

Smart Contract Security Audit V1

The Digital Nutmeg Token Smart Contract

28/1/2023



<https://saferico.com/>

business@saferico.com

https://t.me/SFI_ANN

—

Table of Contents

Table of Contents

Background

Project Information

Token Information

Executive Summary

File and Function Level Report

File in Scope:

Issues Checking Status

Severity Definitions

Audit Findings

Automatic testing

Testing proves

Inheritance graph

Call graph

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Project Information

- **Platform:** Binance Smart Chain
- **Contract Address:** 0x13ebdf10a406bd999f4af92b8b3775ebde52bc7a
- **Code Source:** <https://bscscan.com/token/0x13ebdf10a406bd999f4af92b8b3775ebde52bc7a#code>

Contracts address deployed to test net (Binance)

The Digital Nutmeg Token smart contracts on Binance test-net by the auditor to test every function .

<https://testnet.bscscan.com/address/0x49bc985095fd7eaf14b50c056a474ac127bb87d2>

Token Information:

Name	DNutmeg
Symbol	\$DNMG
Total supply	380,000,000
Decimals	4
Router	0x10ED43C718714eb63d5aA57B78B54704E256024E
Reward Token	0xe9e7CEA3DedcA5984780Bafc599bD69ADd087D56
Marketing Fee	3%
Liquidity Fee	1%
Reflection Fee	2%
Ecosystem Fee	5%
Burn fee	0%
Owner	0x6397fB3817108FBBb8Cd96580EF5D8C26118Cebf
Blacklist mode	true
Burn address	0x00dEaD
Distributor	0x1A8aEbAC445949BBE9cc6123F8c57D600Ae5ad21
PCS V2 Pair	0x75C9fAd157e9B14034C9Fa69837b22Ab816174b1
PCS V2 Router	0x10ED43C718714eb63d5aA57B78B54704E256024E

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Secured**.

Secured	✓
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 3 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

DNutmeg.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
DNutmeg.sol	7a08b8b3fa0125dc94431cabd664621585e5c48c3e82fb83e81413358fe72da7	0x13ebdf10a406bd999f4af92b8b3775ebde52bc7a

- Contract: DNutmeg
- Inherit: IBEP20, Auth
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
_maxTxAmount	✓	Read / public	Passed
_maxWalletToken	✓	Read / public	Passed
decimals	✓	Read / public	Passed
totalSupply	✓	Read / public	Passed
allowance	✓	Read / public	Passed
balanceOf	✓	Read / public	Passed
blacklistMode	✓	Read / public	Passed
BPDisabledForever	✓	Read / public	Passed
BP	✓	Read / public	Passed
bpEnabled	✓	Read / public	Passed
burnFeeReceiver	✓	Read / public	Passed
burnFee	✓	Read / public	Passed

buyCooldownEnabled	✓	Read / public	Passed
cooldownTimerInterval	✓	Read / public	Passed
distributor	✓	Read / public	Passed
ecosystemfee	✓	Read / public	Passed
feeDenominator	✓	Read / public	Passed
getCirculatingSupply	✓	Read / public	Passed
getLiquidityBacking	✓	Read / public	Passed
isAuthorized	✓	Read / public	Passed
getOwner	✓	Read / public	Passed
isOwner	✓	Read / public	Passed
isBlacklisted	✓	Read / public	Passed
isOverLiquified	✓	Read / public	Passed
liquidityFee	✓	Read / public	Passed
marketingFee	✓	Read / public	Passed
name	✓	Read / public	Passed
pair	✓	Read / public	Passed
router	✓	Read / public	Passed
reflectionFee	✓	Read / public	Passed
sellMultiplier	✓	Read / public	Passed
swapEnabled	✓	Read / public	Passed
swapThreshold	✓	Read / public	Passed
tradingOpen	✓	Read / public	Passed
approve	✓	Write / public	Passed
transferFrom	✓	Write / public	Passed
transfer	✓	Write / public	Passed
multiTransfer	✓	Write / public	Passed
approveMax	✓	Write / public	Passed
clearStuckBalance	✓	Write /public	Passed
clearStuckBalance_sender	✓	Write / public	Passed

cooldownEnabled	✓	Write / public	Passed
enable_blacklist	✓	Write / public	Passed
manage_blacklist	✓	Write / public	Passed
multiTransfer_fixed	✓	Write / public	Passed
set_sell_multiplier	✓	Write / public	Passed
setBotProtectionDisableForever	✓	Write / public	Passed
setBPAddress	✓	Write / public	Passed
setBpEnabled	✓	Write / public	Passed
setDistributionCriteria	✓	Write / public	Passed
setFeeReceivers	✓	Write / public	Passed
setFees	✓	Write / public	Passed
setIsDividendExempt	✓	Write / public	Passed
setIsFeeExempt	✓	Write / public	Passed
setIsTimelockExempt	✓	Write / public	Passed
setIsTxLimitExempt	✓	Write / public	Passed
setMaxTxPercent_base1000	✓	Write / public	Passed
setMaxWalletPercent_base1000	✓	Write / public	Passed
setSwapBackSettings	✓	Write / public	Passed
transferOwnership	✓	Write / public	Passed
setTargetLiquidity	✓	Write / public	Passed
setTxLimit	✓	Write / public	Passed
tradingStatus	✓	Write / public	Passed
unauthorize	✓	Write / public	Passed

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	Passed
2	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Design Logic.	Passed
6	Timestamp dependence.	Passed with notes
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed with notes
10	Methods execution permissions.	Passed
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Note	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Audit Findings

Critical:

No Critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found.

Medium:

No Medium severity vulnerabilities were found.

Low:

#Owner privileges (In the period when the owner isn't renounced)

Description

The owner can change the fees.

The owner can change the trade statue.

The owner can airdrop to any address.

The owner can add any address to blacklist.

```
function setFees(uint256 _liquidityFee, uint256 _reflectionFee, uint256
 _marketingFee, uint256 _ecosystemfee, uint256 _burnFee, uint256 _feeDenominator)
external authorized {
    liquidityFee = _liquidityFee;
    reflectionFee = _reflectionFee;
    marketingFee = _marketingFee;
    ecosystemfee = _ecosystemfee;
    burnFee = _burnFee;
    totalFee =
 _liquidityFee.add(_reflectionFee).add(_marketingFee).add(_ecosystemfee).add(_burnFee);
    feeDenominator = _feeDenominator;
    require(totalFee < feeDenominator/2, "Fees cannot be more than 50%");
}
function tradingStatus(bool _status) public onlyOwner {
    tradingOpen = _status;
}
```

Remediation

Make these functions internal in next version or the team should announce the investors before doing anything to give them time if they want to do anything.

P.S: This issue is common to the majority of reward smart contracts.

Status: **Acknowledged.**

#Pragma version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.17 instead of ^0.7.4). contracts should be deployed with the same compiler version and flags that they have been tested the most with. Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, the latest compiler which may have higher risks of undiscovered bugs. Contracts may also be deployed by others and the pragma indicates the compiler version intended by the original authors.

Remediation

Remove the ^ sign to lock the pragma version.

Comment:

The smart contract deployed since a year ago.

Status: **Acknowledged**

#Use of block.timestamp for comparisons

Description

The value of block.timestamp can be manipulated by the miner.
And conditions with strict equality is difficult to achieve -
block.timestamp

Remediation

Avoid use of block.timestamp

Status: **Acknowledged**

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Naming Conventions

Description

The contract follows a consistent naming convention where we are private variables with leading "_" and public variables without it. But we have missed to comply to the condition for certain variable names "_maxTxAmount" which is public.

Remediation

Remove "_" from external variable names and add it to private variable names.

Status: **Acknowledged**

Automatic Testing

1- Check for security

7a08b8b3fa0125dc94431cabd664621585e5c48c3e82fb83e81413358fe72da7

File: DNutm... | Language: solidity | Size: 28059 bytes | Date: 2023-01-26T12:03:54.617Z

Critical	High	Medium	Low	Note
0	0	0	0	0



2- SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

☒ Select all ☒ Autorun Run

Security

☒ Select Security

- ☒ Transaction origin:
'tx.origin' used
- ☒ Check-effects-interaction:
Potential reentrancy bugs
- ☒ Inline assembly:
Inline assembly used
- ☒ Block timestamp:
Can be influenced by miners
- ☒ Low level calls:
Should only be used by experienced devs
- ☒ Block hash:
Can be influenced by miners
- ☒ Selfdestruct:
Contracts using destructed contract can be broken

Gas & Economy

☒ Select Gas & Economy

- ☒ Gas costs:
Too high gas requirement of functions
- ☒ This on local calls:
Invocation of local functions via 'this'
- ☒ Delete dynamic array:
Use require/assert to ensure complete deletion
- ☒ For loop over dynamic array:
Iterations depend on dynamic array's size
- ☒ Ether transfer in loop:
Transferring Ether in a for/while/do-while loop

SOLIDITY STATIC ANALYSIS

ERC

☒ Select ERC

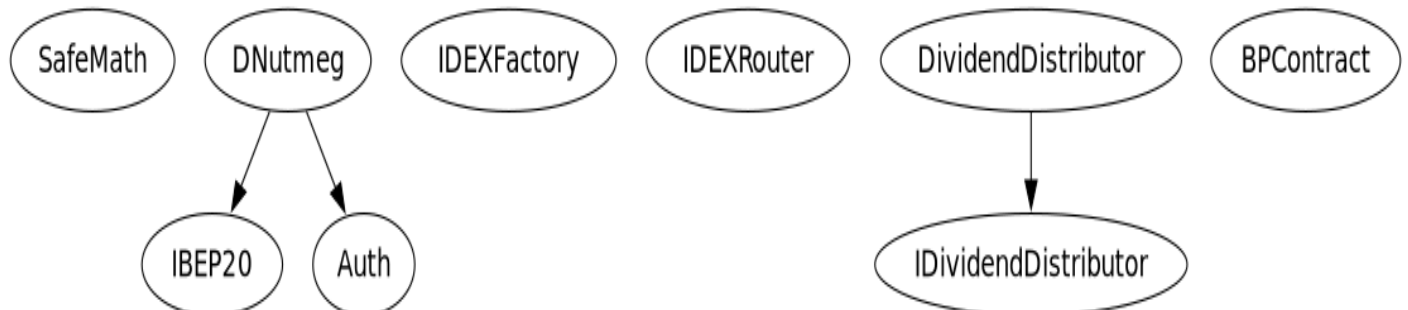
- ☒ ERC20:
'decimals' should be 'uint8'

Miscellaneous

☒ Select Miscellaneous

- ☒ Constant/View/Pure functions:
Potentially constant/view/pure functions
- ☒ Similar variable names:
Variable names are too similar
- ☒ No return:
Function with 'returns' not returning
- ☒ Guard conditions:
Ensure appropriate use of require/assert
- ☒ Result not used:
The result of an operation not used
- ☒ String length:
Bytes length != String length
- ☒ Delete from dynamic array:
'delete' leaves a gap in array
- ☒ Data truncated:
Division on int/uint values truncates the result

3- Inheritance graph



4- SOLIDITY UNIT TESTING

SOLIDITY UNIT TESTING

✓ >

Test your smart contract in Solidity.

Select directory to load and generate test files.

Test directory:

☒ Select all

☒ tests/DNutmeg_test.sol

Progress: 1 finished (of 1)

PASS testSuite

(tests/DNutmeg_test.sol)

✓ Before all

✓ Check success

✓ Check success2

✓ Check failure

✓ Check sender and value

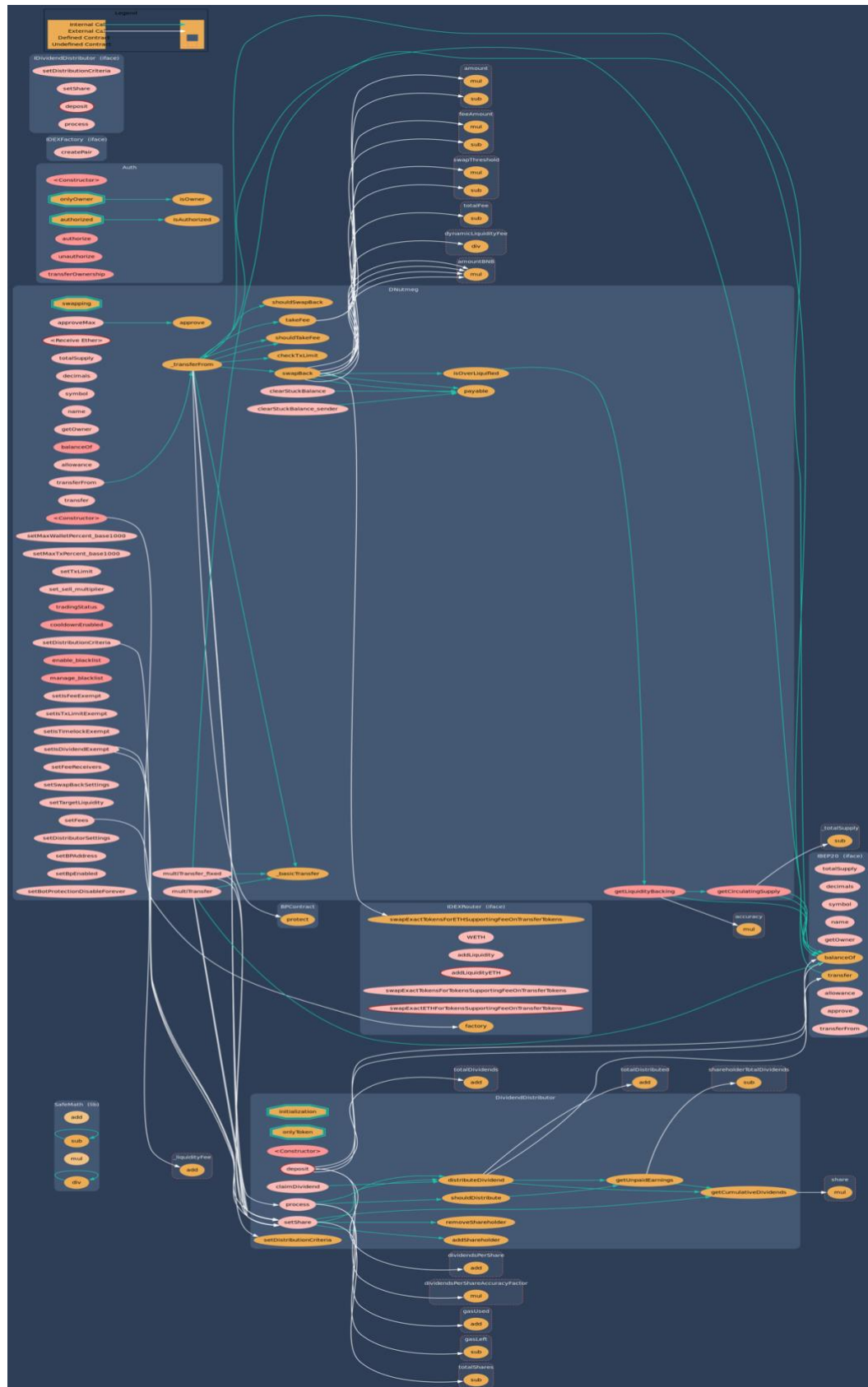
Result for tests/DNutmeg_test.sol

Passed: 5

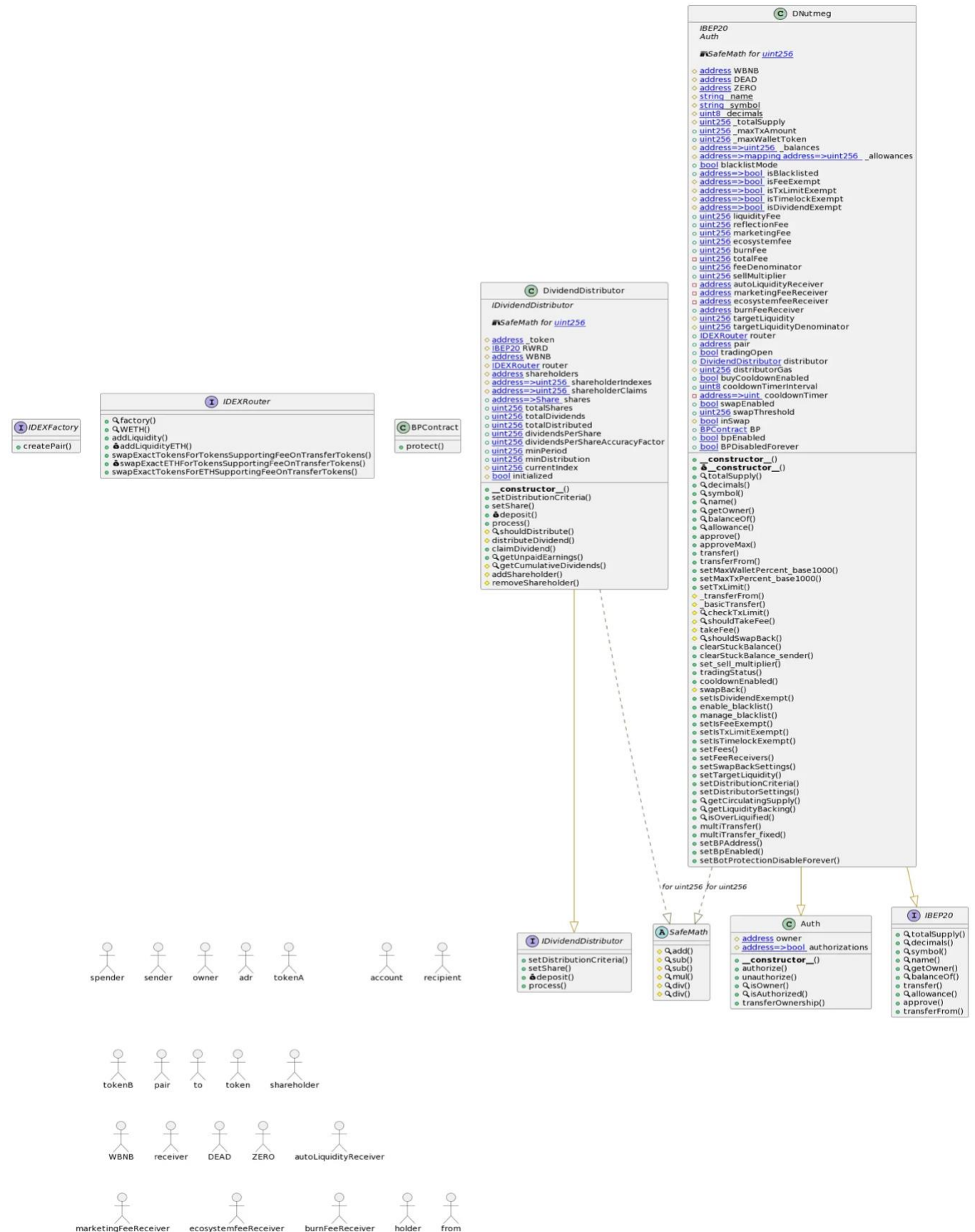
Failed: 0

Time Taken: 0.33s

5- Call graph



Unified Modeling Language (UML)



Functions signature

Sighash		Function Signature
=====		
771602f7	=>	add(uint256,uint256)
b67d77c5	=>	sub(uint256,uint256)
e31bdc0a	=>	sub(uint256,uint256,string)
c8a4ac9c	=>	mul(uint256,uint256)
a391c15b	=>	div(uint256,uint256)
b745d336	=>	div(uint256,uint256,string)
18160ddd	=>	totalSupply()
313ce567	=>	decimals()
95d89b41	=>	symbol()
06fdde03	=>	name()
893d20e8	=>	getOwner()
70a08231	=>	balanceOf(address)
a9059cbb	=>	transfer(address,uint256)
dd62ed3e	=>	allowance(address,address)
095ea7b3	=>	approve(address,uint256)
23b872dd	=>	transferFrom(address,address,uint256)
b6a5d7de	=>	authorize(address)
f0b37c04	=>	unauthorize(address)
2f54bf6e	=>	isOwner(address)
fe9fbb80	=>	isAuthorized(address)
f2fde38b	=>	transferOwnership(address)
c9c65396	=>	createPair(address,address)
c45a0155	=>	factory()
ad5c4648	=>	WETH()
e8e33700	=>	
addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256)		
f305d719	=>	addLiquidityETH(address,uint256,uint256,uint256,address,uint256)
5c11d795	=>	
swapExactTokensForTokensSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)		
b6f9de95	=>	
swapExactETHForTokensSupportingFeeOnTransferTokens(uint256,address[],address,uint256)		
791ac947	=>	
swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)		
2d48e896	=>	setDistributionCriteria(uint256,uint256)
14b6ca96	=>	setShare(address,uint256)
d0e30db0	=>	deposit()
ffb2c479	=>	process(uint256)
8c21cd52	=>	shouldDistribute(address)
5319504a	=>	distributeDividend(address)
f0fc6bca	=>	claimDividend()
28fd3198	=>	getUnpaidEarnings(address)
e68af3ac	=>	getCumulativeDividends(uint256)
db29fe12	=>	addShareholder(address)
9babdad6	=>	removeShareholder(address)
7e2f3afd	=>	protect(address,address,uint256)
571ac8b0	=>	approveMax(address)
09302dc6	=>	setMaxWalletPercent_base1000(uint256)
bd9ab537	=>	setMaxTxPercent_base1000(uint256)
5c85974f	=>	setTxLimit(uint256)
cb712535	=>	_transferFrom(address,address,uint256)
f0774e71	=>	_basicTransfer(address,address,uint256)

```
4afa518a => checkTxLimit(address,uint256)
e7c44c69 => shouldTakeFee(address)
a7dd4bbc => takeFee(address,uint256,bool)
0d5c6cea => shouldSwapBack()
1da1db5e => clearStuckBalance(uint256)
44a33fd2 => clearStuckBalance_sender(uint256)
ec72d65f => set_sell_multiplier(uint256)
0d295980 => tradingStatus(bool)
2d594567 => cooldownEnabled(bool,uint8)
6ac5eeee => swapBack()
f708a64f => setIsDividendExempt(address,bool)
5e562f3b => enable_blacklist(bool)
8e2eee84 => manage_blacklist(address[],bool)
658d4b7f => setIsFeeExempt(address,bool)
f84ba65d => setIsTxLimitExempt(address,bool)
50db71fb => setIsTimelockExempt(address,bool)
86f6c3c1 => setFees(uint256,uint256,uint256,uint256,uint256,uint256)
3c8e556d => setFeeReceivers(address,address,address,address)
df20fd49 => setSwapBackSettings(bool,uint256)
201e7991 => setTargetLiquidity(uint256,uint256)
9d1944f5 => setDistributorSettings(uint256)
2b112e49 => getCirculatingSupply()
d51ed1c8 => getLiquidityBacking(uint256)
1161ae39 => isOverLiquified(uint256,uint256)
1ca0a28d => multiTransfer(address,address[],uint256[])
335f6a43 => multiTransfer_fixed(address,address[],uint256)
d39b7e4f => setBPAddress(address)
206a8a22 => setBpEnabled(bool)
c818c280 => setBotProtectionDisableForever()
```

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/DNutmeg.sol	11aa54fd044613519788ff704d5734b69b09b759














Contracts Description Table

Contract	Type	Bases	
L	**Function Name**	**Visibility**	**Mutability**
Modifiers			
SafeMath	Library		
L add	Internal		
L sub	Internal		
L sub	Internal		
L mul	Internal		
L div	Internal		
L div	Internal		
IBEP20	Interface		
L totalSupply	External	!	NO!
L decimals	External	!	NO!
L symbol	External	!	NO!
L name	External	!	NO!
L getOwner	External	!	NO!
L balanceOf	External	!	NO!
L transfer	External		NO!
L allowance	External	!	NO!
L approve	External		NO!
L transferFrom	External		NO!
Auth	Implementation		
L <Constructor>	Public		NO!
L authorize	Public		onlyOwner
L unauthorize	Public		onlyOwner
L isOwner	Public	!	NO!
L isAuthorized	Public	!	NO!
L transferOwnership	Public		onlyOwner
IDEXFactory	Interface		
L createPair	External		NO!
IDEXRouter	Interface		
L factory	External	!	NO!
L WETH	External	!	NO!
L addLiquidity	External		NO!
L addLiquidityETH	External		NO!
L swapExactTokensForTokensSupportingFeeOnTransferTokens	External		NO!
L swapExactETHForTokensSupportingFeeOnTransferTokens	External		NO!



```

| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! |  | NO! | |
| | | |
| **IDividendDistributor** | Interface | | |
| L | setDistributionCriteria | External ! |  | NO! |
| L | setShare | External ! |  | NO! |
| L | deposit | External ! |  | NO! |
| L | process | External ! |  | NO! |
| | | |
| **DividendDistributor** | Implementation | IDividendDistributor | | |
| L | <Constructor> | Public ! |  | NO! |
| L | setDistributionCriteria | External ! |  | onlyToken |
| L | setShare | External ! |  | onlyToken |
| L | deposit | External ! |  | onlyToken |
| L | process | External ! |  | onlyToken |
| L | shouldDistribute | Internal  | | |
| L | distributeDividend | Internal  |  | | |
| L | claimDividend | External ! |  | NO! |
| L | getUnpaidEarnings | Public ! | | NO! |
| L | getCumulativeDividends | Internal  | | |
| L | addShareholder | Internal  |  | | |
| L | removeShareholder | Internal  |  | | |
| | | |
| **BPContract** | Implementation | | |
| L | protect | External ! |  | NO! |
| | | |
| **DNutmeg** | Implementation | IBEP20, Auth | | |
| L | <Constructor> | Public ! |  | Auth |
| L | <Receive Ether> | External ! |  | NO! |
| L | totalSupply | External ! | | NO! |
| L | decimals | External ! | | NO! |
| L | symbol | External ! | | NO! |
| L | name | External ! | | NO! |
| L | getOwner | External ! | | NO! |
| L | balanceOf | Public ! | | NO! |
| L | allowance | External ! | | NO! |
| L | approve | Public ! |  | NO! |
| L | approveMax | External ! |  | NO! |
| L | transfer | External ! |  | NO! |
| L | transferFrom | External ! |  | NO! |
| L | setMaxWalletPercent_base1000 | External ! |  | onlyOwner |
| L | setMaxTxPercent_base1000 | External ! |  | onlyOwner |
| L | setTxLimit | External ! |  | authorized |
| L | _transferFrom | Internal  |  | | |
| L | _basicTransfer | Internal  |  | | |
| L | checkTxLimit | Internal  | | |
| L | shouldTakeFee | Internal  | | |
| L | takeFee | Internal  |  | | |
| L | shouldSwapBack | Internal  | | |
| L | clearStuckBalance | External ! |  | authorized |
| L | clearStuckBalance_sender | External ! |  | authorized |
| L | set_sell_multiplier | External ! |  | onlyOwner |
| L | tradingStatus | Public ! |  | onlyOwner |
| L | cooldownEnabled | Public ! |  | onlyOwner |
| L | swapBack | Internal  |  | swapping |
| L | setIsDividendExempt | External ! |  | authorized |
| L | enable_blacklist | Public ! |  | onlyOwner |
| L | manage_blacklist | Public ! |  | onlyOwner |
| L | setIsFeeExempt | External ! |  | authorized |

```

L	setIsTxLimitExempt	External	!		authorized
L	setIsTimelockExempt	External	!		authorized
L	setFees	External	!		authorized
L	setFeeReceivers	External	!		authorized
L	setSwapBackSettings	External	!		authorized
L	setTargetLiquidity	External	!		authorized
L	setDistributionCriteria	External	!		authorized
L	setDistributorSettings	External	!		authorized
L	getCirculatingSupply	Public	!		NO
L	getLiquidityBacking	Public	!		NO
L	isOverLiquified	Public	!		NO
L	multiTransfer	External	!		onlyOwner
L	multiTransfer_fixed	External	!		onlyOwner
L	setBPAddress	External	!		onlyOwner
L	setBpEnabled	External	!		onlyOwner
L	setBotProtectionDisableForever	External	!		onlyOwner

Legend

Symbol	Meaning
	Function can modify state
	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “Secured”.

- ✓ No mint function.
- ✓ No volatile code.
- ✓ No high severity issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of 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 the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

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 Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico 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 Saferico 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, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, 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.