



# CFG NINJA AUDITS

Security Assessment

**Johnny Bravo Inu Token**

May 26, 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.
 Informational	Function Detected

## Manual Code Review Risk Results

Contract Priviledge	Description
 Buy Tax	8
 Sale Tax	8
 Cannot Sale	Pass
 Cannot Sale	Pass
 Max Tax	100
 Modify Tax	Up to 100%
 Fee Check	Fail
 Is Honeypot?	Yes, Transfer Failed
 Trading Cooldown	Not Detected
 Can Pause Trade?	Pass
 Pause Transfer?	Not Detected



Contract Priviledge	Description
 Max Tx?	Fail
 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	0xa9b126e4b67f06117559e88fc54bf01c134a80eb
 Self Destruct?	Not Detected
 Other?	Not Detected
 Other?	Not Detected
 Holders	1
 Auditor Confidence	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.



# Table of Contents

## **1 Assessment Summary**

## **2 Project Overview**

2.1 Token Summary

2.2 Risk Analysis Summary

2.3 Main Contract Assessed

## **3 Smart Contract Risk Checks**

3.1 Mint Check

3.2 Fees Check

3.3 Blacklist Check

3.4 MaxTx Check

3.5 Pause Trade Check

3.6 Contract Ownership

3.7 Liquidity Ownership

3.8 KYC Check

## **4 Smart Contract Vulnerability Checks**

4.1 Smart Contract Vulnerability Details

4.2 Smart Contract Inheritance Details

4.3 Smart Contract Privileged Functions

## **5 Technical Findings Details**

## **6 Social Media Check(Informational)**

## **7 Assessment Results and Notes(Important)**

7.1 Score Results

## **8 Disclaimer**



# Assessment Summary

This report has been prepared for Johnny Bravo Inu Token on the Binance Smart Chain 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.



# Project Overview

## Token Summary

Parameter	Result
Address	0xB93c5B81c1b8c2E8b78ad8Aca83012974347a790
Name	Johnny Bravo Inu
Token Tracker	Johnny Bravo Inu (BRAVO)
Decimals	9
Supply	100,000,000,000,000,000
Platform	Binance Smart Chain
compiler	v0.8.17+commit.8df45f5f
Contract Name	JOHNNYBRAVO
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	<a href="https://bscscan.com/address/0xB93c5B81c1b8c2E8b78ad8Aca83012974347a790#code">https://bscscan.com/address/0xB93c5B81c1b8c2E8b78ad8Aca83012974347a790#code</a>
Payment Tx	0x



# Project Overview

## Simulation Summary

Parameter	Result
Transfer From Owner	Fail
Transfer From Holder	Pass
Add Liquidity	Fail
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	
SwapAndSaleNoFee	Pass
SwapAndSale No/Fee TX	
ExcludeFromFees	Pass
LaunchPad	PinkSale



Parameter	Result
Pool Creation	Pass
Pool Creation TX	
Pool Finalize	Pass
Pool Finalize TX	
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
Johnny Bravo Inu	0xB93c5B81c1b8c2E8b78ad8Aca83012974347a790	Yes

## TestNet Contract Assessed Contract Name

Name	Contract	Live
Johnny Bravo Inu	0x1E54A5561FFD86F361290D5771BEa463ebe0F630	Yes

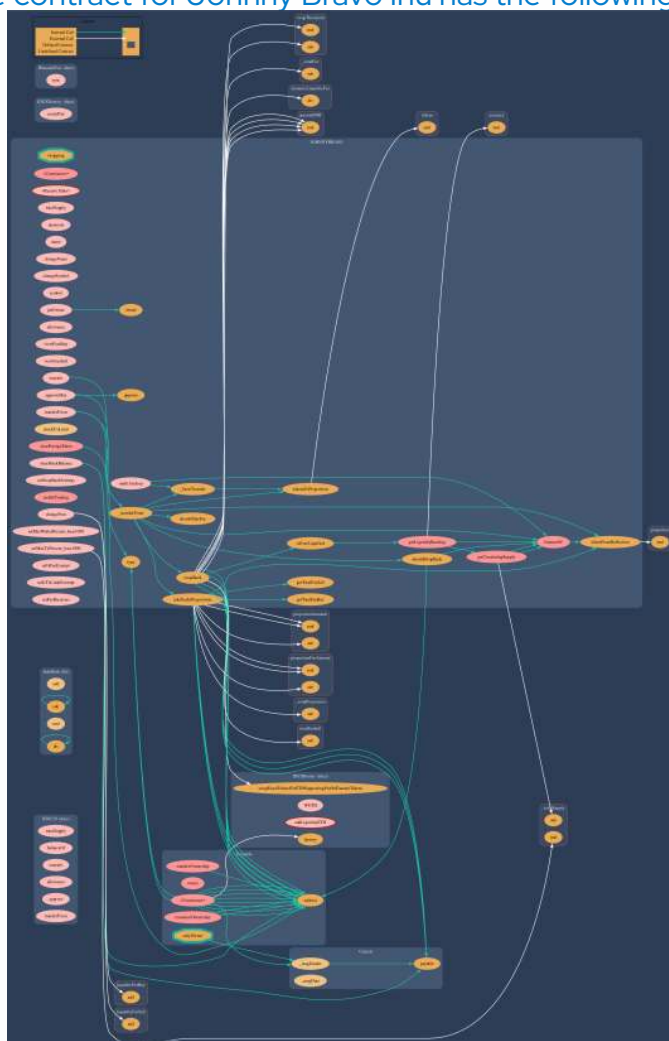
## Solidity Code Provided

SolID	File Sha-1	FileName
JBravo	e78bd2982e392d0b2d0a903ca3d87ea6c0eb794jbravo.sol c	
JBravo		
JBravo		
JBravo		



# Call Graph

The contract for Johnny Bravo Inu has the following call graph structure.



# KYC Information

**The Project Owners of Johnny Bravo Inu is not KYC.**

KYC Information Notes:

Auditor Notes: KYC to be completed by PinkSale, project will be a SAFU Project.

Project Owner Notes:



# 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	jbravo.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	jbravo.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	jbravo.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	jbravo.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	jbravo.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	jbravo.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	jbravo.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	jbravo.sol	L: 0 C: 0
SWC-108	Low	State variable visibility is not set..	jbravo.sol	L: 220 C: 11
SWC-109	Pass	Uninitialized Storage Pointer.	jbravo.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	jbravo.sol	L: 0 C: 0



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



ID	Severity	Name	File	location
SWC-126	Pass	Insufficient Gas Griefing.	jbravo.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	jbravo.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	jbravo.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	jbravo.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	jbravo.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	jbravo.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	jbravo.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	jbravo.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	jbravo.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	jbravo.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	jbravo.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-108 - State Variable Default Visibility

### CWE-710: Improper Adherence to Coding Standards

#### Description:

Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

#### Remediation:

Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

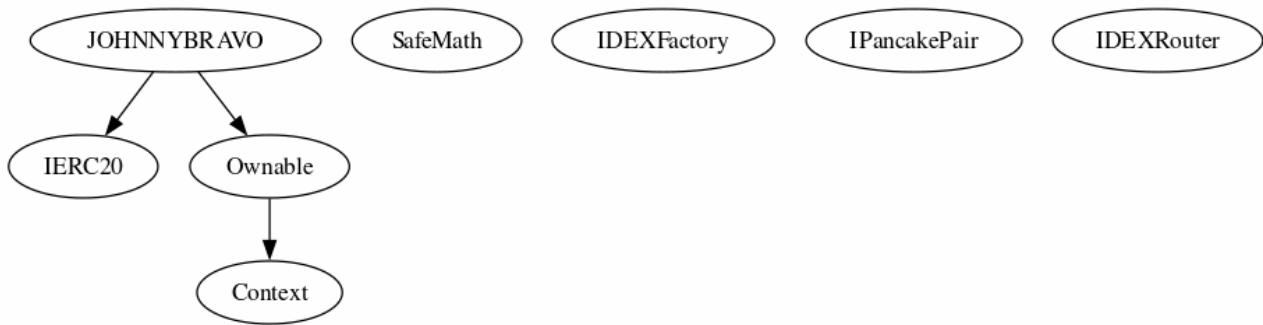
#### References:

Ethereum Smart Contract Best Practices - Explicitly mark visibility in functions and state variables



# Inheritance

The contract for Johnny Bravo 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
multiAirdrop		External
setFeeReceivers		External
setIsTxLimitExempt		External
setIsFeeExempt		External
setMaxTxPercent_base1000		External
setMaxWalletPercent_base1000		External
changeFees		External
claimStuckTokens		External
enableTrading		public
setSwapBackSettings		public



# Smart Contract Advance Checks



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



ID	Severity	Name	Result	Status
BRAVO-16	Medium	Invalid collection of Taxes during Transfer.	Pass	Not-Found
BRAVO-17	Informational	Conformance to numeric notation best practice.	Pass	Not-Found
BRAVO-18	Informational	Enable Trade and Exclude Exist to create a whitelist.	Pass	Not-found



## BRAVO-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	 Minor	jbravo.sol: L: 220 C: 11	 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
_name		internal
_symbol		internal
_totalSupply		internal
_allowances		internal
isFeeExempt		internal
isTxLimitExempt		internal
liquidityFeeBuy		internal
teamFeeBuy		internal
devFeeBuy		internal
marketingFeeBuy		internal
largeSwapThreshold		internal

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.



## BRAVO-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	 Minor	jbravo.sol: 422,14	 Pending

### Description

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

The given input is missing the check for the all onlyOwners 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. all onlyOwners are missing required function.



# BRAVO-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	 Minor	jbravo.sol: 422, 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.



## BRAVO-11 | multiAirdrop present in code..

Category	Severity	Location	Status
Optimization	 Major	jbravo.sol: 571,14	 Pending

### Description

multiAirdrop present in code, this means it can be used to send to multiple address inside code.

### Remediation



remove the multiAirdrop from code is ideal to keep it clean.

### Project Action





## BRAVO-14 | Unnecessary Use Of SafeMath

Category	Severity	Location	Status
Logical Issue	 Medium	jbravo.sol: 79,9	 Pending

### 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	3	0	0
 Informational	1	0	0
Total	5	0	0



# Social Media Checks

Social Media	URL	Result
Twitter	<a href="https://twitter.com/johnnybravoinu">https://twitter.com/johnnybravoinu</a>	Pass
Other		Fail
Website	<a href="https://www.johnnybravoinu.com">https://www.johnnybravoinu.com</a>	Pass
Telegram	<a href="https://t.me/johnnybravogroup">https://t.me/johnnybravogroup</a>	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:**



# Audit Result

## Final Audit Score

Review	Score
Security Score	65
Auditor Score	70

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 Fail



## Assessment Results

### Important Notes:

- No issues or vulnerabilities were found.
- The Contract needs improvements.
- Please DYOR on the project.
- It has a change name and change symbol in code.
- failed to add lp.
- TransferHelper: TRANSFER\_FROM\_FAILED, code need to be reviewed.

**Auditor Score =70**  
**Audit Fail**



# 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.

