AegisX

Smart Contract Audits | KYC

PALLADIUM

Security Assessment

TresChain Token

January 19, 2023

Table of Contents

Assessment Summary

This report has been prepared for TresChain Token on the Ethereum network. AegisX 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.

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	0	0	0
Medium	0	0	0
Minor	2	2	0
Informational	3	3	0
Total	5	5	0

Project Overview

Contract Summary

Parameter	Result
Address	0x66bc84b4270cA0F056E27e5cD77B2401522191c6
Name	TresChain
Token Tracker	TresChain (TRES)
Decimals	18
Supply	25,000,000
Platform	Ethereum
compiler	v0.8.17+commit.8df45f5f
Contract Name	TresChain
Optimization	200
LicenseType	MIT
Language	Solidity
Codebase	https://etherscan.io/token/0x66bc84b4270cA0F056E27e5cD77B2401522191c6#code
Payment Tx	

Project Overview

Risk Analysis Summary

Parameter	Result
Buy Tax	0%
Sale Tax	0%
Is honeypot?	Clean
Can edit tax?	No
Is anti whale?	No
Is blacklisted?	No
Is whitelisted?	Yes(Cooldown)
Holders	N/A
Security Score	94
Auditor Score	89
Confidence Level	High

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.

Main Contract Assessed Contract Name

Name	Contract	Live
TresChain	0x66bc84b4270cA0F056E27e5cD77B2401522191c6	Yes

TestNet Contract Assessed Contract Name

Name	Contract	Live
TresChain	0x72a9583E2180f5e6eF1fDF22F9a64F89a467a403	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
TresChain	b1e294d196a7256e0caef85e97b5ef5c7902a497	TresChain.sol

Mint Check

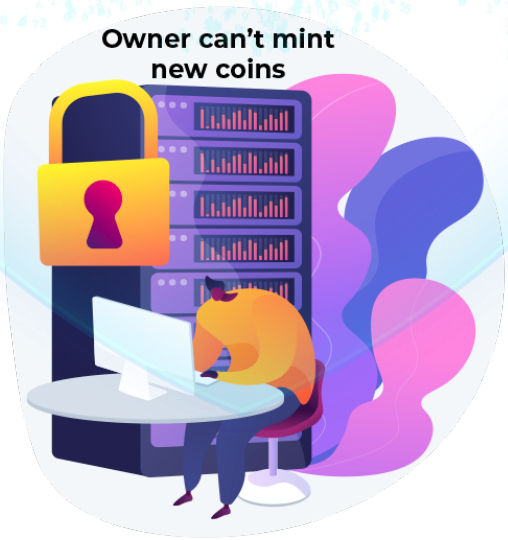
The project owners of TresChain do not have a mint function in the contract, owner cannot mint tokens after initial deploy.

The Project has a Total Supply of 25,000,000 and cannot mint any more than the Max Supply.

Mint Notes:

Auditor Notes: A mint Function does not exist.

Project Owner Notes:



Fees Check

The project owners of TresChain do not have the ability to set fees higher than 25%.

The team may have fees defined; however, they can't set those fees higher than 25% or may not be able to configure the same.

Tax Fee Notes:

Auditor Notes: The contract does not have a tax.

Project Owner Notes:

Blacklist Check

The project owners of TresChain do not have a blacklist function on their contract.

The project allow owners to transfer their tokens without any restrictions.

Token owner cannot blacklist the contract: Malicious or compromised owners can trap contracts relying on tokens with a blacklist.

Blacklist Notes:

Auditor Notes: The contract does not have a blacklist function.

Project Owner Notes:

MaxTx Check

The Project Owners of TresChain cannot set max tx amount

The Team allows any investors to swap, transfer or sell their total amount if needed.

MaxTX Notes:

Auditor Notes: The contract does not have a max tx function.

Project Owner Notes:

Pause Trade Check

The Project Owners of TresChain don't have the ability to stop or pause trading.

The Team has done a great job to avoid stop trading, and investors has the ability to trade at any given time without any problems

Pause Trade Notes:

Auditor Notes: The contract does not have a pause function but does have a 3 mins trade cooldown logic. The intention is to prevent MEV bots or front-runner bots from taking advantage of other investors.

Project Owner Notes:

Contract Ownership

The contract ownership of TresChain is not currently renounced. The ownership of the contract grants special powers to the protocol creators, making them the sole addresses that can call sensible ownable functions that may alter the state of the protocol.

The current owner is the address 0x9118f844F0D8da88c73331fd9aCF7D9Dc1906066 which can be viewed:
HERE

The owner wallet has the power to call the functions displayed on the privileged functions chart below, if the owner's wallet is compromised, they could exploit these privileges.

We recommend the team renounce ownership at the right time, if possible, or gradually migrate to a timelock with governing functionalities regarding transparency and safety considerations.

We recommend the team use a Multisignature Wallet if the contract is not going to be renounced; this will give the team more control over the contract.

Liquidity Ownership

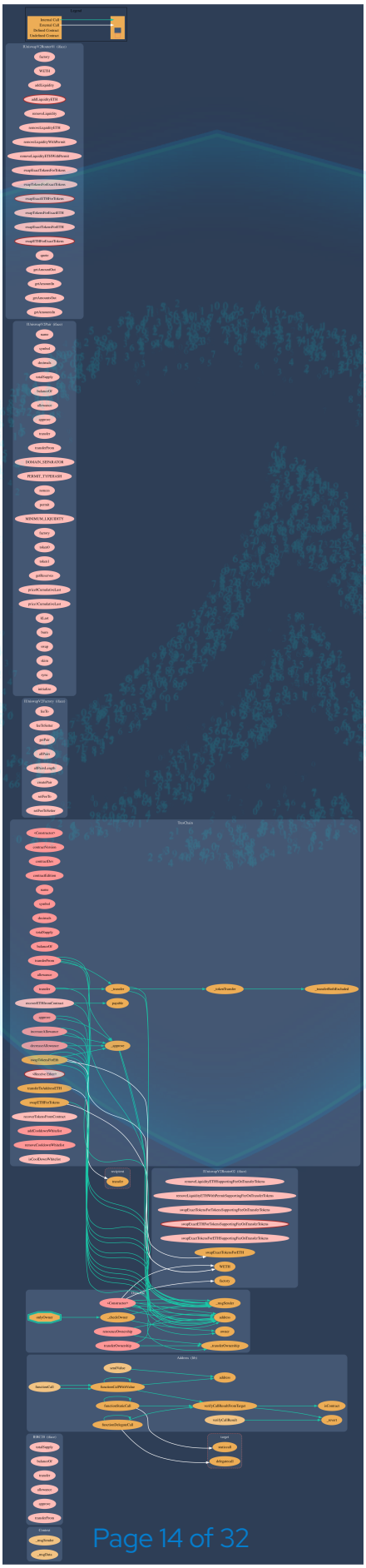
The token does not have liquidity at the moment of the audit, block 16373698

If liquidity is unlocked, then the token developers can do what is infamously known as 'rugpull'. Once investors start buying token from the exchange, the liquidity pool will accumulate more and more coins of established value (e.g., ETH or BNB or Tether). This is because investors are basically sending these tokens of value to the exchange, to get the new token. Developers can withdraw this liquidity from the exchange, cash in all the value and run off with it. Liquidity is locked by renouncing the ownership of liquidity pool (LP) tokens for a fixed time period, by sending them to a time-lock smart contract. Without ownership of LP tokens, developers cannot get liquidity pool funds back. This provides confidence to the investors that the token developers will not run away with the liquidity money. It is now a standard practice that all token developers follow, and this is what really differentiates a scam coin from a real one.

Read More

Call Graph

The contract for TresChain has the following call graph structure.



AegisX

KYC Information

The Project Owners of TresChain are not KYC'd. .

The owner wallet has the power to call the functions displayed on the priviliged functions chart below, if the owner wallet is compromised this privileges could be exploited.

We recommend the team to renounce ownership at the right timing if possible, or gradually migrate to a timelock with governing functionalities in respect of transparency and safety considerations.

KYC Information Notes:

Auditor Notes:

Project Owner Notes:

Smart Contract Vulnerability Checks

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	TresChain.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	TresChain.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	TresChain.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	TresChain.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	TresChain.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	TresChain.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	TresChain.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	TresChain.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	TresChain.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	TresChain.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	TresChain.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	TresChain.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	TresChain.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-113	Pass	Multiple calls are executed in the same transaction.	TresChain.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	TresChain.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	TresChain.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	TresChain.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	TresChain.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	TresChain.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	TresChain.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randonmness.	TresChain.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	TresChain.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	TresChain.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	TresChain.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	TresChain.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	TresChain.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	TresChain.sol	L: 0 C: 0

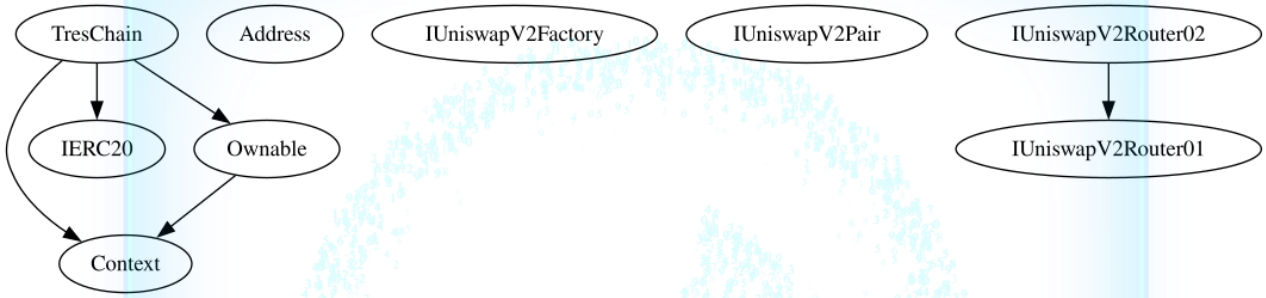
ID	Severity	Name	File	location
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	TresChain.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	TresChain.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	TresChain.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	TresChain.sol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

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

Inheritance

The contract for TresChain has the following inheritance structure.

The Project has a Total Supply of 25,000,000



Privileged Functions (onlyOwner)

Function Name	Parameters	Visibility
renounceOwnership		Public
transferOwnership	address newOwner	Public
addCooldownWhitelist	address whitelistAddy	Public
removeCooldownWhitelist	address whitelistAddy	Public

TRES-01 | Potential Sandwich Attacks.

Category	Severity	Location	Status
Security	Informational	TresChain.sol: 1026,13, 1098,13	Pending

Description

A sandwich attack might happen when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by frontrunning (before the transaction being attacked) a transaction to purchase one of the assets and make profits by back running (after the transaction being attacked) a transaction to sell the asset. The following functions are called without setting restrictions on slippage or minimum output amount, so transactions triggering these functions are vulnerable to sandwich attacks, especially when the input amount is large:

Function Name	Slippage
swapExactTokensForETH	50%
swapExactETHForTokensSupportingFeeOnTransferTokens	50%

Remediation

We recommend setting reasonable minimum output amounts, instead of 0, based on token prices when calling the aforementioned functions.

Project Action

An oracle has been implemented, commendably. However, with a 50% slippage, this still may result a sandwich attack.

Referrences:

What Are Sandwich Attacks in DeFi — and How Can You Avoid Them?.

TRES-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	Informational	TresChain.sol:	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
owner		Public
renounceOwnership		Public
transferOwnership	address newOwner	Public
contractVersion		Public
contractDev		Public
contractEdition		Public
name		Public
symbol		Public
decimals		Public
totalSupply		Public
balanceOf	address account	Public
transfer	address recipient, uint256 amount	Public

Function Name	Parameters	Visibility
allowance	address owner, address spender	Public
approve	address spender, uint256 amount	Public
transferFrom	address sender, address recipient, uint256 amount	Public
increaseAllowance	address spender, uint256 addedValue	Public
decreaseAllowance	address spender, uint256 subtractedValue	Public
addCooldownWhitelist	address whitelistAddy	Public
removeCooldownWhitelist	address whitelistAddy	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.

Project Action

Although most of the listed functions are pure functions that aren't called by other functions, please review these public functions and change those that can be changed to external. ie. approve, addCooldownWhitelist.

References:

external vs public best practices.

TRES-08 | Dead Code Elimination.

Category	Severity	Location	Status
Coding Style	Minor	TresChain.sol: 1008,5, 1079,5	Pending

Description

Functions that are not used in the contract, and make the size of the codes unncecessarily bigger.

swapTokensForEth
swapETHForTokens

Remediation

Remove unused functions. dead-code elimination (also known as DCE, dead-code removal, dead-code stripping, or dead-code strip) is a compiler optimization to remove code which does not affect the program results. Removing such code has several benefits: it shrinks program size, an important consideration in some contexts, and it allows the running program to avoid executing irrelevant operations, which reduces its running time. It can also enable further optimizations by simplifying program structure.

https://docs.soliditylang.org/en/latest/cheatsheet.html

Project Action

Pending Customer Response

TRES-10 | Initial Token Distribution.

Category	Severity	Location	Status
Centralization / Privilege	Minor	TresChain.sol: 839,9	Pending

Description

All of the TresChain tokens are sent to the contract deployer when deploying the contract.
This could be a centralization risk as the deployer can distribute tokens without obtaining the consensus of the community.

Remediation

We recommend the team to be transparent regarding the initial token distribution process, and the team shall make enough efforts to restrict the access of the private key.

Project Action

The total supply goes to the contract deployer, and it was found to be in the deployer at the time of this audit.

TRES-12 | Centralization Risks In The onlyOwner Role(s)

Category	Severity	Location	Status
Centralization / Privilege	Informational	TresChain.sol: 1122,5	Pending

Description

In the contract TresChain, the role onlyOwner has authority over the functions that lead to centralization risks.

Any compromise to the onlyOwner account(s) may allow the hacker to take advantage of this authority.

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

_owner has a right to exclude addresses from the 3-minute trade cooldown.

Social Media Checks

Social Media	URL	Result
Website	https://www.treschain.com/	Pass
Telegram	https://t.me/treslechesfinance/39	Pass
Twitter	https://twitter.com/treslecheschain	Pass
OtherSocial	https://instagram.com/treslecheschain	Pass

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

Social Media Information Notes:

Auditor Notes: undefined

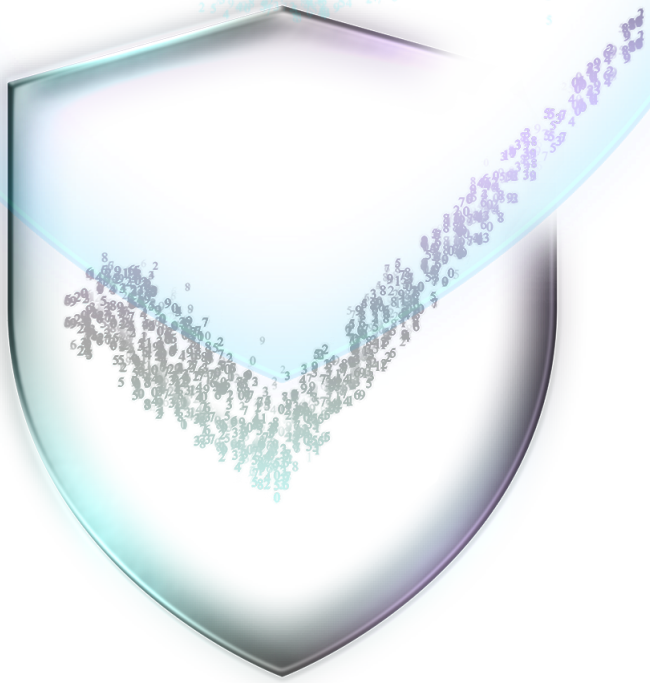
Project Owner Notes:

Assessment Results

Score Results

Review	Score
Overall Score	91/100
Auditor Score	89/100
Review by Section	**Score**
Manual Scan Score	16/18
SWC Scan Score	37/37
Advance Check Score	38/45

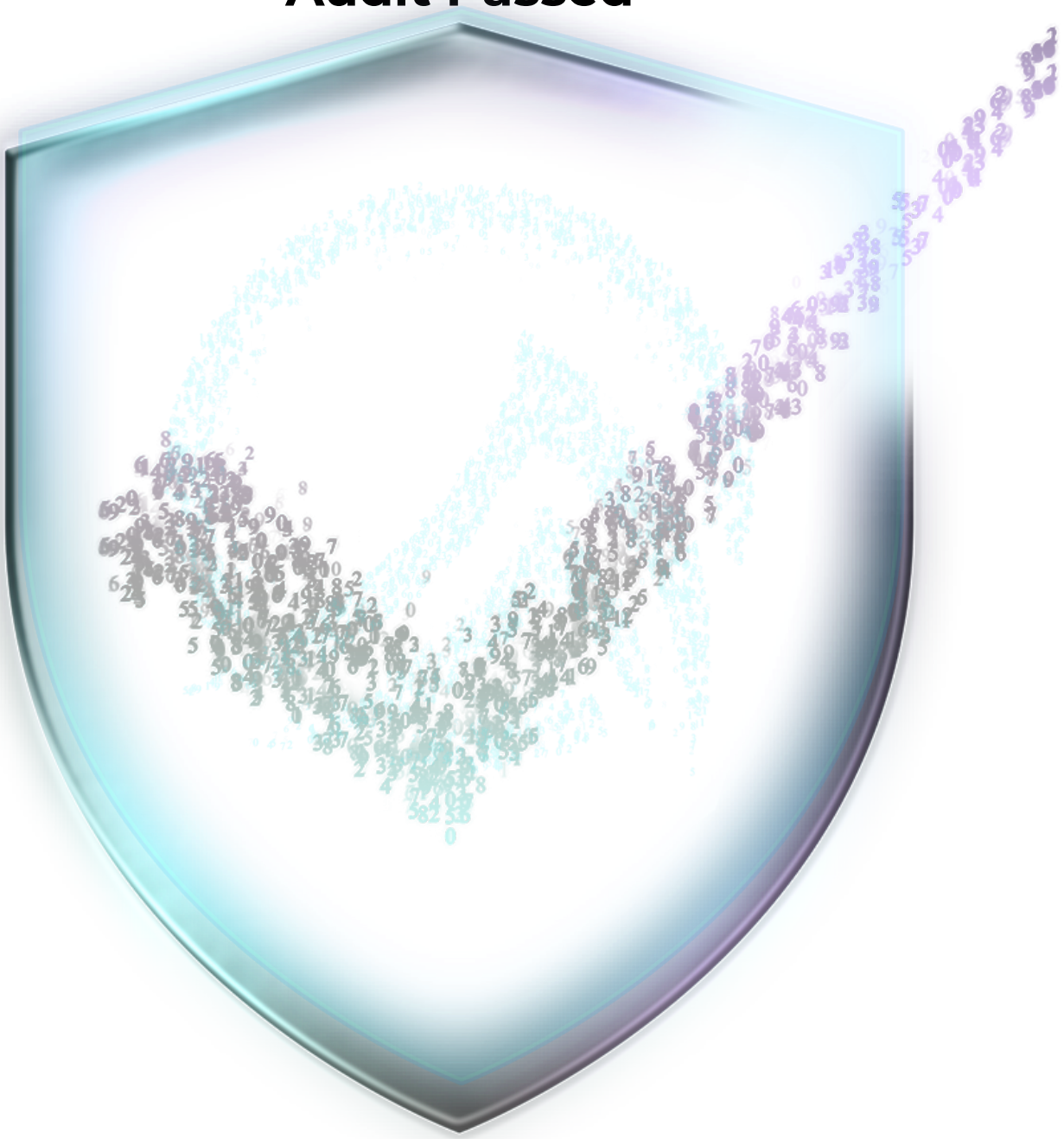
The maximum score is 100, however to attain that value the project must pass the reviews and provide all the data needed for the assessment. Minimum score to pass is 80 points. If a project fails to attain 80 and/or has unresolved critical and/or major and/or medium finding(s) in the Palladium tier assessments, an automatic failure is given. Read our notes and final assessment below.



PASSED

Assessment Results

Auditor Score = 89
Audit Passed



Important Notes from the Auditor:

- The contract does not have a pause trade function but does have a 3 mins trade cooldown logic. The declared intention is to prevent MEV bots or front-runner bots from taking advantage of other investors.
- Two functions, swapTokensForEth & swapETHForTokens, were found to be dead codes.
- The total supply goes to the contract deployer, and it was found to be in the deployer at the time of this audit.
- _owner has a right to exclude addresses from the 3-minute trade cooldown.
- The 3-minute trade cooldown has been verified to work properly.

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

AegisX 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 AegisX is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will AegisX 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 AegisX 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.