



CFG NINJA AUDITS

Security Assessment

Snoopy Inu Token

May 29, 2023

Audit Status: Pass

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. |
|  Informational | Function Detected |

Manual Code Review Risk Results

| Contract Priviledge | Description |
|--|--------------|
|  Buy Tax | 0 |
|  Sale Tax | 3 |
|  Cannot Sale | Pass |
|  Cannot Sale | Pass |
|  Max Tax | 0 |
|  Modify Tax | Not Detected |
|  Fee Check | Pass |
|  Is Honeypot? | Not Detected |
|  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? | Not 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 |
| 🔍 Owner | 0x33a449f597ab4caaa1efb9a24396a76a00ddb703 |
| ● 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 | 0xf7de2FAbeeD2F7586d5A22E219A16f216b90019a |
| Name | Snoopy Inu |
| Token Tracker | Snoopy Inu (Snoopy) |
| Decimals | 18 |
| Supply | 420,690,000,000,000 |
| Platform | Binance Smart Chain |
| compiler | v0.8.19+commit.7dd6d404 |
| Contract Name | Token |
| Optimization | Yes with 200 runs |
| LicenseType | MIT |
| Language | Solidity |
| Codebase | https://bscscan.com/token/0xf7de2FAbeeD2F7586d5A22E219A16f216b90019a#code |
| Payment Tx | 0x635380d660decad0ca8c630c6647a48d14c5f964d905f67a362220c809600eff |



Main Contract Assessed

Contract Name

| Name | Contract | Live |
|------------|--|------|
| Snoopy Inu | 0xf7de2FAbeeD2F7586d5A22E219A16f216b90019a | Yes |

TestNet Contract Assessed

Contract Name

| Name | Contract | Live |
|------------|--|------|
| Snoopy Inu | 0xf062432bd035918d93355561dbdc8102f2c1f9a0 | Yes |

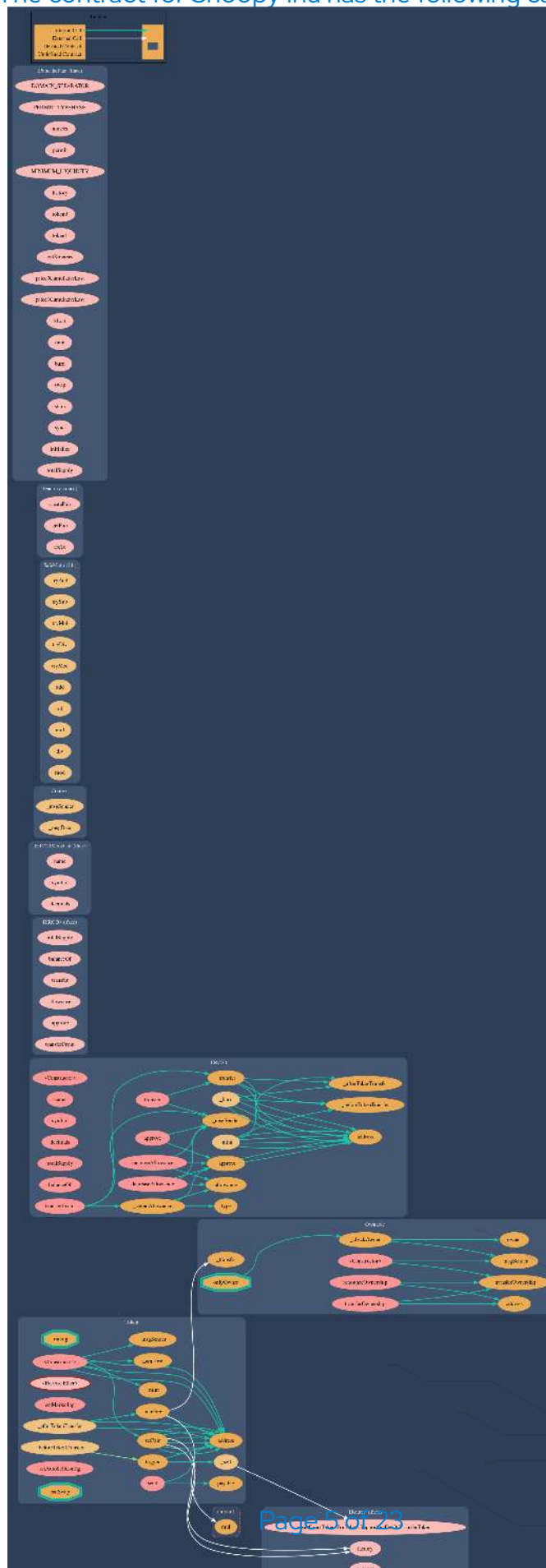
Solidity Code Provided

| SolID | File Sha-1 | FileName |
|--------|--|--------------------|
| Snoopy | 5846d01e25471f0f69e8c74200348b4f7b244f71 | Snoopy.sol |
| Snoopy | 88ee5db718680555f1adec6b749080d6ae5aa919 | ERC20.sol |
| Snoopy | 3f8f9d66083281998547ead9e2a599f5e3d049f8 | IERC20.sol |
| Snoopy | 87b62db9a86c0b9bbc58b51d0d2ae7a8b768880 | IERC20Metadata.sol |
| | 0 | |



Call Graph

The contract for Snoopy Inu 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 | Snoopy.sol | L: 0 C: 0 |
| SWC-101 | Pass | Integer Overflow and Underflow. | Snoopy.sol | L: 0 C: 0 |
| SWC-102 | Pass | Outdated Compiler Version file. | Snoopy.sol | L: 0 C: 0 |
| SWC-103 | Low | A floating pragma is set. | Snoopy.sol | L: 2 C: 0 |
| SWC-104 | Pass | Unchecked Call Return Value. | Snoopy.sol | L: 0 C: 0 |
| SWC-105 | Pass | Unprotected Ether Withdrawal. | Snoopy.sol | L: 0 C: 0 |
| SWC-106 | Pass | Unprotected SELFDESTRUCT Instruction | Snoopy.sol | L: 0 C: 0 |
| SWC-107 | Pass | Read of persistent state following external call. | Snoopy.sol | L: 0 C: 0 |
| SWC-108 | Pass | State variable visibility is not set.. | Snoopy.sol | L: 0 C: 0 |
| SWC-109 | Pass | Uninitialized Storage Pointer. | Snoopy.sol | L: 0 C: 0 |
| SWC-110 | Pass | Assert Violation. | Snoopy.sol | L: 0 C: 0 |



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



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

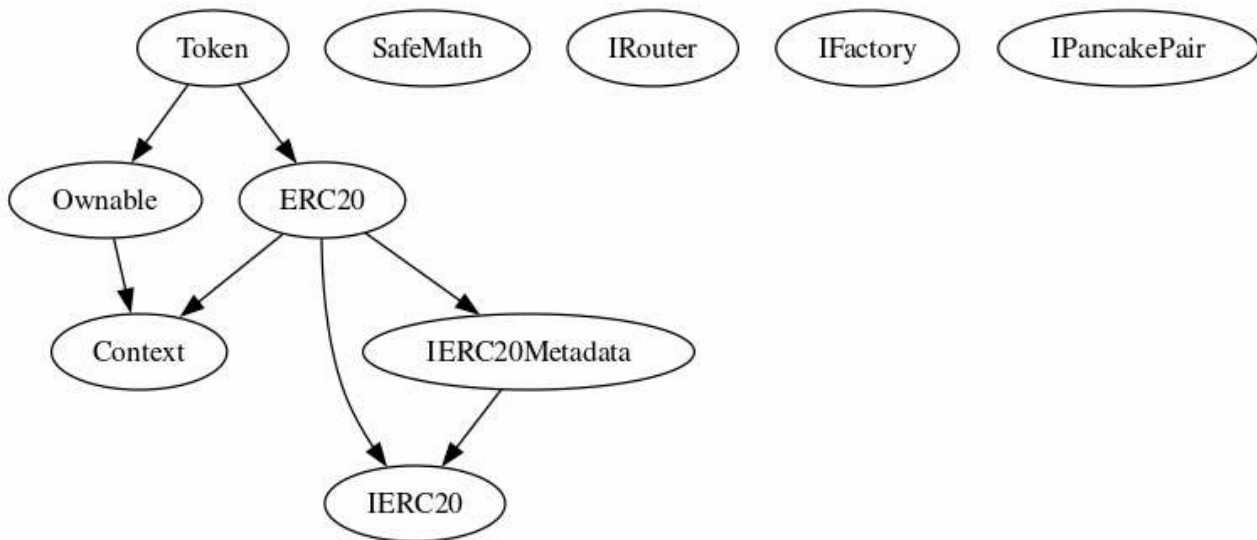
References:

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



Inheritance

The contract for Snoopy Inu 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 |
| setMarketing | | Public |
| setAutoSellConfig | | External |



Smart Contract Advance Checks



| ID | Severity | Name | Result | Status |
|-----------|---------------|--|--------|--------------|
| Snoopy-01 | Minor | Potential Sandwich Attacks. | Pass | Not-Found |
| Snoopy-02 | Minor | Function Visibility Optimization | Fail | Detected |
| Snoopy-03 | Minor | Lack of Input Validation. | Fail | Detected |
| Snoopy-04 | Major | Centralized Risk In addLiquidity. | Pass | Not Detected |
| Snoopy-05 | Minor | Missing Event Emission. | Fail | Detected |
| Snoopy-06 | Minor | Conformance with Solidity Naming Conventions. | Pass | Not Detected |
| Snoopy-07 | Minor | State Variables could be Declared Constant. | Pass | Not-Found |
| Snoopy-08 | Minor | Dead Code Elimination. | Pass | Not-Found |
| Snoopy-09 | Major | Third Party Dependencies. | Pass | Not-Found |
| Snoopy-10 | Major | Initial Token Distribution. | Pass | Not-Found |
| Snoopy-11 | Minor | Multisend is present in code. | Pass | Resolved |
| Snoopy-12 | Major | Centralization Risks In The X Role | Pass | Not-Found |
| Snoopy-13 | Informational | Extra Gas Cost For User.. | Pass | Not-Found |
| Snoopy-14 | Medium | Unnecessary Use Of SafeMath | Fail | Acknowledge |
| Snoopy-15 | Medium | Symbol Length Limitation due to Solidity Naming Standards. | Pass | Not-Found |



| ID | Severity | Name | Result | Status |
|-----------|---------------|--|--------|--------------|
| Snoopy-16 | Medium | Invalid collection of Taxes during Transfer. | Pass | Not-Found |
| Snoopy-17 | Informational | Conformance to numeric notation best practice. | Pass | Not-Found |
| Snoopy-18 | Medium | Stop Transactions by using Enable Trade. | Pass | Not-Detected |



Snoopy-02 | Function Visibility Optimization.

| Category | Severity | Location | Status |
|------------------|---|--------------------------|--|
| Gas Optimization |  Minor | Snoopy.sol: L: 296 C: 11 |  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 |
|---------------|------------|------------|
| marketing | | internal |
| isTrading | | internal |
| token0 | | internal |
| token1 | | internal |
| router | | internal |
| pair | | 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.



Snoopy-03 | Lack of Input Validation.

| Category | Severity | Location | Status |
|---------------|---|-------------------|--|
| Volatile Code |  Minor | Snoopy.sol: 56,14 |  Detected |

Description

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

The given input is missing the check for the setMarketing,setAutoSellConfig is 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. setMarketing,setAutoSellConfig is missing required function.



Snoopy-05 | Missing Event Emission.

| Category | Severity | Location | Status |
|---------------|---|--------------------|--|
| Volatile Code |  Minor | Snoopy.sol: 56, 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.



Snoopy-14 | Unnecessary Use Of SafeMath

| Category | Severity | Location | Status |
|---------------|--|-----------------|---|
| Logical Issue |  Medium | Snoopy.sol: 7,9 |  Acknowledge |

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








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 | 1 | 0 | 0 |
|  Medium | 0 | 0 | 0 |
|  Minor | 2 | 0 | 0 |
|  Informational | 1 | 0 | 0 |
| Total | 4 | 0 | 0 |



Social Media Checks

| Social Media | URL | Result |
|--------------|---|--------|
| Twitter | https://twitter.com/CryptoSnoopy_ | Pass |
| Other | https://cryptosnoopy.art/Whitepaper.pdf | Pass |
| Website | http://cryptosnoopy.art | Pass |
| Telegram | https://t.me/CryptoSnoopy_Finance | 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 | 80/100 |
| Auditor Score | 80/100 |
| Review by Section | Score |
| Manual Scan Score | 35/53 |
| SWC Scan Score | 36 /37 |
| Advance Check Score | 9 /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 Passed



Assessment Results

Important Notes:

- No issues or vulnerabilities were found.
- A few items have been resolved.
- Always DYOR on projects, this project may need to be monitored.

Auditor Score =80
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 requirements 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

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 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 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 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.

