



Security Assessment: BlockBets Casino Token







April 4, 2024

- Audit Status: **Pass**
- Audit Edition: **Standard**
































Risk Analysis

Classifications of Manual Risk Results

Classification	Description
 Critical	Danger or Potential Problems.
 High	Be Careful or Fail test.
 Low	Pass, Not-Detected or Safe Item.
 Informational	Function Detected

Manual Code Review Risk Results

Contract Privilege	Description
 Buy Tax	0%
 Sale Tax	0%
 Cannot Buy	Pass
 Cannot Sale	Pass
 Max Tax	0%
 Modify Tax	No
 Fee Check	Pass
 Is Honeypot?	Not Detected
 Trading Cooldown	Not Detected
 Can Pause Trade?	Pass
 Pause Transfer?	Not Detected
 Max Tx?	Pass
 Is Anti Whale?	Not Detected
 Is Anti Bot?	Not Detected

Contract Privilege	Description
 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	0xf5aC3E38aC4a596AA3bbB8934fF84526Db5Ce63a
 Self Destruct?	Not Detected
 External Call?	Not Detected
 Other?	Not Detected
 Holders	1
 Auditor Confidence	Low to Medium Risk
 KYC Present	No
 KYC URL	

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	BlockBets Casino
Name	BlockBets Casino
Token Tracker	BlockBets Casino (\$BBETS)
Decimals	18
Supply	100,000,000
Platform	BNBCHAIN
compiler	v0.8.18+commit.87f61d96
Contract Name	BBETSToken
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/token/0x45183049F5dfB676aba57fdDE8dd37E794781D9A#code
Payment Tx	Corporate

Main Contract Assessed Contract Name

Name	Contract	Live
BlockBets Casino	BlockBets Casino	Yes

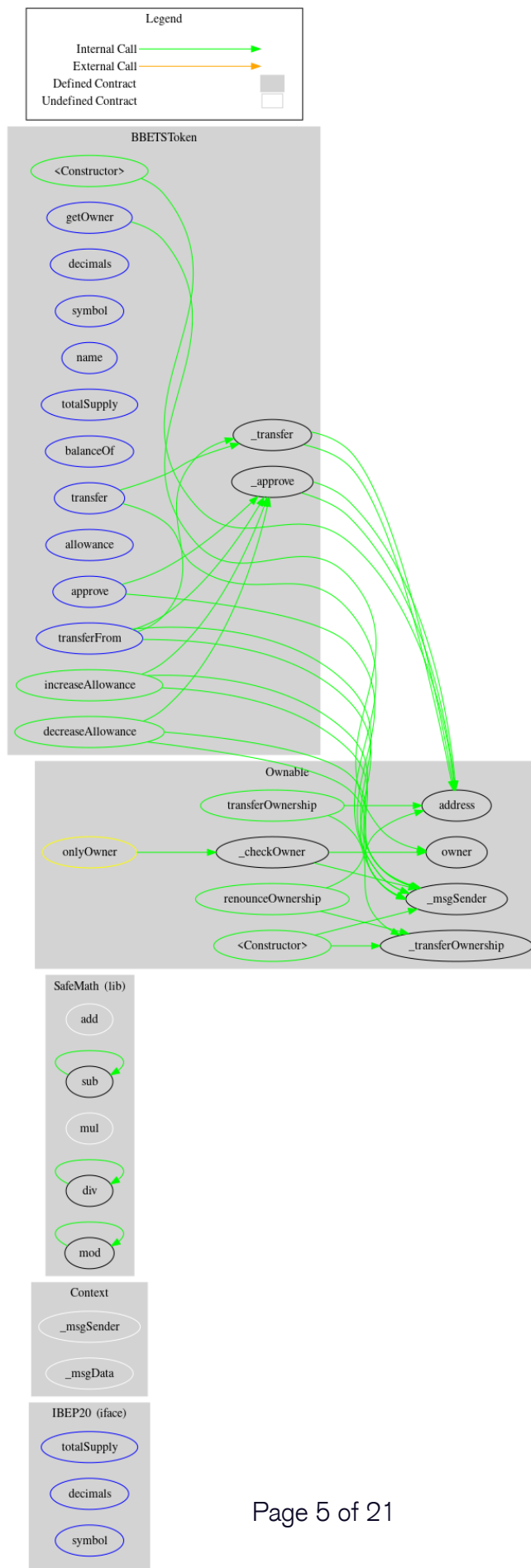
TestNet Contract was Not Assessed

Solidity Code Provided

SolidID	File Sha-1	FileName
BBETS	06ab3099c64783a7d3bacc6471809cb6fd2ed8be	contract.sol
BBETS		
BBETS		
BBETS		
BBETS		
BBETS	undefined	

Call Graph

The contract for BlockBets Casino has the following call graph structure.



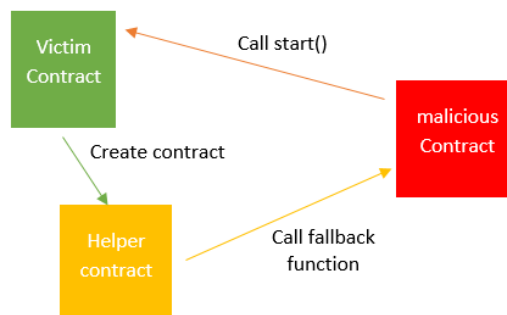
Reentrancy Check

The Project Owners of BlockBets Casino have not configure the Reentrancy Guard library.

**You can read more about Reentrancy issues in the following link.
[Reentrancy After Istanbul.](#)**

We recommend the team to add the library to the contract to avoid potential issues.

We recommend the team to create a new contract with Reentrancy Guard added to the same.



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	contract.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	contract.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	contract.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	contract.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	contract.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	contract.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	contract.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	contract.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	contract.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	contract.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	contract.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	contract.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	contract.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	contract.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	contract.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-115	Pass	Authorization through tx.origin.	contract.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	contract.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	contract.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	contract.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	contract.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randomness.	contract.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	contract.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	contract.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	contract.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	contract.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	contract.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	contract.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	contract.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	contract.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	contract.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	contract.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	contract.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	contract.sol	L: 0 C: 0

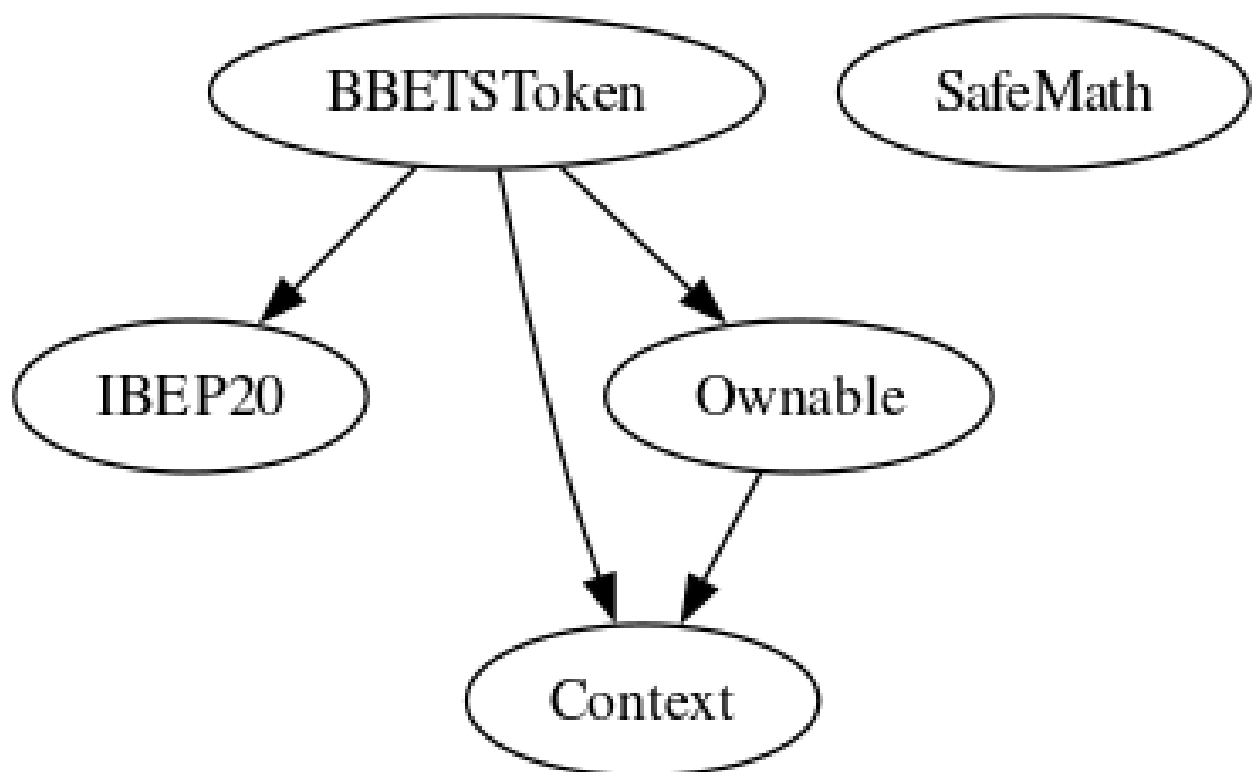
ID	Severity	Name	File	location
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	contract.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	contract.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	contract.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	contract.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 BlockBets Casino has the following inheritance structure.

The Project has a Total Supply of 100,000,000



Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

Function Name	Parameters	Visibility
transferOwnership,address newOwner,Public	renounceOwnership,,Public	

\$BBETS-10 | Initial Token Distribution.

Category	Severity	Location	Status
Centralization / Privilege	 High	contract.sol: L: 376 C: 14	 Detected

Description

All of the BlockBets Casino 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

```
_totalSupply = 100_000_000 * 10**uint256(_decimals);
```

\$BBETS-14 | Unnecessary Use Of SafeMath

Category	Severity	Location	Status
Logical Issue	 Medium	contract.sol: L: 135 C: 9	 Detected

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.
 High	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
 Low	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
 High	1	1	0
 Medium	1	1	0
 Low	0	0	0
 Informational	0	0	0
Total	2	2	0

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/BlockBetsCasi	Pass
Other	https://discord.gg/vyb5YxRGKu	Pass
Website	https://presale.blockbets.casino/	Pass
Telegram	N/A	No

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	86/100
Auditor Score	85/100
Review by Section	Score
Manual Scan Score	18
SWC Scan Score	37
Advance Check Score	31

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 84 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below.

Audit Passed



Assessment Results

Important Notes:

- Reentrancy: Not a direct risk for the current functions, but should be considered for future updates.␣
- Centralization: Owner has significant control; potential risk if compromised.␣
- Approve/Allowance: Race condition in approve could be exploited; recommend using increaseAllowance and decreaseAllowance.␣
- Input Validation: Missing checks for transfers to the contract address.␣
- Burn Mechanism: Absent, limiting token economics flexibility.␣
- Circuit Breaker: No emergency stop functionality.␣
- Gas Optimization: SafeMath is unnecessary for Solidity 0.8.x.␣
- Ownership Events: Missing event emission on ownership renouncement.␣
- Timelocks: No delay on sensitive ownership transfer operations.␣
- Multisig Requirement: Not enforced, single owner decision-making.␣
- Function Visibility: Not explicitly stated, could lead to misuse.␣
- Custom Errors: Not used, less gas-efficient than revert strings.␣

- Deflationary Mechanisms: None present.␣
- Oversized Control: Owner has substantial power over the contract.␣
- Overflow and Underflow: Mitigated by SafeMath, but redundant due to Solidity 0.8.x's built-in checks.␣
- Token Recovery: No mechanism to recover tokens sent to the contract by mistake.␣
- Balance Freezing: No functionality for freezing/unfreezing balances, which affects the ability to respond to security incidents or comply with regulations.␣
- Approval Race Condition: The approve function is susceptible to a known race condition, potentially allowing a spender to exceed the intended allowance.␣
- Overall Summary: The BBETSToken contract's balance management adheres to the standard BEP20 protocol, with arithmetic operations secured by SafeMath. Despite this, the contract has several areas for improvement, including the lack of transfer validation to the contract address, absence of token recovery mechanisms, and the potential for an approval race condition. These issues could lead to token loss or unauthorized token spending.␣
- Classification of Risk: Medium Risk. The balance management system is robust, but the contract's susceptibility to the approval race condition and the potential for token loss due to a lack of validation and recovery mechanisms elevate the overall risk level. It is recommended to address these concerns to ensure a higher degree of security for token

holders.

Auditor Score =85
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

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

