ASSURE DEFI®
THE VERIFICATION GOLD STANDARD

Security Assessment:

Regayov TOKEN

November 26, 2024

- Audit Status: **Pass**
- Audit Edition: **Advance**

Risk Analysis

Classifications of Manual Risk Results

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

Manual Code Review Risk Results

Contract Privilege	Description
Buy Tax	0%
Sale Tax	5%
Cannot Buy	Pass
Cannot Sale	Pass
Max Tax	5%
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	0x57272861395F1858eA5400fbB7A24b7Cebc211A0
Self Destruct?	Not Detected
External Call?	Not-Detected
Other?	Not Detected
Holders	4
Auditor Confidence	Medium
KYC Present	No
KYC URL	

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	0x85BE3628d1b86444ed6405FB53Cb95981380135e
Name	Regayov
Token Tracker	Regayov (HSACV)
Decimals	18
Supply	10,000,000,000
Platform	ETHEREUM
compiler	v0.8.6+commit.11564f7e
Contract Name	Regayov
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://sepolia.etherscan.io/address/0x85be3628d1b86444ed6405fb53cb95981380135e#code
Payment Tx	Corporate

Main Contract Assessed
Contract Name

Name	Contract	Live
Regayov	0x85BE3628d1b86444ed6405FB53Cb95981380135e	Yes

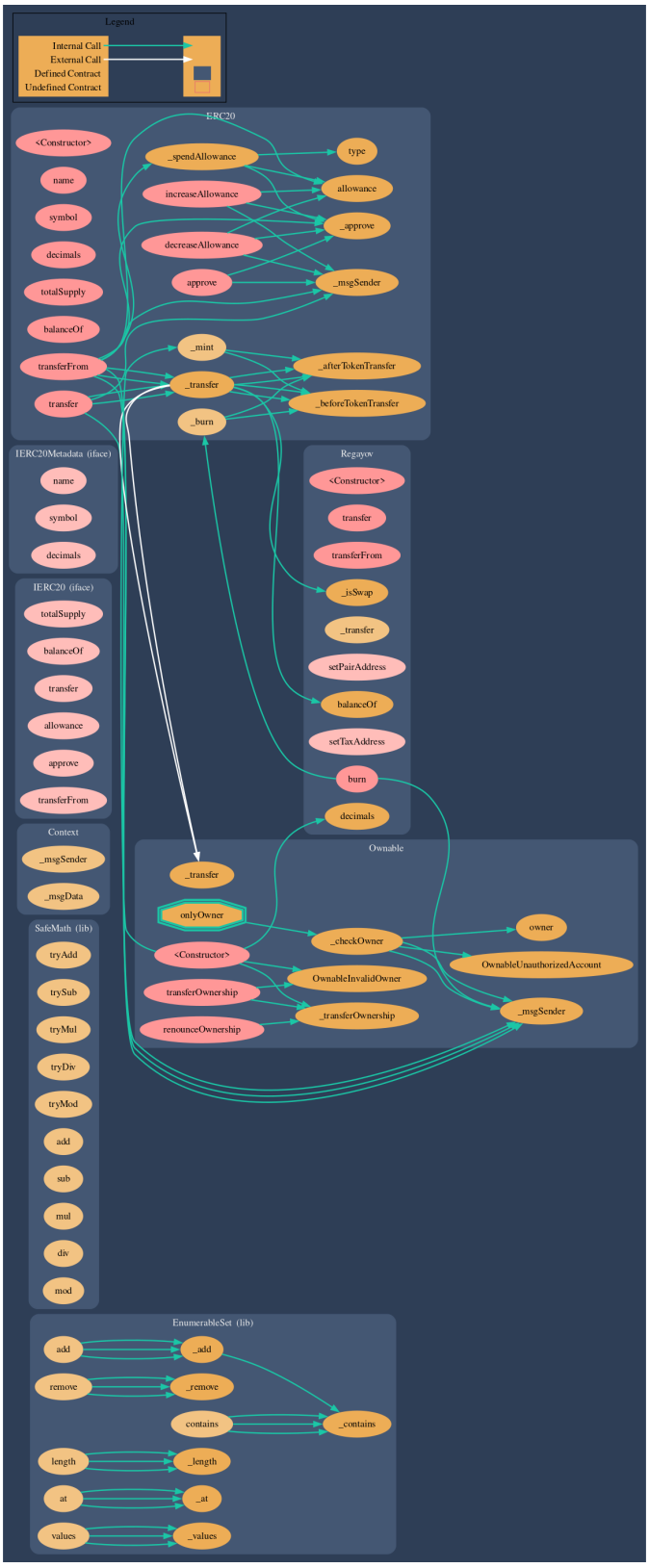
TestNet Contract was Not Assessed

Solidity Code Provided

SolID	File Sha-1	FileName
Regayov	047d40bfd648baa4ab7a4baf8758233b45742b61	Regayov.sol
Regayov		.sol
Regayov		.sol
Regayov		.sol
Regayov		.sol
Regayov		.sol

Call Graph

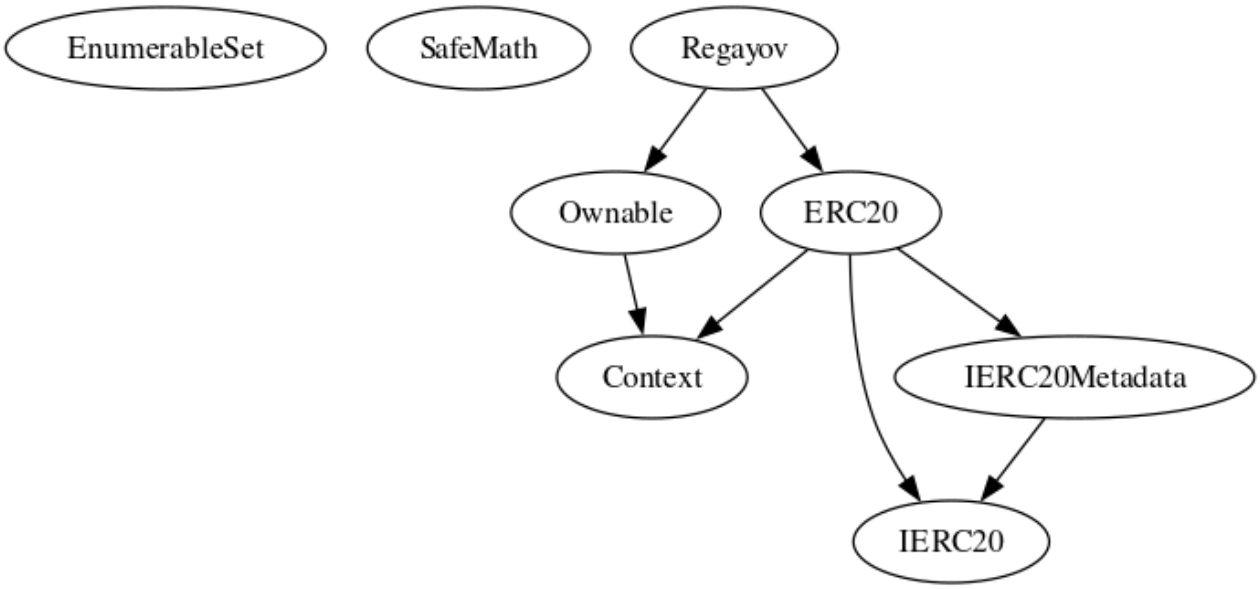
The contract for Regayov has the following call graph structure.



Inheritance

The contract for Regayov has the following inheritance structure.

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



HSACV-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	Low	Regayov.sol: L: 0 C: 12	Detected

Description

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

The given input is missing the check for the onlyOwners need to be revisited for require..

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. onlyOwners need to be revisited for require..

HSACV-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	Low	Regayov.sol: L: 506 C: 12, L: 593 C: 12	Detected

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.

HSACV-14 | Unnecessary Use Of SafeMath

Category	Severity	Location	Status
Logical Issue	Medium	Regayov.sol: L: 0 C: 0	Detected

Description

The SafeMath library is used unnecessarily. With Solidity compiler versions 0.8.0 or newer, arithmetic operations
will automatically revert in case of integer overflow or underflow.
library SafeMath {
An implementation of SafeMath library is found.
using SafeMath for uint256;
SafeMath library is used for uint256 type in contract.

Remediation

We advise removing the usage of SafeMath library and using the built-in arithmetic operations provided by the
Solidity programming language

Project Action

HSACV-19 | Centralization Privileges of.

Category	Severity	Location	Status
	Medium	Regayov.sol: L: 393 C: 14,L: 385 C: 14,L: 341 C: 14,L: 306 C: 14,L: 299 C: 14,L: 269 C: 14	Detected

Description

Centralized Privileges are found on the following functions.

Remediation

undefined

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.
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	2	2	0
Low	2	2	0
Informational	0	0	0
Total	4	4	0

Social Media Checks

Social Media	URL	Result
Twitter		Pass
Other		N/A
Website		Pass
Telegram		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	89/100
Auditor Score	89/100
Review by Section	**Score**
Manual Scan Score	32
Auto Scan Score	37
Advance Check Score	20

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 most pass and provide all the data needed for the assessment. Our Passing Score has been changed to 84 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below.

Audit Passed

Assessment Results

Important Notes:

- Ownership and Access Control: Ensure the owner is a trusted entity due to control over _pairAddress and _taxAddress. Consider implementing a multi-signature wallet for ownership to mitigate risks.ı
- Tax Mechanism: Tax is applied on transfers to _pairAddress. Ensure this address is correctly set and monitored. Verify that the _taxAddress is secure and funds are properly managed.ı
- Allowance Handling: transferFrom does not reset allowance to zero before setting a new one, which can lead to race conditions. Consider implementing the checks-effects-interactions pattern.ı
- Gas Optimization: Usage of EnumerableSet.values() can be costly if the set grows large. Be cautious of potential gas limit issues.ı
- Token Burning: Any user can burn their tokens. Ensure this is intended behavior and does not affect the tokenomics adversely.ı
- Reentrancy: Currently, there are no external calls that could lead to reentrancy. However, if future updates introduce such calls, consider adding reentrancy guards.ı
- Code Quality and Comments: The code is well-structured and uses OpenZeppelin libraries, which are industry standards. Ensure comments and documentation are up-to-date for maintainability.ı

- Upgradeability: The contract is not upgradeable. Ensure the current implementation is robust and consider future-proofing strategies.ı
- Testing and Verification: Conduct thorough testing, especially around the tax logic and ensure edge cases are covered, such as transfers involving _pairAddress and _taxAddress. Verify that the contract behaves as expected under various scenarios, including high transaction volumes.ı
- Event Emission: Ensure all critical state changes are accompanied by appropriate event emissions for transparency and traceability.ı
- Deployment and Monitoring: Monitor the contract post-deployment for any unusual activities or patterns. Implement alerts for significant changes in token balances, especially for _taxAddress.

Auditor Score =89
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

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 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 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.