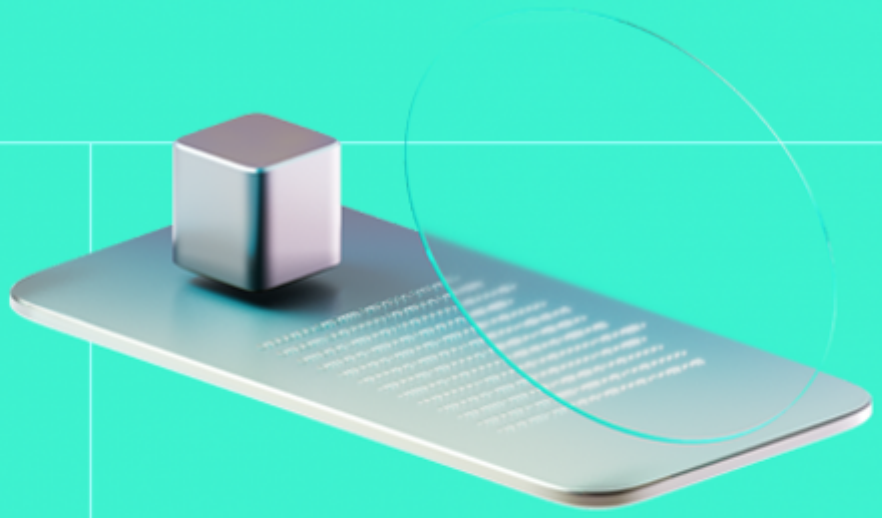
HACKEN

Smart Contract Code Review And Security Analysis Report



Customer: Bixos

Date: 27/02/2024

We express our gratitude to the Bixos team for the collaborative engagement that enabled the execution of this Smart Contract Security Assessment.

Bixos is a simple BEP-20 token, which mints all of the total supply to the deployer.

Platform: EVM

Language: Solidity

Tags: BEP-20

Timeline: 26/02/2024 - 27/02/2024

Methodology: https://hackenio.cc/sc_methodology

Review Scope

Repository	https://github.com/bixos/ubxstoken-smartcontract
Commit	f5f5329

Audit Summary

10/10	8/10	100%	10/10
Security Score	Code quality score	Test coverage	Documentation quality score

Total 9.6/10

The system users should acknowledge all the risks summed up in the risks section of the report

0	0	0	0
Total Findings	Resolved	Accepted	Mitigated

Findings by severity

Critical	0
High	0
Medium	0
Low	0

This report may contain confidential information about IT systems and the intellectual property of the Customer, as well as information about potential vulnerabilities and methods of their exploitation.

The report can be disclosed publicly after prior consent by another Party. Any subsequent publication of this report shall be without mandatory consent.

Document

Name	Smart Contract Code Review and Security Analysis Report for Bixos
Audited By	Giovanni Franchi
Approved By	Yves Toiser
Website	https://bixos.com/
Changelog	26/02/2024 - Preliminary Report

Table of Contents

System Overview

Bixos is a simple BEP-20 with the following contract:

1_UbxsToken — BEP-20 token that mints all initial supply to a deployer. Additional minting is not allowed. Total supply is not defined in a constant but passed as a constructor argument.

It has the following attributes:

- Name: UBXS Token
- Symbol: UBXS
- Decimals: 6
- Total supply: Defined at deploy time.

Privileged roles

- No privileged roles detected.

Executive Summary

This report presents an in-depth analysis and scoring of the customer's smart contract project. Detailed scoring criteria can be referenced in the scoring methodology.

Documentation quality

The total Documentation Quality score is **10** out of **10**.

- Functional requirements are provided.
- Technical description is provided.

Code quality

The total Code Quality score is **8** out of **10**.

- Naming convention best practises are violated.
- The development environment is configured.

Test coverage

Code coverage of the project is **100%**.

- Test coverage is not required for projects below 250 loc.

Security score

Upon auditing, the code was found to contain **0** critical, **0** high, **0** medium, and **0** low severity issues, leading to a security score of **10** out of **10**.

All identified issues are detailed in the “Findings” section of this report.

Summary

The comprehensive audit of the customer's smart contract yields an overall score of **9.6**. This score reflects the combined evaluation of documentation, code quality, test coverage, and security aspects of the project.

Risks

- The total supply of the token is determined during the deployment. It cannot be verified until the contract is deployed.
- All tokens are minted to a single address. The secureness of the supply depends on the secureness of key storage.
- The whitepaper mentions many functionalities that are not implemented in the present scope such as an allocation program and a tokenomics strategy.

Findings

Vulnerability Details

Observation Details

F-2024-1044 - Non-Compliance with Naming Conventions for Contract and File Names - Info

Description:

Solidity best practices suggest that contract names should match their file names and both should be in PascalCase. PascalCase (or Upper Camel Case) is a naming convention where each word begins with a capital letter without spaces. For example, **UbxsToken**.

The observed inconsistency and deviation from PascalCase in the contract codebase can lead to confusion and reduce code readability and maintainability.

Assets:

- contracts/1_UbxsToken.sol [https://github.com/bixos/ubxstoken-smartcontract/blob/master/contracts/1_UbxsToken.sol]

Status: Pending Fix

Recommendations

Recommendation:

It is suggested to modify the file name accordingly to the contract's name the file is hosting.

- current format: `1_UbxsToken.sol`
- suggested format: `UbxsToken.sol`

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed based on best industry practices at the time of the writing of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The report contains no statements or warranties on the identification of all vulnerabilities and security of the code. The report covers the code submitted and reviewed, so it may not be relevant after any modifications. Do not consider this report as a final and sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements.

While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

English is the original language of the report. The Consultant is not responsible for the correctness of the translated versions.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the Consultant cannot guarantee the explicit security of the audited smart contracts.

Appendix 1. Severity Definitions

When auditing smart contracts, Hacken is using a risk-based approach that considers **Likelihood**, **Impact**, **Exploitability** and **Complexity** metrics to evaluate findings and score severities.

Reference on how risk scoring is done is available through the repository in our Github organization:

hknio/severity-formula

Severity	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to the loss of user funds or contract state manipulation.
High	High vulnerabilities are usually harder to exploit, requiring specific conditions, or have a more limited scope, but can still lead to the loss of user funds or contract state manipulation.
Medium	Medium vulnerabilities are usually limited to state manipulations and, in most cases, cannot lead to asset loss. Contradictions and requirements violations. Major deviations from best practices are also in this category.
Low	Major deviations from best practices or major Gas inefficiency. These issues will not have a significant impact on code execution, do not affect security score but can affect code quality score.

Appendix 2. Scope

The scope of the project includes the following smart contracts from the provided repository:

Scope Details

Repository	https://github.com/bixos/ubxstoken-smartcontract
Commit	f5f5329e4243c18395dccff28bbea3b358b4424b
Whitepaper	https://docs.bixos.io/documents/readme
Requirements	https://docs.bixos.io/documents/readme
Technical Requirements	https://docs.bixos.io/documents/readme

Contracts in Scope

./contracts/1_UbxsToken.sol