CFG NINJA AUDITS

Security Assessment

Nibble Token

February 12, 2023

Audit Status: Pass

Table of Contents

Assessment Summary

This report has been prepared for Nibble Token on the Binance Smart Chain network. CFGNINJA provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- Thorough line-by-line manual review of the entire codebase by industry experts.

Project Overview

Token Summary

Parameter	Result
Address	
Name	Nibble
Token Tracker	Nibble (Nibble)
Decimals	18
Supply	888,000
Platform	Binance Smart Chain
compiler	v0.8.17+commit.8df45f5f
Contract Name	NIBBLE
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	
Payment Tx	0x496947fd283af51cd9d017a1969aa5455b5208b8c12b0bb4b31fe9bfc80e9e9e

Project Overview

Risk Analysis Summary

Parameter	Result
Buy Tax	2.5%
Sale Tax	2.5%
Is honeypot?	Clean
Can edit tax?	Yes
Is anti whale?	Yes
Is blacklisted?	No
Is whitelisted?	No
Holders	0
Confidence Level	Low

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

Project Overview

Simulation Summary

Parameter	Result
Transfer From Owner	Pass
Transfer From Holder	Pass
Add Liquidity	Fail, see notes.
Buy from Owner	Pass
Buy from Holder	Pass
Remove Liquidity	Pass
SwapAndLiquify	Not-Present
RemoveLiquidity	Pass
LaunchPad	None

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

MainNet Contract was Not Assessed

TestNet Contract Assessed
Contract Name

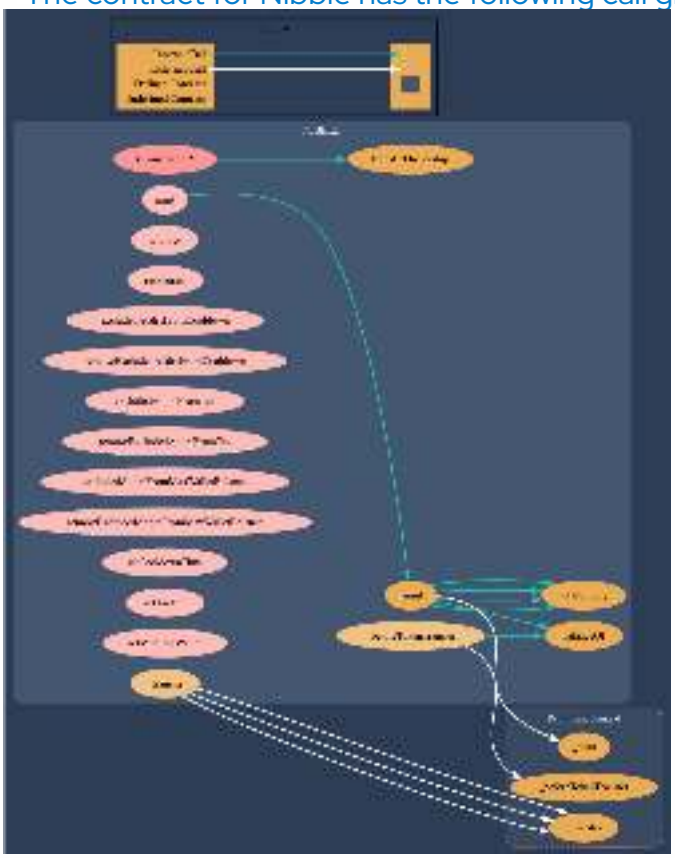
Name	Contract	Live
Nibble	0x5D4033aCcccB896a0c0E5797CBE88d0A878b1Ff3	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
Nibble	4e6b4e76acd25c733fe82bded79ca1fe607fcf8e	Nibble.sol

Call Graph

The contract for Nibble has the following call graph structure.



KYC Information

The Project Owners of Nibble have provided KYC Documentation.

KYC Certificated can be found on the Following:
KYC Data

KYC Information Notes:

Auditor Notes:

Project Owner Notes:

Smart Contract Vulnerability Checks

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) while overlaying a wide range of weakness variants that are specific to smart contracts.

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	Nibble.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	Nibble.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	Nibble.sol	L: 0 C: 0
SWC-103	Low	A floating pragma is set.	Nibble.sol	L: 2 C: 0
SWC-104	Pass	Unchecked Call Return Value.	Nibble.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	Nibble.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	Nibble.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	Nibble.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	Nibble.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	Nibble.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	Nibble.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-111	Pass	Use of Deprecated Solidity Functions.	Nibble.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	Nibble.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	Nibble.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	Nibble.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	Nibble.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	Nibble.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	Nibble.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	Nibble.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	Nibble.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randonmness.	Nibble.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	Nibble.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	Nibble.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	Nibble.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	Nibble.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	Nibble.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-126	Pass	Insufficient Gas Griefing.	Nibble.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	Nibble.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	Nibble.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	Nibble.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U +202E).	Nibble.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	Nibble.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	Nibble.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	Nibble.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	Nibble.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	Nibble.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	Nibble.sol	L: 0 C: 0

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.

Smart Contract Vulnerability Details

SWC-103 - Floating Pragma.

CWE-664: Improper Control of a Resource Through its Lifetime.

References:

Description:

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Remediation:

Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.

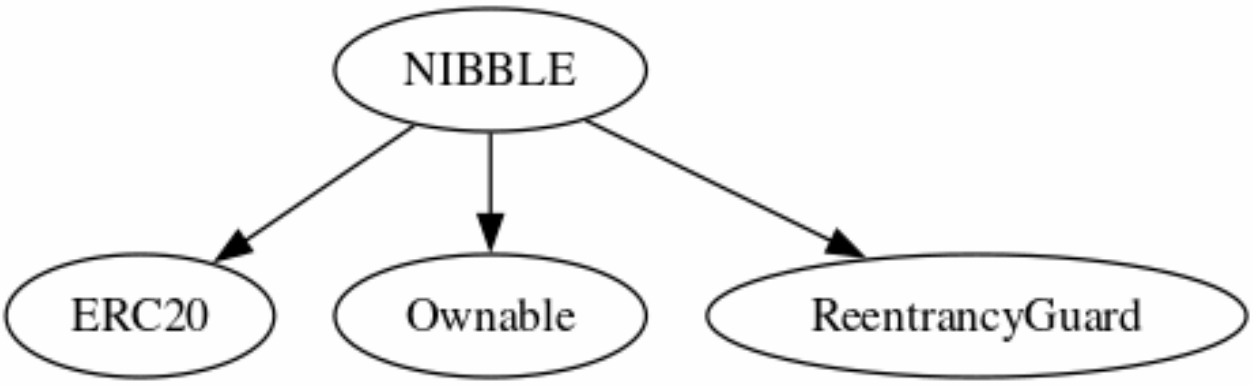
Pragma statements can be allowed to float when a contract is intended for consumption by other developers, as in the case with contracts in a library or EthPM package. Otherwise, the developer would need to manually update the pragma in order to compile locally.

References:

Ethereum Smart Contract Best Practices - Lock pragmas to specific compiler version.

Inheritance

The contract for Nibble has the following inheritance structure.



Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

Function Name	Parameters	Visibility
renounceOwnership		public
transferOwnership	newOwner (address)	public
setMultiSigWallet	address _multiSigWallet	external
setTaxFee	uint96 _taxFee	external
setCooldownTime	uint256 _cooldownTime	external
removeExcludedAddrsFromMaxWalletBalance	address[] calldata _addrs	external
excludedAddrsFromMaxWalletBalance	address[] calldata _addrs	external
removeExcludedAddrsFromTax	address[] calldata _addrs	external
excludedAddrsFromTax	address[] calldata _addrs	external

Function Name	Parameters	Visibility
removeExcludedAddrsFromCooldown	address[] calldata _addrs	external
excludedAddrsFromCooldown	address[] calldata _addrs	external
setPaused	bool _paused	external
setStage	Stages _stage	external

Smart Contract Advance Checks

ID	Severity	Name	Result	Status
Nibble-01	Minor	Potential Sandwich Attacks.	Pass	Not-Found
Nibble-02	Minor	Function Visibility Optimization	Pass	Not-Found
Nibble-03	Minor	Lack of Input Validation.	Fail	Pending
Nibble-04	Major	Centralized Risk In addLiquidity.	Pass	Pending
Nibble-05	Major	Missing Event Emission.	Fail	Pending
Nibble-06	Minor	Conformance with Solidity Naming Conventions.	Fail	Pending
Nibble-07	Minor	State Variables could be Declared Constant.	Pass	Not-Found
Nibble-08	Major	Dead Code Elimination.	Pass	Not-Found
Nibble-09	Major	Third Party Dependencies.	Pass	Not Found
Nibble-10	Major	Initial Token Distribution.	Pass	Not-Found
Nibble-11	Critical	The use of setHoldTime can lead to a pause trade or honeyPot State	Pass	Not-Found
Nibble-12	Major	Centralization Risks In The X Role	Pass	Not Found
Nibble-13	Informational	Extra Gas Cost For User..	Pass	Pending
Nibble-14	Medium	Unnecessary Use Of SafeMath	Pass	Not-Found

ID	Severity	Name	Result	Status
Nibble-15	Medium	Symbol Length Limitation due to Solidity Naming Standards.	Pass	Not-Found
Nibble-16	Medium	Invalid collection of Taxes during Transfer.	Pass	Not-Found

Nibble-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	Minor	Nibble.sol: 178,14	Pending

Description

The given input is missing the check for the non-zero address.

The given input is missing the check for the setMultiSigWallet, setTaxFee, setCooldownTime,removeExcludedAddrsFromMaxWalletBalance,excludedAddrsFromMaxWalletBalance,removeExcludedAddrsFromTax, excludedAddrsFromTax,removeExcludedAddrsFromCooldown,excludedAddrsFromCooldown,setPaused is missing required function.

Remediation

We advise the client to add the check for the passed-in values to prevent unexpected errors as below:

```
...
 require(receiver != address(0), "Receiver is the zero address");
...
...
require(value X limitation, "Your not able to do this function");
...
```

We also recommend customer to review the following function that is missing a required validation. setMultiSigWallet, setTaxFee, setCooldownTime,removeExcludedAddrsFromMaxWalletBalance,excludedAddrsFromMaxWalletBalance,removeExcludedAddrsFromTax, excludedAddrsFromTax,removeExcludedAddrsFromCooldown,excludedAddrsFromCooldown,setPaused is missing required function.

Nibble-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	Major	Nibble.sol: 178, 14	Pending

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.The linked code does not create an event for the transfer.

Remediation

Emit an event for critical parameter changes. It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

Nibble-06 | Conformance with Solidity Naming Conventions.

Category	Severity	Location	Status
Coding Style	Minor	Nibble.sol: 294,14	Pending

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.

excludedAddrsFromCooldown
removeExcludedAddrsFromCooldown
excludedAddrsFromTax
removeExcludedAddrsFromTax
excludedAddrsFromMaxWalletBalance

Remediation

Follow the Solidity naming convention.

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

Technical Findings Summary

Classification of Risk

Severity	Description
Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
Major	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
Minor	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

Findings

Severity	Found	Pending	Resolved
Critical	0	0	0
Major	1	0	0
Medium	0	0	0
Minor	2	0	0
Informational	0	0	0
Total	3	0	0

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/8bit_arcade1	Pass
Other	https://discord.gg/sgcA5dhh	Pass
Website	https://8bit-arcade.com/	Pass
Telegram	https://t.me/The8bitcryptocommunity	Pass

We recommend to have 3 or more social media sources including a completed working websites.

Social Media Information Notes:

Auditor Notes: undefined

Project Owner Notes:

Assessment Results

Score Results

Review	Score
Overall Score	99/100
Auditor Score	80/100
Review by Section	**Score**
Manual Scan Score	49/50
SWC Scan Score	36 /37
Advance Check Score	14 /16

The Following Score System Has been Added to this page to help understand the value of the audit, the maximun score is 100, however to attain that value the project most pass and provide all the data needed for the assessment. Our Passing Score has been changed to 80 Points, if a project does not attain 80% is an automatic failure. Read our notes and final assessment below.

Audit Passed

Assessment Results

Important Notes:

- The contract can be simplified in many ways, is currently using a few incorrect logics.
- Add Liquidity Failed since Pair is not created during deployment, pair needs to be created manually, then excluded from fees in order to add liquidity to the token.
- The transfer works as expected.
- Add/Remove Fees,Wallet,Tax using an array. This can be simplified by using an isExcluded logic and set to true or false to be coded in a more simplified method and to save coding space. Current naming for functions is not using solidity naming structure correctly.
- Mint is set in 3 stages, each stage mints 286,000 so this token max supply will be 888,000.
- A 40% tax on sales (initially). This tax will be reduced for every subsequent mint until Platform is LIVE; then, it will be removed after consulting with the community of holders and conducting a vote (proceeds from tax will be locked into a wallet which will be used for competitions,

Kichstarter funding, liquidity and anything else the holding community vote for).

Auditor Score =80
Audit Passed

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that actagainst the nature of decentralization, such as explicit ownership or specialized access roles incombination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on howblock.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that mayresult in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to makethe codebase more legible and, as a result, easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code,such as a constructor assignment imposing different require statements on the input variables than a setterfunction.

Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

Disclaimer

CFGNINJA has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocation for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and CFGNINJA is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will CFGNINJA or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.