# CFG NINJA AUDITS

## Security Assessment

# Bunnies Battle Token Token

December 22, 2022

# Table of Contents

# Assessment Summary

This report has been prepared for Bunnies Battle Token Token on the Binance network. CFGNINJA 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 | 1 | 1 | 0 |
| 🟠 Major | 0 | 0 | 0 |
| 🟡 Medium | 1 | 1 | 0 |
| 🟢 Minor | 2 | 2 | 0 |
| 🔵 Informational | 1 | 1 | 0 |
| Total | 5 | 5 | 0 |

# Project Overview

## Token Summary

| Parameter | Result |
|---|---|
| Address | 0xa646AA42babc29ec42300bdbfC9E82D14375c292 |
| Name | Bunnies Battle Token |
| Token Tracker | Bunnies Battle Token (BBT) |
| Decimals | 18 |
| Supply | 1,000,000,000 |
| Platform | Binance |
| compiler | v0.8.17+commit.8df45f5f |
| Contract Name | BBT |
| Optimization | Yes with 200 runs |
| LicenseType | MIT |
| Language | Solidity |
| Codebase | https://bscscan.com/address/0xa646AA42babc29ec42300bdbfC9E82D14375c292 |
| Payment Tx | 0x4e4632d06b1e2323461a07e8402ac4609fd78b9defdce524dc82282b7eb67de3 |

# Project Overview

## Risk Analysis Summary

| Parameter | Result |
| --- | --- |
| Buy Tax | 3% |
| Sale Tax | 5% |
| Is honeypot? | Clean |
| Can edit tax? | No |
| Is anti whale? | No |
| Is blacklisted? | Yes |
| Is whitelisted? | No |
| Holders | 1 |
| Confidence Level | Medium |

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 |
|------|----------|------|
| Bunnies Battle Token | 0xa646AA42babc29ec42300bdbfC9E82D14375c292 | Yes |

# TestNet Contract was Not Assessed

# Solidity Code Provided

| SolID | File Sha-1 | FileName |
|-------|-----------|----------|
| bbt | db3f97b1ccaf927588b9e6e14f6c2d00ee32f2df | bbt2.sol |

# Mint Check

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

The Project has a Total Supply of 1,000,000,000 and cannot mint any more than the Max Supply.

**Mint Notes:**

**Auditor Notes:**

**Project Owner Notes:**

Owner can't mint new coins

# Fees Check

## The project owners of Bunnies Battle Token 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.
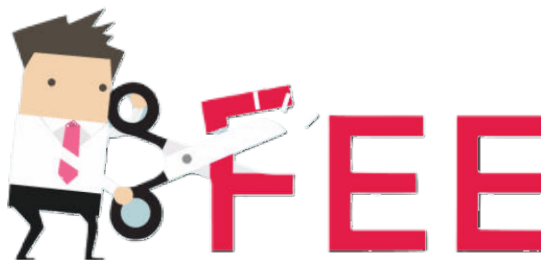
**Tax Fee Notes:**

**Auditor Notes:** The contract currently has 3% buy and 5% sale taxes, and cannot be set higher than 4.

**Project Owner Notes:**



Fees can be changed up to a maximum of 25%

# Blacklist Check

The project owners of Bunnies Battle Token have the ability to Blacklist holders from transferring their tokens.

We recommend the Team be careful with a blacklist function as this can prevent a holder from buying/selling/transferring their assets. Malicious or compromised owners can trap contracts relying on tokens with a blacklist

**Blacklist Notes:**

**Auditor Notes:** blacklist is set for those tx origin from early buy

**Project Owner Notes:** undefined

# MaxTx Check

The Project Owners of Bunnies Battle Token can set max tx amount.

The ability to set MaxTx can be used as bad actor, this can limit the ability of investors to sale their tokens at any given time if is set too low..

We recommend the project to set MaxTx to Total Supply or simiar to avoid swap or transfer from failures

**MaxTX Notes:**

**Auditor Notes:** topHoldersAmount is a max wallet

**Project Owner Notes:**

Project Has MaxTX

# Pause Trade Check

**The Project Owners of Bunnies Battle Token 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: There is an Open Trade so holders cant trade until is enable.**

**Project Owner Notes:**



Owner can't pause trading

# Contract Ownership

The contract ownership of Bunnies Battle Token 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 0x824d0788fe28b534d6b898fa50fb79ae7c0e7bcb 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 24123365

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

# KYC Information

## The Project Owners of Bunnies Battle Token is not KYC.

**KYC Information Notes:**

**Auditor Notes:**

**Project Owner Notes:**

# Smart Contract Vulnerability Checks

| ID | Severity | Name | File | location |
|---|---|---|---|---|
| SWC-100 | Pass | Function Default Visibility | bbt2.sol | L: 0 C: 0 |
| SWC-101 | Pass | Integer Overflow and Underflow. | bbt2.sol | L: 0 C: 0 |
| SWC-102 | Pass | Outdated Compiler Version file. | bbt2.sol | L: 0 C: 0 |
| SWC-103 | Low | A floating pragma is set. | bbt2.sol | L: 7 C: 0 |
| SWC-104 | Pass | Unchecked Call Return Value. | bbt2.sol | L: 0 C: 0 |
| SWC-105 | Pass | Unprotected Ether Withdrawal. | bbt2.sol | L: 0 C: 0 |
| SWC-106 | Pass | Unprotected SELFDESTRUCT Instruction | bbt2.sol | L: 0 C: 0 |
| SWC-107 | Pass | Read of persistent state following external call. | bbt2.sol | L: 0 C: 0 |
| SWC-108 | Pass | State variable visibility is not set.. | bbt2.sol | L: 405 C: 9 |
| SWC-109 | Pass | Uninitialized Storage Pointer. | bbt2.sol | L: 0 C: 0 |
| SWC-110 | Pass | Assert Violation. | bbt2.sol | L: 0 C: 0 |
| SWC-111 | Pass | Use of Deprecated Solidity Functions. | bbt2.sol | L: 0 C: 0 |
| SWC-112 | Pass | Delegate Call to Untrusted Callee. | bbt2.sol | L: 0 C: 0 |

| ID | Severity | Name | File | location |
|---|---|---|---|---|
| SWC-113 | Pass | Multiple calls are executed in the same transaction. | bbt2.sol | L: 0 C: 0 |
| SWC-114 | Pass | Transaction Order Dependence. | bbt2.sol | L: 0 C: 0 |
| SWC-115 | Low | Authorization through tx.origin. | bbt2.sol | L: 1375 C: 27, L: 1413 C: 12 |
| SWC-116 | Pass | A control flow decision is made based on The block.timestamp environment variable. | bbt2.sol | L: 0 C: 0 |
| SWC-117 | Pass | Signature Malleability. | bbt2.sol | L: 0 C: 0 |
| SWC-118 | Pass | Incorrect Constructor Name. | bbt2.sol | L: 0 C: 0 |
| SWC-119 | Pass | Shadowing State Variables. | bbt2.sol | L: 0 C: 0 |
| SWC-120 | Pass | Potential use of block.number as source of randonmness. | bbt2.sol | L: 0 C: 0 |
| SWC-121 | Pass | Missing Protection against Signature Replay Attacks. | bbt2.sol | L: 0 C: 0 |
| SWC-122 | Pass | Lack of Proper Signature Verification. | bbt2.sol | L: 0 C: 0 |
| SWC-123 | Pass | Requirement Violation. | bbt2.sol | L: 0 C: 0 |
| SWC-124 | Pass | Write to Arbitrary Storage Location. | bbt2.sol | L: 0 C: 0 |
| SWC-125 | Pass | Incorrect Inheritance Order. | bbt2.sol | L: 0 C: 0 |
| SWC-126 | Pass | Insufficient Gas Griefing. | bbt2.sol | L: 0 C: 0 |

| ID | Severity | Name | File | location |
|---|---|---|---|---|
| SWC-127 | Pass | Arbitrary Jump with Function Type Variable. | bbt2.sol | L: 0 C: 0 |
| SWC-128 | Pass | DoS With Block Gas Limit. | bbt2.sol | L: 0 C: 0 |
| SWC-129 | Pass | Typographical Error. | bbt2.sol | L: 0 C: 0 |
| SWC-130 | Pass | Right-To-Left-Override control character (U+202E). | bbt2.sol | L: 0 C: 0 |
| SWC-131 | Pass | Presence of unused variables. | bbt2.sol | L: 0 C: 0 |
| SWC-132 | Pass | Unexpected Ether balance. | bbt2.sol | L: 0 C: 0 |
| SWC-133 | Pass | Hash Collisions with Multiple Variable Length Arguments. | bbt2.sol | L: 0 C: 0 |
| SWC-134 | Pass | Message call with hardcoded gas amount. | bbt2.sol | L: 0 C: 0 |
| SWC-135 | Pass | Code With No Effects (Irrelevant/Dead Code). | bbt2.sol | L: 0 C: 0 |
| SWC-136 | Pass | Unencrypted Private Data On-Chain. | bbt2.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.

# Smart Contract Vulnerability Details

## SWC-115 - Authorization through tx.origin

### CWE-477: Use of Obsolete Function

### Description:

tx.origin is a global variable in Solidity which returns the address of the account that sent the transaction. Using the variable for authorization could make a contract vulnerable if an authorized account calls into a malicious contract. A call could be made to the vulnerable contract that passes the authorization check since tx.origin returns the original sender of the transaction which in this case is the authorized account.

### Remediation:

tx.origin should not be used for authorization. Use msg.sender instead.

### References:
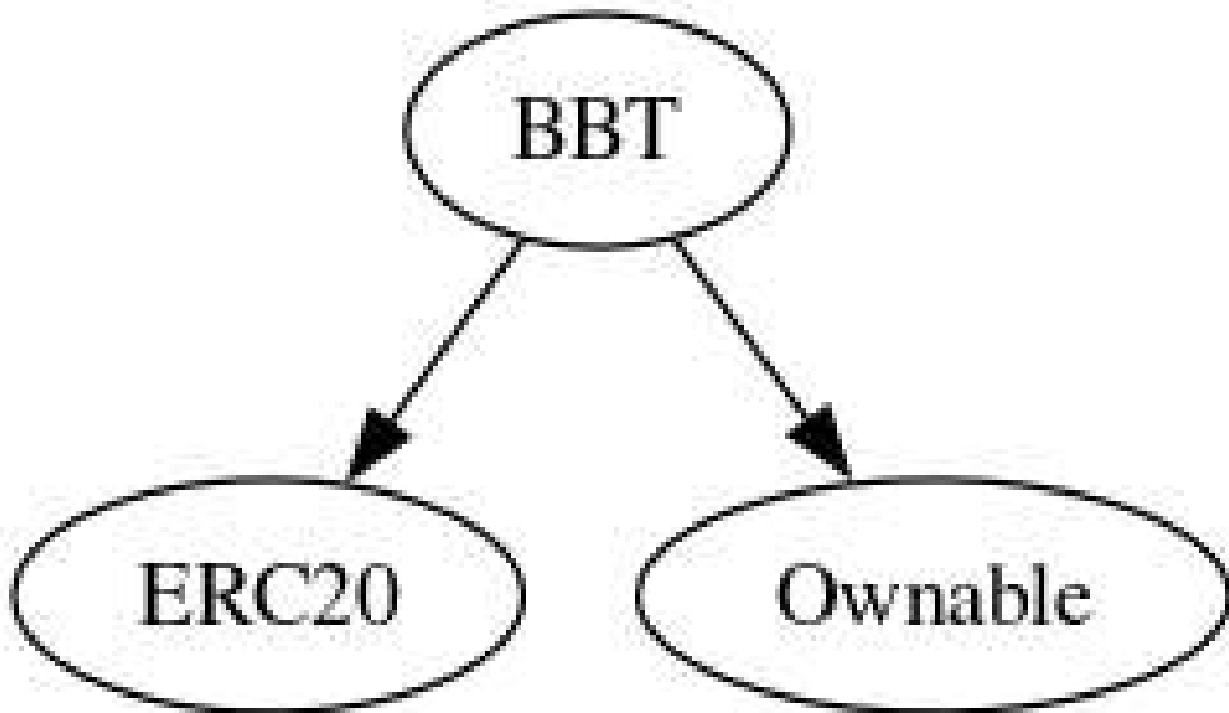
Solidity Documentation - tx.origin

Ethereum Smart Contract Best Practices - Avoid using tx.origin

SigmaPrime - Visibility.

# Inheritance

The contract for Bunnies Battle Token has the following inheritance structure.

# Smart Contract Advance Checks

| ID | Severity | Name | Result | Status |
|---|---|---|---|---|
| BBT-01 | Minor | Potential Sandwich Attacks. | Fail | Pending |
| BBT-02 | Minor | Function Visibility Optimization | Pass | Pending |
| BBT-03 | Minor | Lack of Input Validation. | Fail | Pending |
| BBT-04 | Major | Centralized Risk In addLiquidity. | Pass | Not Found |
| BBT-05 | Major | Missing Event Emission. | Fail | Pending |
| BBT-06 | Minor | Conformance with Solidity Naming Conventions. | Pass | Not Found |
| BBT-07 | Minor | State Variables could be Declared Constant. | Pass | Not Found |
| BBT-08 | Major | Dead Code Elimination. | Pass | Pending |
| BBT-09 | Major | Third Party Dependencies. | Pass | Not Found |
| BBT-10 | Major | Initial Token Distribution. | Fail | Pending |
| BBT-11 | Critical | distributeTokensBetween Holders is a multisender of tokens from contract. | Fail | Pending |
| BBT-12 | Major | Centralization Risks In The X Role | Pass | Not Found |
| BBT-13 | Informational | Extra Gas Cost For User.. | Fail | Pending |
| BBT-14 | Informational | Unnecessary Use Of SafeMath | Fail | Pending |

# BBT-01 | Potential Sandwich Attacks.

| Category | Severity | Location | Status |
|----------|----------|----------|--------|
| Security | 🟢 Minor | bbt2.sol: 1452,13 | 🗎 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:

  - swapExactTokensForETHSupportingFeeOnTransferTokens()
  - addLiquidityETH()

## Remediation

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

## Referrences:

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

# BBT-03 | Lack of Input Validation.

| Category | Severity | Location | Status |
|----------|----------|----------|--------|
| Volatile Code | 🟢 Minor | bbt2.sol: 69,9 | 🗎 Pending |

## Description

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

The given input is missing the check for the setExcludedFromFeesStatus,setCollection and setThresholdAmount 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");
    ...
```

We also recommend customer to review the following function that is missing a required validation. setExcludedFromFeesStatus,setCollection and setThresholdAmount is missing required function.

# BBT-05 | Missing Event Emission.

| Category | Severity | Location | Status |
|----------|----------|----------|--------|
| Volatile Code | 🟢 Major | bbt2.sol: 478, 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.

# BBT-10 | Initial Token Distribution.

| Category | Severity | Location | Status |
|---|---|---|---|
| Centralization / Privilege | 🟠 Major | bbt2.sol: 53,6 | Pending |

## Description

All of the Bunnies Battle Token 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

Token Distribution goes to the msg.sender.

# BBT-11 | distributeTokensBetweenHolders is a multisender of tokens from contract..

| Category | Severity | Location | Status |
|----------|----------|----------|--------|
| Security | 🟠 Critical | bbt2.sol: 103,14 | ⧗ Pending |

## Description

When combined with icollection if this is set to current contract address instead of nft contract can be used to send tokens to list instead of NFTs.

## Remediation

consider using another method to do a multi transfer.

## Project Action

# BBT-13 | Extra Gas Cost For User.

| Category | Severity | Location | Status |
|----------|----------|----------|--------|
| Logical Issue | ℹ️ Informational | bbt2.sol: 236, 8 | 🗎 Pending |

## Description

The user may trigger a tax distribution during the transfer process, which will cost a lot of gas and it is unfair to let
  a single user bear it.

## Remediation

We advise the client to make the owner responsible for the gas costs of the tax distribution.

## Project Action

# Social Media Checks

| Social Media | URL | Result |
|---|---|---|
| Twitter | https://twitter.com/BunniesGame | Pass |
| Other | | Fail |
| Website | https://bunniesbattle.io/ | Pass |
| Telegram | https://t.me/BunniesBattle | 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 | 76/100 |
| Auditor Score | 60/100 |
| Review by Section | Score |
| Manual Scan Score | 25/35 |
| SWC Scan Score | 35 /37 |
| Advance Check Score | 16 /28 |

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:

- Contract have a multi send function that when combined with other values can be used against holder.

- We recommend to remove the multi sender, to add requirement rules for the missing items.

- _balances(address.this) may not be functional and taxes not collected, i suggest the customer do another round of testing.

**Auditor Score =60**
**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 '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.