QuillAudits

Audit Report
September, 2024

For

bako safe

Table of Content

Executive Summary

Project Name	Bako Safe
Project URL	https://www.bako.global/
Overview	Bako Safe is a multisig wallet which can be used to transfer coins, tokens on fuel blockchain
Audit Scope	https://github.com/Bako-Labs/bako-safe/tree/main/packages/sway
Commit Hash	333bd5035ae12af6b1168b4e6a73de72f4e3a4a4
Language	Sway
Blockchain	Fuel
Method	Manual Analysis, Functional Testing, Automated Testing
First Review	12th September 2024 - 25th September 2024
Updated Code Received	24th September 2024
Second Review	25th September 2024
Fixed In	https://github.com/Bako-Labs/bako-safe/commits/main/

Number of Security Issues per Severity

2
Issues Found

🟥 High 🟨 Medium

🟩 Low 🟪 Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	0	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	0	0	2

Checked Vulnerabilities

- ✓ Arbitrary write to storage
- ✓ Centralization of control
- ✓ Ether theft
- ✓ Improper or missing events
- ✓ Logical issues and flaws
- ✓ Arithmetic Computations Correctness
- ✓ Race conditions/front running
- ✓ Re-entrancy
- ✓ Malicious libraries
- ✓ Address hardcoded
- ✓ Divide before multiply
- ✓ Integer overflow/underflow
- ✓ ERC's conformance
- ✓ Missing Zero Address Validation
- ✓ Revert/require functions
- ✓ Upgradeable safety
- ✓ Using inline assembly
- ✓ Style guide violation
- ✓ Parallel Execution safety
- ✓ UTXO Model Verification
- ✓ FuelVM Opcodes
- ✓ Cross-Chain Interactions
- ✓ Modular Design
- ✓ Access Control Vulnerabilities
- ✓ Denial of Service (DoS)
- ✓ Oracle Manipulation
- ✓ Signature Replay Attacks
- ✓ Improper Handling of External Calls
- ✓ Proxy Storage Collision
- ✓ Use of Deprecated Functions

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Foundry, Solhint, Mythril, Slither, Solidity statistic analysis.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

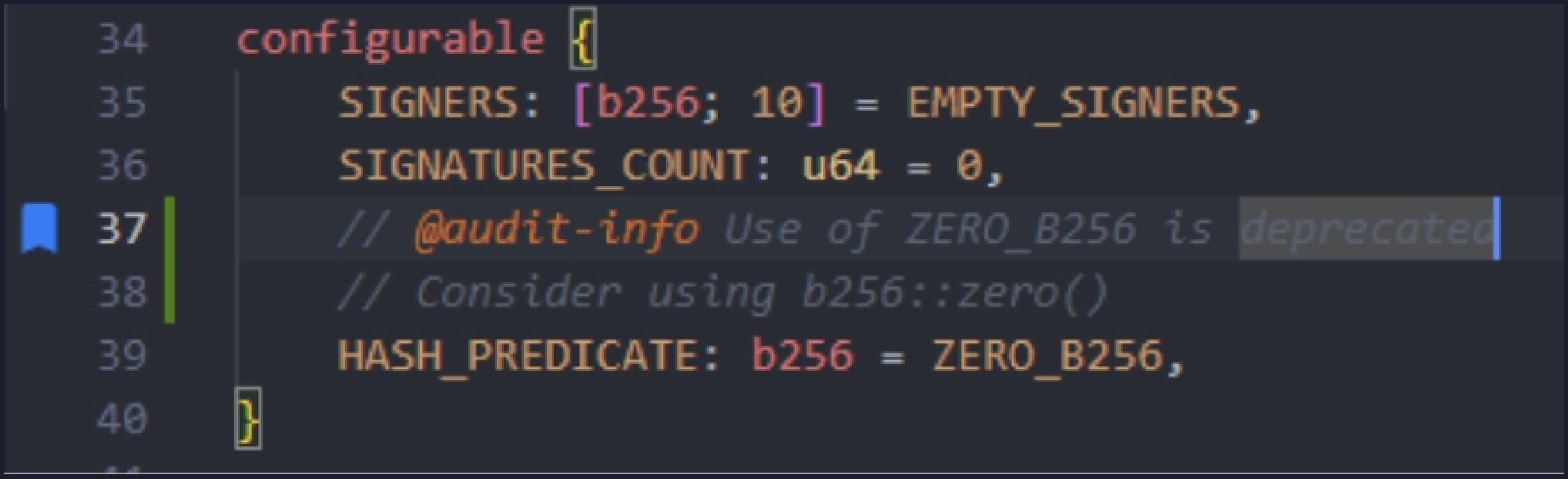
Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

Informational Issues

1. Use of ZERO_B256 is deprecated

Path

configurables


```
configurable {
    SIGNERS: [b256; 10] = EMPTY_SIGNERS,
    SIGNATURES_COUNT: u64 = 0,
    // @audit-info Use of ZERO_B256 is deprecated
    // Consider using b256::zero()
    HASH_PREDICATE: b256 = ZERO_B256,
}
```

Description

Use of the ZERO_B256 is deprecated as per the updated sway language standards

Recommendation

It is advised to use b256::zero()

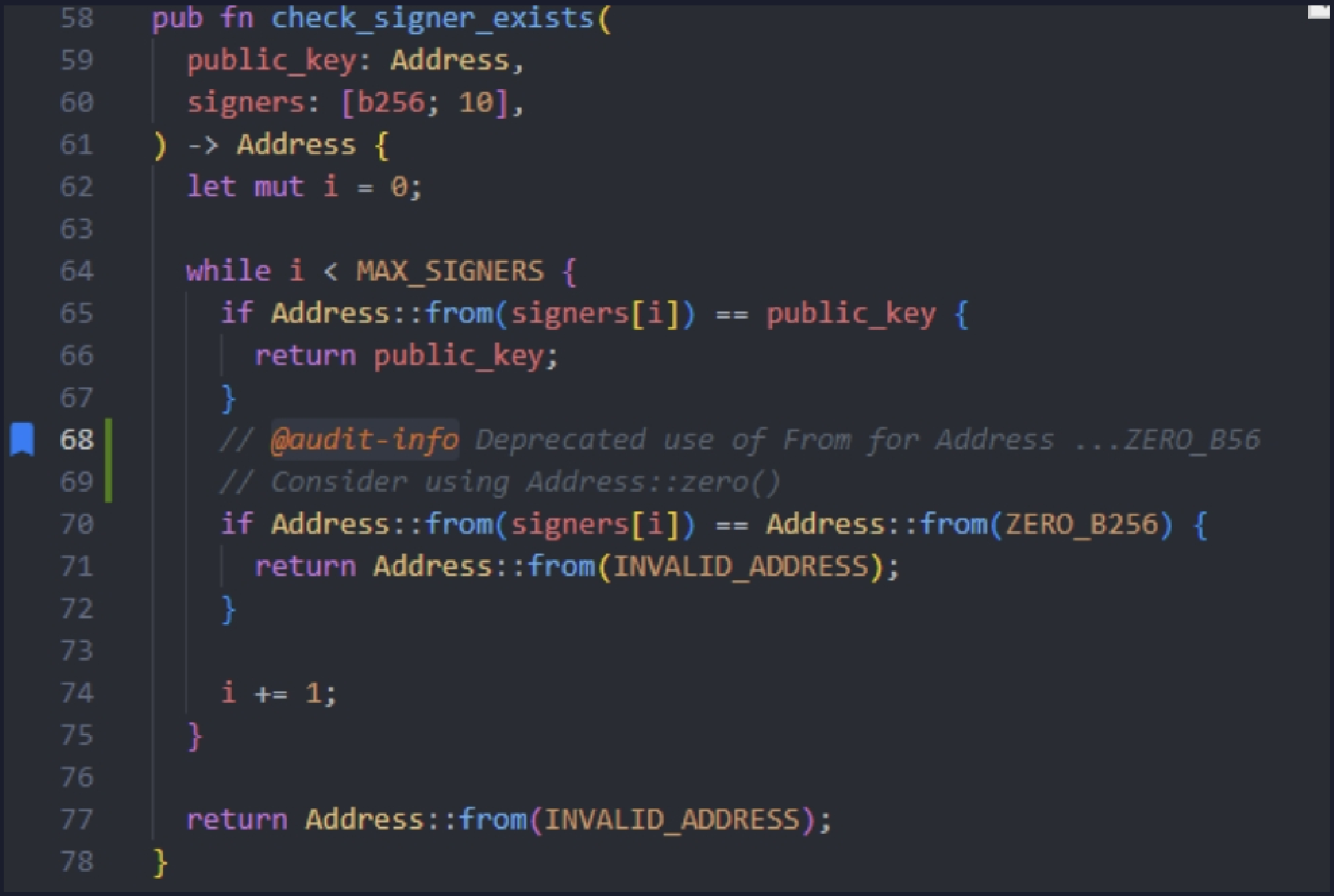
Status

Resolved

2. Use of from for Address zero is deprecated

Path

validations.sw



```
pub fn check_signer_exists(
  public_key: Address,
  signers: [b256; 10],
) -> Address {
  let mut i = 0;

  while i < MAX_SIGNERS {
    if Address::from(signers[i]) == public_key {
      return public_key;
    }
    // @audit-info Deprecated use of From for Address ...ZERO_B56
    // Consider using Address::zero()
    if Address::from(signers[i]) == Address::from(ZERO_B256) {
      return Address::from(INVALID_ADDRESS);
    }

    i += 1;
  }

  return Address::from(INVALID_ADDRESS);
}
```

Description

Use of the Address::from(ZERO_B256) is deprecated as per the updated sway language standards

Recommendation

It is advised to use Address::zero()

Status

Resolved

Functional Tests Cases

Some of the tests performed are mentioned below:

- ✓ Check if same address is used for bako safe multisig for say ¾
- ✓ Check if zero address can be used for signing
- ✓ Check if another address can be used for signing

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of Bako Safe. We performed our audit according to the procedure described above.

Some issues of informational severity were found,which the Bako Safe Team has Fixed.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in Bako Safe. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of Bako Safe. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of Bako Safe to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

About QuillAudits

QuillAudits is a leading name in Web3 security, offering top-notch solutions to safeguard projects across DeFi, GameFi, NFT gaming, and all blockchain layers. With six years of expertise, we've secured over 1000 projects globally, averting over $30 billion in losses. Our specialists rigorously audit smart contracts and ensure DApp safety on major platforms like Ethereum, BSC, Arbitrum, Algorand, Tron, Polygon, Polkadot, Fantom, NEAR, Solana, and others, guaranteeing your project's security with cutting-edge practices.

1000+
Audits Completed

$30B
Secured

1M+
Lines of Code Audited

Follow Our Journey

Audit Report
September, 2024

For

bako safe

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com