

Assure DeFi™

The Verification **Gold Standard**™



Security Assessment **Rebirth Token**

September 30, 2023

Audit Status: Fail

Audit Edition: Advance



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	Pass
● 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?	Fail
● Is Proxy?	Not Detected
● Can Take Ownership?	Not Detected
● Hidden Owner?	Not Detected
● Owner	0x1E427c35b3b54D5Ee14043b681A0E56595AEB0F6
● Self Destruct?	Not Detected
● External Call?	Not Detected
● Other?	Not Detected
● Holders	1
● Auditor Confidence	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.

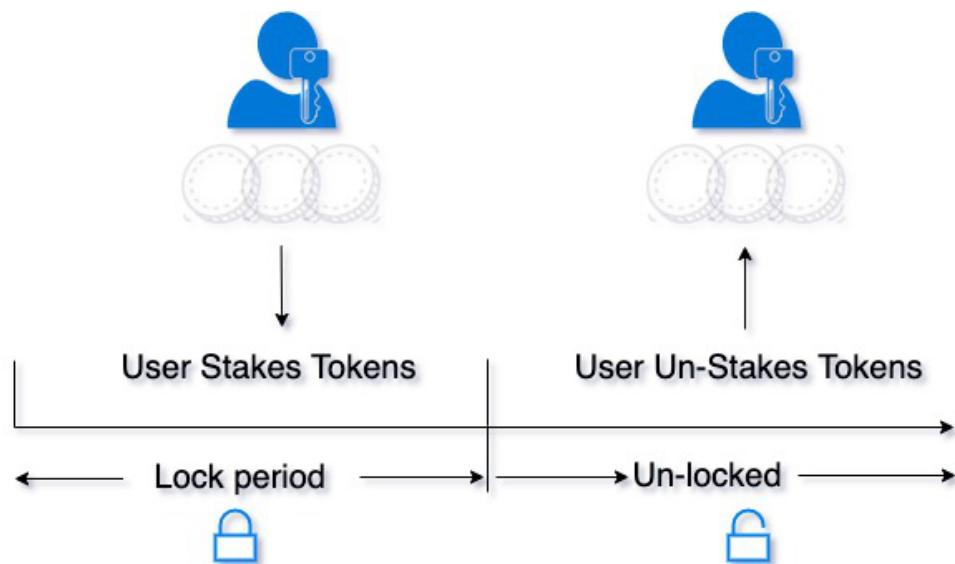
Name	Contract	Live
Rebirth	0xB1251C00A523194DFeC22580b61EEaCb820B9396	Yes
Project Overview		
Parameter	Result	Solidity Code Provided
Solidity Address	0x7a29fb472f0e62bce6127f125e8d5ad2a9fa3809	Rebirth.sol
Rebirth Name	c3f58141bd14b23bdbd26220a32dfc5765d86ae0	Rebirth.sol
Rebirth Token Tracker	Rebirth (RBH)	
Rebirth Decimals	18	
Rebirth Supply	1,000,000,000	
Platform	Ethereum	
compiler	v0.8.19+commit.7dd6d404	
Contract Name	rebirth	
Optimization	Yes with 200 runs	
License Type	Unlicensed	
Language	Solidity	
Codebase	https://etherscan.io/tx/0xb6ef0a2e2e90803762a9aa917ec93408cf74b48ddb75afaa67fe5c3b590dd91c	
Payment Tx	Corporate	

MainNet Contract was Not Assessed

TestNet Contract Assessed Contract Name

What is a Staking Contract

A smart contract which allows users to stake and un-stake a specified ERC20 token. Staked tokens are locked for a specific length of time (set by the contract owner at the outset). Once the time period has elapsed, the user can remove their tokens again.



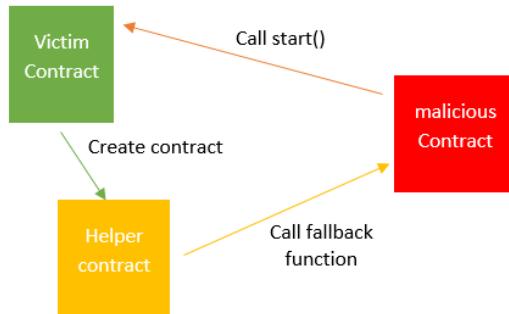
Reentrancy Check

The Project Owners of Rebirth have not configure the Reentrancy Guard library.

**You can read more about Reentrancy issues in the following link.
Reentrancy After Istanbul.**

We recommend the team to add the library to the contract to avoid potential issues.

We recommend the team to create a new contract with Reentrancy Guard added to the same.



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	Rebirth.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	Rebirth.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	Rebirth.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	Rebirth.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	Rebirth.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	Rebirth.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	Rebirth.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	Rebirth.sol	L: 0 C: 0
SWC-108	Low	State variable visibility is not set..	Rebirth.sol	L: 37 C: 22, L: 38 C: 23
SWC-109	Pass	Uninitialized Storage Pointer.	Rebirth.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	Rebirth.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	Rebirth.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	Rebirth.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	Rebirth.sol	L: 0 C: 0

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

ID	Severity	Name	File	location
SWC-132	Pass	Unexpected Ether balance.	Rebirth.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	Rebirth.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	Rebirth.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	Rebirth.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	Rebirth.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-108 - State Variable Default Visibility

CWE-710: Improper Adherence to Coding Standards

Description:

Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

Remediation:

Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

References:

Ethereum Smart Contract Best Practices - Explicitly mark visibility in functions and state variables

Smart Contract Vulnerability Details

SWC-120 - Weak Sources of Randomness from Chain Attributes

CWE-330: Use of Insufficiently Random Values

Description:

Solidity allows for ambiguous naming of state variables when inheritance is used. Contract A with a variable x could inherit contract B that also has a state variable x defined. This would result in two separate versions of x, one of them being accessed from contract A and the other one from contract B. In more complex contract systems this condition could go unnoticed and subsequently lead to security issues.

Shadowing state variables can also occur within a single contract when there are multiple definitions on the contract and function level.

Remediation:

Using commitment scheme, e.g. RANDAO. Using external sources of randomness via oracles, e.g. Oraclize. Note that this approach requires trusting in oracle, thus it may be reasonable to use multiple oracles. Using Bitcoin block hashes, as they are more expensive to mine.

References:

How can I securely generate a random number in my smart contract?)

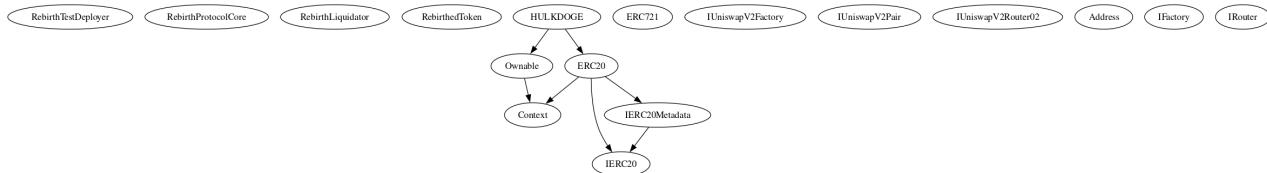
When can BLOCKHASH be safely used for a random number? When would it be unsafe?

The Run smart contract.

Inheritance

The contract for Rebirth has the following inheritance structure.

The Project has a Total Supply of 1,000,000,000



Privileged Functions (onlyOwner)

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

Function Name	Parameters	Visibility
CreatePool	address TokenAddress, address Pair Address, uint256	Public
ClosePool	uint256 PoolID, uint256 LengthInHOurs, uint256 SoftCap, string memory TokenName, string memory TokenSymbol	Public

Smart Contract Advance Checks

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

RBH-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	 Major	Rebirth.sol: L: 187 C: 24, L: 204 C: 23	 Pending

Description

The following functions are declared as public and are not invoked in any of the contracts contained within the projects scope:

Function Name	Parameters	Visibility
CreatePool	address TokenAddress, address PairAddress, uint256 HoursTillOpen, uint256 LengthInHours, uint256 SoftCap, string memory TokenName, string memory TokenSymbol	Public
ClosePool	uint256 PoolID	Public

The functions that are never called internally within the contract should have external visibility

Remediation

We advise that the function's visibility specifiers are set to external, and the array-based arguments change their data location from memory to calldata, optimizing the gas cost of the function.

References:

external vs public best practices.

RBH-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	 Minor	Rebirth.sol: L: 187 C: 24, L: 265 C: 15, L: 269 C: 15, L: 273 C: 14, L: 277 C: 14, L: 282 C: 14, L: 288 C: 14, L: 292 C: 14, L: 296 C: 14, L: 309 C: 14	 Pending

Description

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

The given input is missing the check for the .

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. .

RBH-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	 Medium	Rebirth.sol: L: 187 C: 24, L:204 C: 23, L: 265 C: 15, L: 269 C: 15, L: 273 C: 14, L: 277 C: 14, L: 282 C: 14, L: 288 C: 14, L: 292 C: 14,L: 296 C: 14, L: 301 C: 14, L: 309 C: 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.

RBH-11 | Contract use a custom Ownership library..

Category	Severity	Location	Status
Optimization	 Critical	Rebirth.sol: L: 89 C: 14	 Pending

Description

Contract use onlySuperAdmin and other cutom owners are not visible.

Remediation

We reccomend to use and RBAC style library or a simple ownership.

Project Action

RBH-12 | Centralization Risks In The Role

Category	Severity	Location	Status
Centralization / Privilege	● Major	Rebirth.sol: L: 509 C: 18, L: 574 C: 15	🕒 Pending

Description

In the contract , the role has authority over the following functions:
function burn(), to burn anyone's account at any amount.
function burnFrom(), to burn anyone's account at the number in the range of _allowed .
Any compromise to the account may allow the hacker to take advantage of this authority.
We understand the role could be assigned to the smart contract , however, the
is a map and more addresses could be added.

Remediation

The risk describes the current project design and potentially makes iterations to improve in the security operation
and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Project Action

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	1	0	0
● High	1	0	0
● Medium	0	0	0
◆ Low	2	0	0
i Informational	1	0	0
Total	5	0	0

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/RebirthEth	Pass
Other		Fail
Website	https://rebirthprotocol.io/	Pass
Telegram	https://t.me/rebirthprotoleth	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	48/100
Auditor Score	65/100
Review by Section	Score
Manual Scan Score	-10/53
SWC Scan Score	33 /37
Advance Check Score	25 /19

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 Fail



Assessment Results

Important Notes:

- Multiple issues/vulnerabilities were found.
- We recommend to use an RBAC style library and other custom owners are not visible.
- Incorrect method used for mint and burn, it does not follow erc standard libraries and this can lead to major exploits later on.

Auditor Score =65
Audit Fail



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.

