



# Cyberscope

## Audit Report

# GUFO

June 2023

Network    BSC

Address    0x2699Be6153b70F4dB08bE89ac309361b5E3a0437

Audited by    © cyberscope

# Analysis

● Critical ● Medium ● Minor / Informative ● Pass

| Severity | Code | Description             | Status |
|----------|------|-------------------------|--------|
| ●        | ST   | Stops Transactions      | Passed |
| ●        | OTUT | Transfers User's Tokens | Passed |
| ●        | ELFM | Exceeds Fees Limit      | Passed |
| ●        | MT   | Mints Tokens            | Passed |
| ●        | BT   | Burns Tokens            | Passed |
| ●        | BC   | Blacklists Addresses    | Passed |

# Diagnostics

● Critical ● Medium ● Minor / Informative

| Severity | Code | Description                                | Status     |
|----------|------|--------------------------------------------|------------|
| ●        | AOI  | Arithmetic Operations Inconsistency        | Unresolved |
| ●        | RSML | Redundant SafeMath Library                 | Unresolved |
| ●        | L04  | Conformance to Solidity Naming Conventions | Unresolved |
| ●        | L19  | Stable Compiler Version                    | Unresolved |

# Table of Contents

|                                                  |           |
|--------------------------------------------------|-----------|
| <b>Analysis</b>                                  | <b>1</b>  |
| <b>Diagnostics</b>                               | <b>2</b>  |
| <b>Table of Contents</b>                         | <b>3</b>  |
| <b>Review</b>                                    | <b>4</b>  |
| Audit Updates                                    | 4         |
| Source Files                                     | 4         |
| <b>Findings Breakdown</b>                        | <b>5</b>  |
| AOI - Arithmetic Operations Inconsistency        | 6         |
| Description                                      | 6         |
| Recommendation                                   | 6         |
| RSMML - Redundant SafeMath Library               | 7         |
| Description                                      | 7         |
| Recommendation                                   | 7         |
| L04 - Conformance to Solidity Naming Conventions | 8         |
| Description                                      | 8         |
| Recommendation                                   | 8         |
| L19 - Stable Compiler Version                    | 9         |
| Description                                      | 9         |
| Recommendation                                   | 9         |
| <b>Functions Analysis</b>                        | <b>10</b> |
| <b>Flow Graph</b>                                | <b>11</b> |
| <b>Summary</b>                                   | <b>12</b> |
| <b>Disclaimer</b>                                | <b>13</b> |
| <b>About Cyberscope</b>                          | <b>14</b> |

## Review

|                  |                                                                                                                                                             |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Contract Name    | GUFO                                                                                                                                                        |
| Compiler Version | v0.8.9+commit.e5eed63a                                                                                                                                      |
| Optimization     | 200 runs                                                                                                                                                    |
| Explorer         | <a href="https://bscscan.com/address/0x2699be6153b70f4db08be89ac309361b5e3a0437">https://bscscan.com/address/0x2699be6153b70f4db08be89ac309361b5e3a0437</a> |
| Address          | 0x2699be6153b70f4db08be89ac309361b5e3a0437                                                                                                                  |
| Network          | BSC                                                                                                                                                         |
| Symbol           | GUFO                                                                                                                                                        |
| Decimals         | 9                                                                                                                                                           |
| Total Supply     | 1,000,000,000,000                                                                                                                                           |

## Audit Updates

|               |             |
|---------------|-------------|
| Initial Audit | 11 Jun 2023 |
|---------------|-------------|

## Source Files

|          |                                                                  |
|----------|------------------------------------------------------------------|
| Filename | SHA256                                                           |
| GUFO.sol | 6e4b5315aa166f8192804e814312cf6a460a834a1d6f346e5223fd7494f6b702 |

## Findings Breakdown



|                       |   |
|-----------------------|---|
| ● Critical            | 0 |
| ● Medium              | 0 |
| ● Minor / Informative | 4 |

| Severity              | Unresolved | Acknowledged | Resolved | Other |
|-----------------------|------------|--------------|----------|-------|
| ● Critical            | 0          | 0            | 0        | 0     |
| ● Medium              | 0          | 0            | 0        | 0     |
| ● Minor / Informative | 4          | 0            | 0        | 0     |

## AOI - Arithmetic Operations Inconsistency

|             |                     |
|-------------|---------------------|
| Criticality | Minor / Informative |
| Status      | Unresolved          |

### Description

The contract uses both the SafeMath library and native arithmetic operations. The SafeMath library is commonly used to mitigate vulnerabilities related to integer overflow and underflow issues. However, it was observed that the contract also employs native arithmetic operators (such as +, -, \*, /) in certain sections of the code.

The combination of SafeMath library and native arithmetic operations can introduce inconsistencies and undermine the intended safety measures. This discrepancy creates an inconsistency in the contract's arithmetic operations, increasing the risk of unintended consequences such as inconsistency in error handling, or unexpected behavior.

```
balances[owner] = balances[owner].sub(numTokens);  
...  
balances[msg.sender] -= _amount;
```

### Recommendation

To address this finding and ensure consistency in arithmetic operations, it is recommended to standardize the usage of arithmetic operations throughout the contract. The contract should be modified to either exclusively use SafeMath library functions or entirely rely on native arithmetic operations, depending on the specific requirements and design considerations. This consistency will help maintain the contract's integrity and mitigate potential vulnerabilities arising from inconsistent arithmetic operations.

## RSML - Redundant SafeMath Library

|             |                     |
|-------------|---------------------|
| Criticality | Minor / Informative |
| Location    | GUFO.sol            |
| Status      | Unresolved          |

### Description

SafeMath is a popular Solidity library that provides a set of functions for performing common arithmetic operations in a way that is resistant to integer overflows and underflows.

Starting with Solidity versions that are greater than or equal to 0.8.0, the arithmetic operations revert to underflow and overflow. As a result, the native functionality of the Solidity operations replaces the SafeMath library. Hence, the usage of the SafeMath library adds complexity, overhead and increases gas consumption unnecessarily.

```
library SafeMath {...}
```

### Recommendation

The team is advised to remove the SafeMath library. Since the version of the contract is greater than `0.8.0` then the pure Solidity arithmetic operations produce the same result.

If the previous functionality is required, then the contract could exploit the `unchecked { ... }` statement.

Read more about the breaking change on

<https://docs.soliditylang.org/en/v0.8.16/080-breaking-changes.html#solidity-v0-8-0-breaking-changes>.

## L04 - Conformance to Solidity Naming Conventions

|                    |                     |
|--------------------|---------------------|
| <b>Criticality</b> | Minor / Informative |
| <b>Location</b>    | GUFO.sol#L64        |
| <b>Status</b>      | Unresolved          |

### Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX\_VALUE, ERROR\_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
uint256 _amount
```

### Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.

Find more information on the Solidity documentation

<https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention>.

## L19 - Stable Compiler Version

|             |                     |
|-------------|---------------------|
| Criticality | Minor / Informative |
| Location    | GUFO.sol#L3         |
| Status      | Unresolved          |

### Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.9;
```

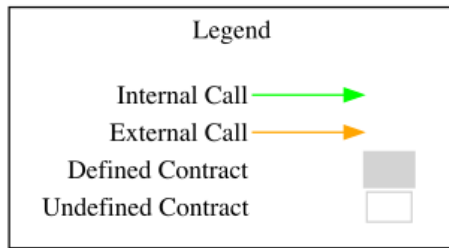
### Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

## Functions Analysis

| Contract | Type           | Bases      |            |           |
|----------|----------------|------------|------------|-----------|
|          | Function Name  | Visibility | Mutability | Modifiers |
|          |                |            |            |           |
| GUFO     | Implementation |            |            |           |
|          |                | Public     | ✓          | -         |
|          | totalSupply    | Public     |            | -         |
|          | balanceOf      | Public     |            | -         |
|          | transfer       | Public     | ✓          | -         |
|          | approve        | Public     | ✓          | -         |
|          | allowance      | Public     |            | -         |
|          | transferFrom   | Public     | ✓          | -         |
|          | burn           | Public     | ✓          | -         |
|          |                |            |            |           |
| SafeMath | Library        |            |            |           |
|          | sub            | Internal   |            |           |
|          | add            | Internal   |            |           |

## Flow Graph



SafeMath (lib)

sub

add

GUFO

<Constructor>

totalSupply

balanceOf

transfer

approve

allowance

transferFrom

burn

## Summary

GUFO contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. GUFO is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

## Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security. Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/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 and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

## About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.



**The Cyberscope team**

<https://www.cyberscope.io>