CFG NINJA AUDITS

Security Assessment

Arcstar Token

June 5, 2023

Audit Status: Fail

Audit Edition: Advance

POWERED BY
BLADE POOL

Risk Analysis

Classifications of Manual Risk Results

Classification	Description
🔴 Critical	Danger or Potential Problems.
🟠 Major	Be Careful or Fail test.
🟢 Minor	Pass, Not-Detected or Safe Item.
ℹ Informational	Function Detected

Manual Code Review Risk Results

Contract Priviledge	Description
🟢 Buy Tax	5
🟢 Sale Tax	5
🟢 Cannot Sale	Pass
🔴 Cannot Sale	Fail, if set to 100% setSniperFee
🔴 Max Tax	100
ℹ Modify Tax	Detected
🟢 Fee Check	Fail
🔴 Is Honeypot?	Detected, if own address or presale address is added on setSniperFee
🟢 Trading Cooldown	Not Detected
🟢 Can Pause Trade?	Pass
🟢 Pause Transfer?	Not Detected

Contract Priviledge	Description
🟢 Max Tx?	Pass
🟢 Is Anti Whale?	Not Detected
🔴 Is Anti Bot?	Detected
🟢 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
ℹ Owner	0x18550c57785fd3c0dcaa82ec5a60bd62a445feb9
🟢 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.

Project Overview

Token Summary

Parameter	Result
Address	0xbB4F5e6F6eF38919e12099810f8CCd082D24A1E4
Name	Arcstar
Token Tracker	Arcstar (ARCSTAR)
Decimals	18
Supply	100,000,000,000
Platform	Ethereum
compiler	v0.8.17+commit.8df45f5f
Contract Name	Arcstar
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/token/0xe953f346876D4Bf0074718de931156F39Bc5B307#code
Payment Tx	0x513e9aebd6b36804cca46a6eafa913e4d30de29cadebe80dc2a6ad315a3c5f7f

Project Overview

Simulation Summary

Parameter	Result
Transfer From Owner	Pass
Transfer From Holder	Pass
Add Liquidity	Pass
RemoveLiquidity	Pass
Buy from Owner	Pass
Buy from Holder	Pass
Sale from Owner	Pass
Sale from Holder	Pass
Remove Liquidity	Pass
SwapAndLiquify	Pass
SwapAndSale w/Fee	Pass
SwapAndSale TX	0x9dc7e06de78457e9F15a2d646Ed0e448C45E1F45
SwapAndSaleNoFee	Pass
SwapAndSale No/Fee TX	0x9dc7e06de78457e9F15a2d646Ed0e448C45E1F45
ExcludeFromFees	Pass

Parameter	Result
LaunchPad	PinkSale
Pool Creation	Pass
Pool Creation TX	https://testnet.bscscan.com/tx/0xa4bc84ce8c5e961f6e8d2ca9cc5a68c0413bfc34240f5ad58954a4bb75ee6920
Pool Finalize	Pass
Pool Finalize TX	https://testnet.bscscan.com/tx/0xa4bc84ce8c5e961f6e8d2ca9cc5a68c0413bfc34240f5ad58954a4bb75ee6920
Enable	Pass

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
Arcstar	0xbB4F5e6F6eF38919e12099810f8CCd082D24A1E4	Yes

TestNet Contract Assessed
Contract Name

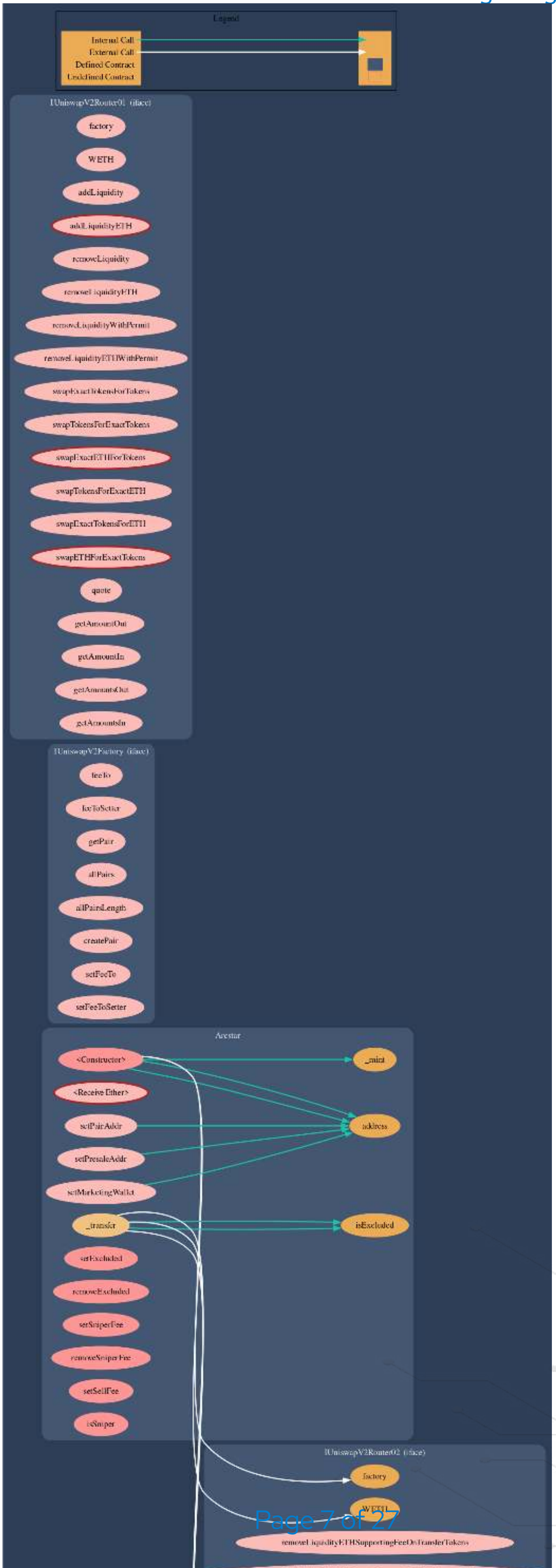
Name	Contract	Live
Arcstar	0x9dc7e06de78457e9F15a2d646Ed0e448C45E1F45	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
Arcstar	5027db676652e5576a737326c7c358a4283bd610	Arcstar.sol
Arcstar	d0573b947cdd287ef57810fa9a7cb84c28b8a50b	IUniswapV2Factory.sol
Arcstar	0269305d732acc36030c66f0df8ba1b8be849365	IUniswapV2Router02.sol
Arcstar		

Call Graph

The contract for Arcstar 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	Arcstar.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	Arcstar.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	Arcstar.sol	L: 0 C: 0
SWC-103	Low	A floating pragma is set.	Arcstar.sol	L: 4 C: 0
SWC-104	Pass	Unchecked Call Return Value.	Arcstar.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	Arcstar.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	Arcstar.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	Arcstar.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	Arcstar.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	Arcstar.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	Arcstar.sol	L: 0 C: 0

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

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

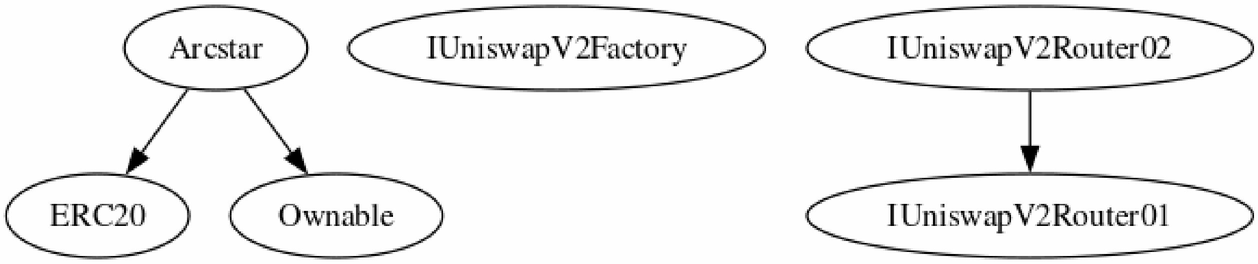
Pragma statements can be allowed to float when a contract is intended for consumption by other developers, as in the case with contracts in a library or EthPM package. Otherwise, the developer would need to manually update the pragma in order to compile locally.

References:

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

Inheritance

The contract for Arcstar 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	address newOwner	Public
setSellFee		Public
removeSniperFee		Public
setSniperFee		Public
removeExcluded		Public
setExcluded		Public
setPresaleAddr		External
setPairAddr		External
setMarketingWallet		External

Smart Contract Advance Checks

ID	Severity	Name	Result	Status
ARCSTAR-01	Minor	Potential Sandwich Attacks.	Pass	Not Detected
ARCSTAR-02	Minor	Function Visibility Optimization	Fail	Detected
ARCSTAR-03	Minor	Lack of Input Validation.	Fail	Detected
ARCSTAR-04	Major	Centralized Risk In addLiquidity.	Pass	Not Detected
ARCSTAR-05	Minor	Missing Event Emission.	Fail	Detected
ARCSTAR-06	Minor	Conformance with Solidity Naming Conventions.	Pass	Detected
ARCSTAR-07	Minor	State Variables could be Declared Constant.	Pass	Not Detected
ARCSTAR-08	Minor	Dead Code Elimination.	Pass	Not Detected
ARCSTAR-09	Major	Third Party Dependencies.	Pass	Not Detected
ARCSTAR-10	Major	Initial Token Distribution.	Fail	Detected
ARCSTAR-11	Critical	Sniperbot is present on the transfer.	Fail	Acknowledge by Project.
ARCSTAR-12	Major	Centralization Risks In The X Role	Pass	Not Detected
ARCSTAR-13	Informational	Extra Gas Cost For User..	Pass	Not Detected

ID	Severity	Name	Result	Status
ARCSTAR-14	Medium	Unnecessary Use Of SafeMath	Pass	Not Detected
ARCSTAR-15	Medium	Symbol Length Limitation due to Solidity Naming Standards.	Pass	Not Detected
ARCSTAR-16	Critical	Taxes can be up to 100%	Fail	Detected
ARCSTAR-17	Informational	Conformance to numeric notation best practice.	Pass	Not Detected
ARCSTAR-18	Medium	Stop Transactions by using Enable Trade.	Pass	Not Detected

ARCSTAR-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	Minor	Arcstar.sol: L: 124 C: 24	Detected

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
setSellFee		internal
removeSniperFee		internal
setSniperFee		public
removeExcluded		public
setExcluded		public
setExcluded		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.

ARCSTAR-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	● Minor	Arcstar.sol: 123,14	Detected

Description

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

The given input is missing the check for the removeSniperFee,setSniperFee,removeExcluded,setExcluded onlyOwner are 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. removeSniperFee,setSniperFee,removeExcluded,setExcluded onlyOwner are missing required function.

ARCSTAR-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	🟢 Minor	Arcstar.sol: 123, 14	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.

ARCSTAR-10 | Initial Token Distribution.

Category	Severity	Location	Status
Centralization / Privilege	🟠 Major	Arcstar.sol: 39,13	Detected

Description

All of the Arcstar 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

_mint(msg.sender, initialSupply); was found on the constructor.

ARCSTAR-11 | Sniperbot is present on the transfer..

Category	Severity	Location	Status
Optimization	🔴 Critical	Arcstar.sol: 58,14	Acknowledge by Project.

Description

During the transfer it perform an additional tax to snipers. if (sellSniperFee[sender] > 0 && (recipient == pairAddr || sender != pairAddr)) {tax = baseUnit * uint256(sellSniperFee[sender]); } else if (buySniperFee[recipient] > 0 && sender == pairAddr)

Remediation

Review current sniperBot logic to ensure it only capture early buyers, otherwise this is effectively a blacklist function.

Project Action

The reason is that we're afraid of liquidity snipers which get tokens from nowhere, they change addresses and auto sniper doesn't work unfortunately as they do this during liquidity adding. While anyone anytime can add liquidity, these guys can easily frontrun the transaction and get a lot of tokens and drain liq.

ARCSTAR-16 | Taxes can be up to 100%.

Category	Severity	Location	Status
Logical Issue	● Critical	Arcstar.sol: 102, 9	Detected

Description

The current definition of taxes can be set up to 100% for specific wallets, we suggest to modify the function not to be dynamic but to be a static resolution.

```
function setSniperFee(address[] memory account, uint8 _sellFee, uint8 _buyFee) public onlyOwner {
  for (uint256 i = 0; i < account.length; i++) {
    if (_sellFee > 0) {
      sellSniperFee[account[i]] = _sellFee;
    }
    if (_buyFee > 0) {
      buySniperFee[account[i]] = _buyFee;
    }
  }
}
```

due to the logic written in here may results in a honeypot.

Remediation

We advise the team to review the following logic..

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.
🟠 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.
ℹ️ 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	2	0	0
🟠 Major	1	0	0
🟡 Medium	0	0	0
🟢 Minor	2	0	0
ℹ️ Informational	1	0	0
Total	6	0	0

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/ArcstarDAO	Pass
Other	https://medium.com/@arcstarbsc	Pass
Website	https://arcstardao.com/	Pass
Telegram	https://t.me/ArcstarDAO	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	79/100
Auditor Score	75/100
Review by Section	**Score**
Manual Scan Score	29/53
SWC Scan Score	36 /37
Advance Check Score	14 /19

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 Fail

Assessment Results

Important Notes:

- No issues or vulnerabilities were found.
- The project has a sniperFee that seems to be higher than 5, we are reviewing the current sniper function on testnet to validate the process.
- The project owner can setFee to 100% to an array of wallets, this wallet won't be able to sell the tokens, however, during testing, they were able to buy and 100% of the tokens were sent to the marketing wallet.

Auditor Score =75
Audit Fail

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 ʻas 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.