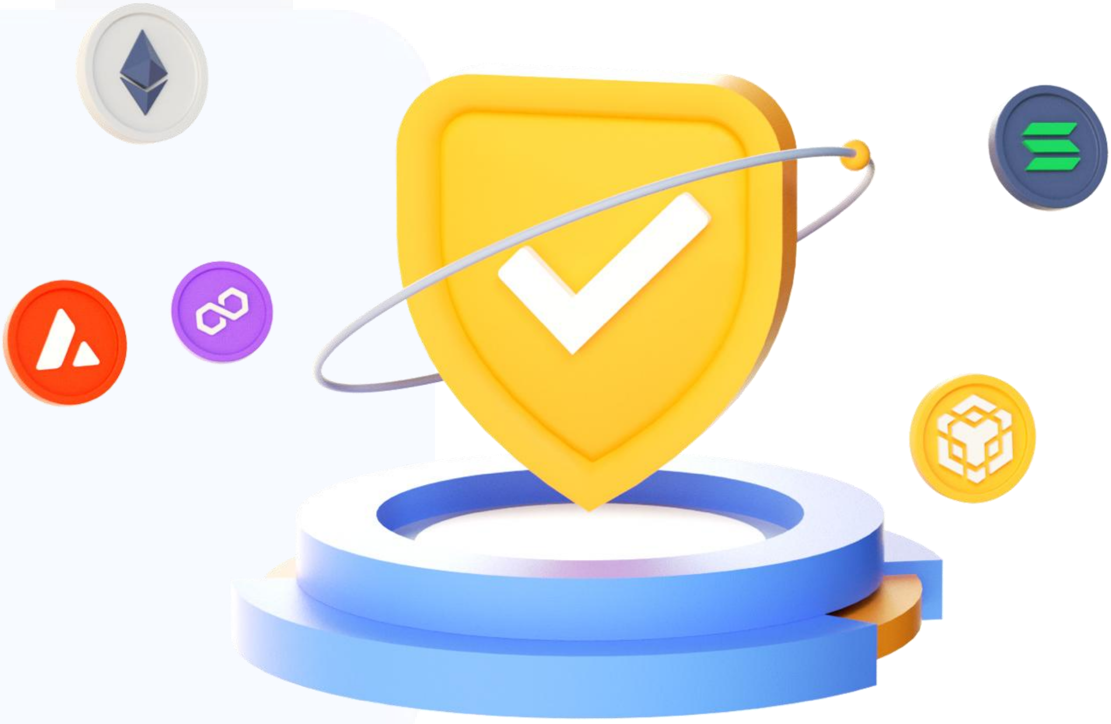
coinsurance

# Smart Contract Security Assessment

# WiFi Cash

Dec 9th, 2022



This report may contain confidential knowledge about the intellectual property of the applicant as well as the analysis on the vulnerabilities of the smart contract. Such information can be used internally by the customer to make adjustments and it can be published after all critical vulnerabilities are amended by the decision of the applicant.

| | |
|---|---|
| **Title** | Analysis Report on the Smart Contract of WiFi Cash |
| **Conducted By** | Michael Holland |
| **Ecosystem** | BEP20 |
| **Platform** | BNB Chain / Solidity |
| **Language** | Solidity |
| **Analytic Methods** | Functional Testing, Computer-Aided architecture analysis, manual review |
| **Date of Completion** | 2022-12-5 |
| **Initial Review** | 2022-12-9 |

## List of contents

## Summary

Coinsurance was requested by WiFi Cash (The “Applicant”) to perform a security analysis of its smart contract code. This report shows the findings of the security analysis.

## Scope

The scope of the security analysis covers the smart contract in the repository:

Deployed Contract:

https://bscscan.com/address/0x43a2581adcc9baf660066bc9e3cf71e208cc8bf6

Coinsurance has reviewed this smart contract for both common and specific vulnerabilities. Here is the list of the vulnerabilities that we vetted:

| Category | Checked Item |
|---|---|
| Code Analysis | ▪ Reentrancy<br>▪ Contract Ownership<br>▪ Timestamp<br>▪ Gas Limit<br>▪ Transaction-ordering violation<br>▪ EIP violation<br>▪ Unchecked external call<br>▪ Unchecked algorithm<br>▪ Unsafe type inference<br>▪ Implicit visibility<br>▪ Deployment Consistency<br>▪ Repository Consistency |
| Functional Analysis | ▪ Business Logics<br>▪ Functionality<br>▪ Access Control & Authorization<br>▪ Token Supply manipulation<br>▪ Assets integrity<br>▪ User Balances manipulation<br>▪ Data Consistency<br>▪ Kill-Switch |

## Contract Scoring

Score measurement is detailed in the methodology section.

## Documentation score

The applicant submitted no functional or technical requirements to the contract. However, the description of the constructor matches the code. The total documentation score is 10 out of 10.

## Code score

The code score is 10 out of 10. No unit tests were provided. The NatSpec blocks are not comprehensive with some hardcoded values.

## Architecture score

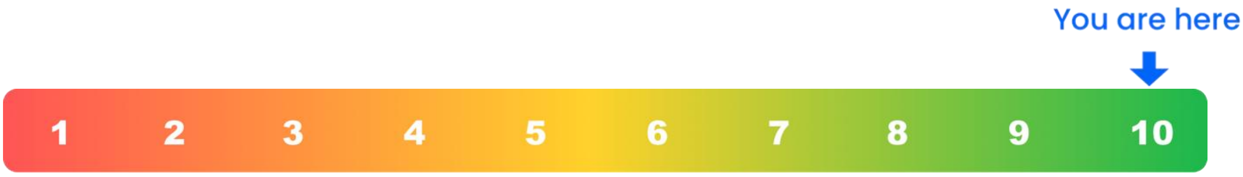
The architecture score is 10 out of 10. The contract has a clear architecture.

## Security score

After Thorough Analysis, our security staff found 2 low-level issue. The security score is 10 out of 10.

## Total score

By the generalization of each aspect of the security analysis, the applicant’s smart contract receives the total score of 10/10



## Severity Classification

| Risk Level | Definition |
|---|---|
| Critical | Critical vulnerabilities can be easily exploited for data manipulations and can lead to total loss of investors' assets. |
| Major | Major vulnerabilities impact how a smart contract is executed. |
| Medium-level | Medium-level vulnerabilities can also be concerning, but they do not lead to asset loss or data manipulation. |
| Low-level | Low-level vulnerabilities are typically a result of the outdated code snippets and do not entail significant risks. |

## Findings

**Critical Vulnerabilities – None**

**High-level Vulnerabilities – None**

**Medium-level Vulnerabilities – None**

**Low-level Vulnerabilities – 2**

1. Minted token missing event record.

Token minting for the first time might not show event record, the tracking of the minted token might malfunction.

Function: _mint

Suggestion: Add emit Transfer(address(0), account, amount);

Status: Resolved

2. Approved spender address not checked, might result in 0x0 address.

During spender authorization, there is no spender address to be checked which could result in a 0x0 address and invalid authorization.

Function: approve

Suggestion: Add spender address verification: require(spender != address(0), "ERC20: approve to the zero address");

Status: Resolved

# Disclaimers

## Coinsurance Disclaimer

The smart contract provided by the applicant has been audited based on an institutional level of rigidity and prudence. All contract vulnerabilities disclosed by this security analysis are concluded from the contract's source code compilation, deployment and functionality.

The audit makes no absolute warranties on the security of the project or the code. Be aware that this security analysis only audits the smart contract of the project, however, the interest of the investors could also be compromised by the illegitimate business practices as well as the misleading marketing campaigns. As an investor, you should not solely rely on the conclusion of this report to judge the project's legitimacy.

## Technical Disclaimer

Smart contracts are deployed on blockchains. The blockchain platform can be vulnerable to hacks and cyber-attacks. Therefore, our security analysis does not guarantee the external security of the project audited.