

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Contract Analysis	4
Contract Diagnostics	5
STC - Succeeded Transfer Check	6
Description	6
Recommendation	6
CO - Code Optimization	7
Description	7
Recommendation	7
DSM - Data Structure Misuse	8
Description	8
Recommendation	8
L01 - Public Function could be Declared External	9
Description	9
Recommendation	9
L02 - State Variables could be Declared Constant	10
Description	10
Recommendation	10
L04 - Conformance to Solidity Naming Conventions	11
Description	11
Recommendation	11
L05 - Unused State Variable	12
Description	12

Recommendation	12
L09 - Dead Code Elimination	13
Description	13
Recommendation	13
L13 - Divide before Multiply Operation	14
Description	14
Recommendation	14
L14 - Uninitialized Variables in Local Scope	15
Description	15
Recommendation	15
Contract Functions	16
Contract Flow	20
Domain Info	21
Summary	22
Disclaimer	23
About Cyberscope	24

Contract Review

Contract Name	NEVISTRADEBANK
Testing Deploy	https://testnet.bscscan.com/token/0x1a984838C102D54DaB36e3fcEF8f2AA28F3c5cBD
Domain	https://nevis.investments/

Source Files

Filename	SHA256
contract.sol	ee5c8791cf9e29a4955e20890a266bdba300105aebc0d4c16d294af03ff90b5a

Audit Updates

Initial Audit	28th June 2022
Corrected	

Contract Analysis

The users have the ability to invest and withdraw tokens from the Nevis Trade Bank. In addition the users have the ability to become referrers to other users. Nevis Trade Bank has one plan which locks your tokens for 365 days and returns 12% interest.

- On every investment on the contract the owner keeps a 10% fee on the contract. If a referrer is set, 1% bonus goes to the referrer's bonus.
- The users can withdraw tokens when the plan time is passed.
- A referrer can withdraw his bonus if he has dividends in the Nevis Trade Bank.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	STC	Succeeded Transfer Check
●	CO	Code Optimization
●	DSM	Data Structure Misuse
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L05	Unused State Variable
●	L09	Dead Code Elimination
●	L13	Divide before Multiply Operation
●	L14	Uninitialized Variables in Local Scope

STC - Succeeded Transfer Check

Criticality

minor

Location

contract.sol#L446,L579

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
token.transferFrom(address(msg.sender),address(this), _amount);
```

```
token.transfer(msg.sender,totalAmount);
```

Recommendation

The contract should check if the result of the transfer methods is successful.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L456

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.

Unnecessary variable check. This code segment can perform fewer operations.

```
if (user.referrer != address(0)) {  
    address upline = user.referrer;  
    if (upline != address(0)) {
```

The function `buy_token` swaps to the same address, so it can be called with the sum of `PROJECT_FEE` and `TOKEN_FEE` to avoid a function call.

```
uint256 fee = _amount.mul(PROJECT_FEE).div(PERCENTS_DIVIDER);  
buy_token(fee);  
uint256 purchase_amount= _amount.mul(TOKEN_FEE).div(PERCENTS_DIVIDER);  
buy_token(purchase_amount);
```

Recommendation

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

DSM - Data Structure Misuse

Criticality

minor

Location

contract.sol#L384

Description

The contract uses the valuable `_confirmedSnipers` as an array. The business logic of the contract does not require to iterate this structure sequentially. Thus, unnecessary loops are produced that increase the required gas. Since the contract plan is always one, it could be a variable rather than an array.

```
Plan[] internal plans;
```

Recommendation

The contract could use a data structure that provides instant access. For instance, a Set or a Map would fit better to the business logic of the contract. This way the time complexity will be reduced from $O(n)$ to $O(1)$.

L01 - Public Function could be Declared External

Criticality

minor

Location

contract.sol#L67,556,645,641,42,78,38,53,73,34,637,46,62,58

Description

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

```
allowance
approve
totalSupply
getContractBUSDBalance
name
increaseAllowance
transfer
symbol
decreaseAllowance
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality

minor

Location

contract.sol#L31,359,367,30,28,32,371,20,368,372,370

Description

Constant state variables should be declared constant to save gas.

```
NEVIS_TOKEN  
Pair  
totalRefBonus  
busd  
BUSD  
_decimals  
_limitSupply  
_name  
totalInvested  
...
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L425,372,28,31,181,197,370,359,440,180,22,30,495,356,218,371,594,32

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.

```
_decimals
_Amount
BUSD
WETH
NevisSwapRouter
buy_token
_name
nevis_token
DOMAIN_SEPARATOR
...
```

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#L28,410

Description

There are segments that contain unused state variables.

```
referralAddress  
_limitSupply
```

Recommendation

Remove unused state variables.

L09 - Dead Code Elimination

Criticality

minor

Location

contract.sol#L94

Description

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

```
_burn
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality

minor

Location

contract.sol#L613

Description

Performing divisions before multiplications may cause lose of prediction.

```
share = user.deposits[i].amount.mul(plans[user.deposits[i].plan].percent).div(PANPER_DIVIDER)
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality

minor

Location

contract.sol#L616

Description

There are variables that are defined in the local scope and are not initialized.

```
totalAmount
```

Recommendation

All the local scoped variables should be initialized.

Contract Functions

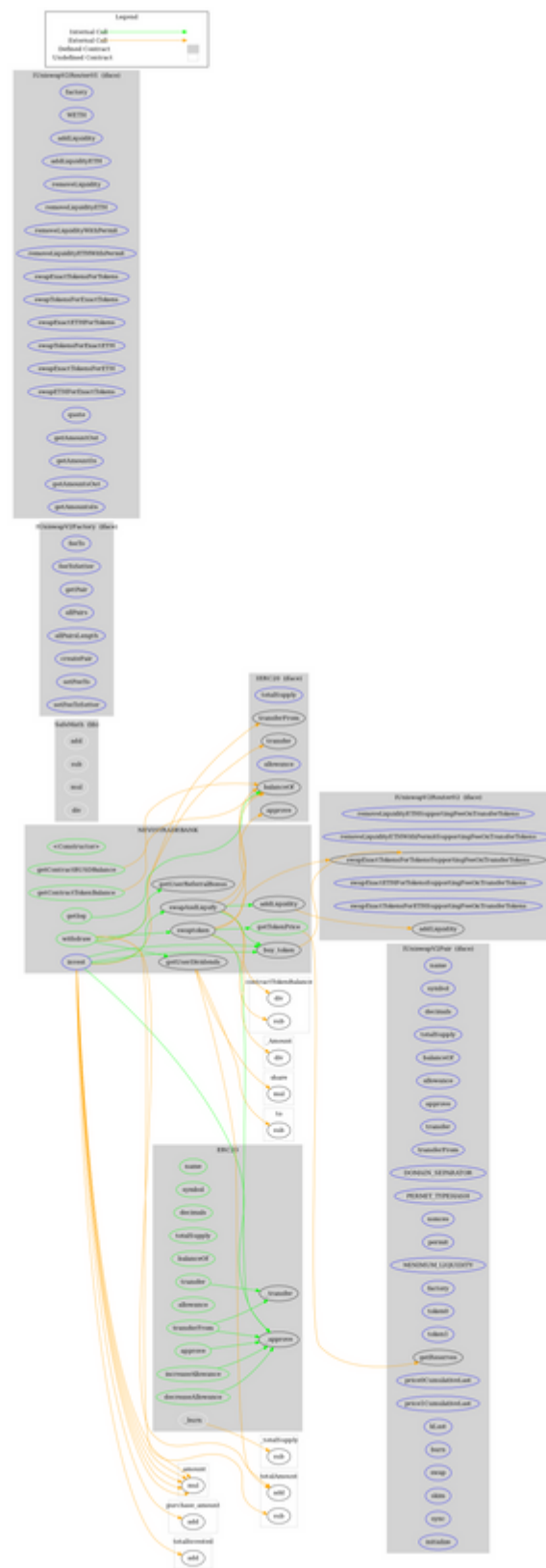
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
ERC20	Implementation	IERC20		
	name	Public		-
	symbol	Public		-
	decimals	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		
	mul	Internal		

	div	Internal		
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	✓	-

	swapExactETHForTokensSupporting FeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupporting FeeOnTransferTokens	External	✓	-
NEVISTRADEB ANK	Implementation	ERC20		
	<Constructor>	Public	✓	-
	invest	External	Payable	-
	buy_token	Private	✓	
	swapAndLiquify	Private	✓	
	addLiquidity	Private	✓	
	withdraw	Public	Payable	-
	getTokenPrice	Public		-
	swaptoken	Private	✓	
	getUserDividends	Public		-
	getUserReferralBonus	Public		-
	getContractBUSDBalance	Public		-
	getContractTokenBalance	Public		-
	getlog	Public		-

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

Nevis Investment Token is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

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>