

Assure DeFi™

The Verification **Gold Standard™**



Security Assessment **ROBIN HOOD Token**

October 13, 2023

Audit Status: Pass

Audit Edition: Standard



ASSURE DEFI™
THE VERIFICATION GOLD STANDARD

Risk Analysis

Classifications of Manual Risk Results

Classification	Description
● Critical	Danger or Potential Problems.
● High	Be Careful or Fail test.
● Low	Pass, Not-Detected or Safe Item.
● Informational	Function Detected

Manual Code Review Risk Results

Contract Privilege	Description
● Buy Tax	0%
● Sale Tax	0%
● Cannot Sale	Pass
● Cannot Sale	Pass
● Max Tax	0%
● Modify Tax	No
● Fee Check	Pass
● Is Honeypot?	Not Detected
● Trading Cooldown	Not Detected
● Can Pause Trade?	Pass
● Pause Transfer?	Not Detected.
● Max Tx?	Pass
● Is Anti Whale?	Not Detected
● Is Anti Bot?	Not Detected

Contract Privilege	Description
● Is Blacklist?	Not Detected
● Blacklist Check	Pass
● is Whitelist?	Not Detected
● Can Mint?	Pass
● Is Proxy?	Not Detected
● Can Take Ownership?	Not Detected
● Hidden Owner?	Not Detected
● Owner	no
● Self Destruct?	Not Detected
● External Call?	Not Detected
● Other?	Not Detected
● Holders	1
● Auditor Confidence	Low Risk

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

Token Summary

Parameter	Result
Address	0x7321A13bBF58220BCa94a605EFF3e24DAee86376
Name	ROBIN HOOD
Token Tracker	ROBIN HOOD (HOOD)
Decimals	18
Supply	10,000,000,000
Platform	Ethereum
compiler	v0.8.20+commit.a1b79de6
Contract Name	RobinHood
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://etherscan.io/token/0x7321A13bBF58220BCa94a605EFF3e24DAee86376#code
Payment Tx	Corporate

Main Contract Assessed Contract Name

Name	Contract	Live
ROBIN HOOD	0x7321A13bBF58220BCa94a605EFF3e24DAee86376	Yes

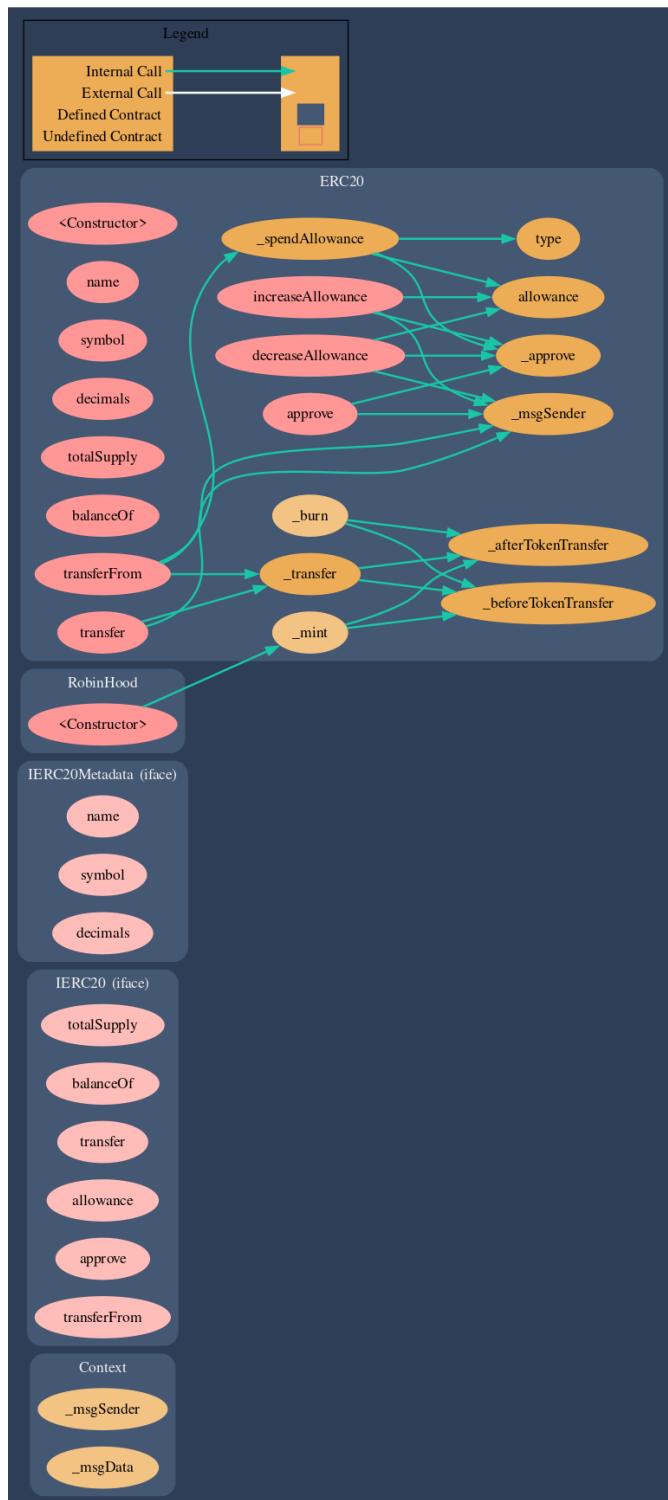
TestNet Contract was Not Assessed

Solidity Code Provided

SolidID	File Sha-1	FileName
Hood	ede5b383aab9b536ee360fd8fe6273034c30b2dc	robinhood.sol
Hood		
Hood		
Hood		

Call Graph

The contract for ROBIN HOOD has the following call graph structure.



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	robinhood.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	robinhood.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	robinhood.sol	L: 0 C: 0
SWC-103	Low	A floating pragma is set.	robinhood.sol	L: 43 C: 0
SWC-104	Pass	Unchecked Call Return Value.	robinhood.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	robinhood.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	robinhood.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	robinhood.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	robinhood.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	robinhood.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	robinhood.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	robinhood.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	robinhood.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	robinhood.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-114	Pass	Transaction Order Dependence.	robinhood.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	robinhood.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	robinhood.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	robinhood.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	robinhood.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	robinhood.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randomness.	robinhood.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	robinhood.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	robinhood.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	robinhood.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	robinhood.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	robinhood.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	robinhood.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	robinhood.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	robinhood.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	robinhood.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	robinhood.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	robinhood.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-132	Pass	Unexpected Ether balance.	robinhood.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	robinhood.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	robinhood.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	robinhood.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	robinhood.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.

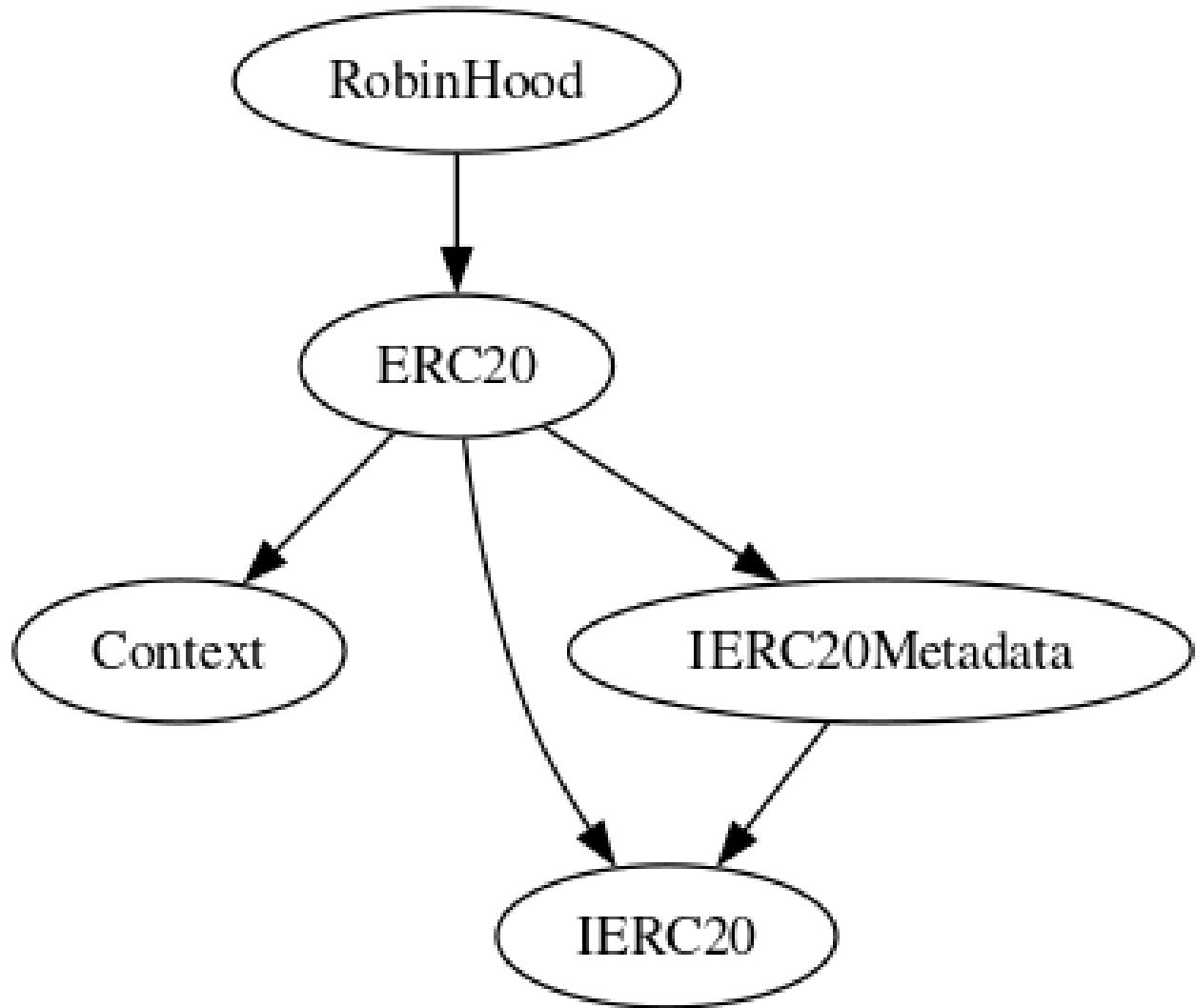
References:

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

Inheritance

The contract for ROBIN HOOD has the following inheritance structure.

The Project has a Total Supply of 10,000,000,000



Smart Contract Advance Checks

ID	Severity	Name	Result	Status
HOOD-01	Low	Potential Sandwich Attacks.	Pass	Not Detected
HOOD-02	Informational	Function Visibility Optimization	Pass	Not Detected
HOOD-03	Low	Lack of Input Validation.	Pass	Not Detected
HOOD-04	High	Centralized Risk In addLiquidity.	Pass	Not Detected
HOOD-05	Low	Missing Event Emission.	Pass	Not Detected
HOOD-06	Low	Conformance with Solidity Naming Conventions.	Pass	Not Detected
HOOD-07	Low	State Variables could be Declared Constant.	Pass	Not Detected
HOOD-08	Low	Dead Code Elimination.	Pass	Not Detected
HOOD-09	High	Third Party Dependencies.	Pass	Not Detected
HOOD-10	High	Initial Token Distribution.	Pass	Not Detected
HOOD-11	Medium	'difficulty' was replaced by 'prevrandaو'	Pass	Resolved
HOOD-12	High	Centralization Risks In The X Role	Pass	Not Detected
HOOD-13	Informational	Extra Gas Cost For User..	Pass	Not Detected
HOOD-14	Medium	Unnecessary Use Of SafeMath	Pass	Resolved
HOOD-15	Medium	Symbol Length Limitation due to Solidity Naming Standards.	Pass	Not Detected
HOOD-16	Medium	Taxes can be up to 100%	Pass	Not Detected
HOOD-17	Logical Issue	Conformance to numeric notation best practice.	Pass	Not Detected
HOOD-18	Critical	Stop Transactions by using Enable Trade.	Pass	Not Detected

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.
● High	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
◆ Low	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.
i 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
● High	0	0	0
● Medium	0	0	0
◆ Low	0	0	0
i Informational	0	0	0
Total	0	0	0

Social Media Checks

Social Media	URL	Result
Twitter	https://Twitter.com/hoodrobinEth	Pass
Other		Fail
Website	https://robinhoodcoin.vip	Pass
Telegram	https://T.me/robinhoodcoineth	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	100/100
Auditor Score	95/100
Review by Section	Score
Manual Scan Score	38/53
SWC Scan Score	35 /37
Advance Check Score	36 /40

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project must 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:

- This is a basic contract with zero tax.

**Auditor Score =95
Audit Passed**



Appendix

Finding Categories

Centralization / Privilege

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

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM 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 how `block.timestamp` works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being 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 may result in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the 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 setter function.

Coding Best Practices

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

Disclaimer

Assure Defi 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 advocacy 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 depreciation 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 Assure Defi is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will Assure Defi 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 Assure Defi 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.

