FomoBSC

Smart Contract Audit Report

SYSFIXED

23 Dec 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
FomoBSC	FOMO	BSC

Addresses

Contract address	0xb93Dab50c96B90C63B590bcafa91Bf2FDd8403AF
Contract deployer address	0x4b1059d2cCeAB40326C6f78CA6C0486d6733bf27

Project Website

https://fomobsc.io/

Codebase

https://bscscan.com/address/0xb93Dab50c96B90C63B590bcafa91Bf2FDd8403AF#contracts

SUMMARY

One-stop crypto investment, social intelligence, state-of-the-art community driven resource app with defi built tools for BNBChain. Think utility, Discover FomoBSC— Projects incubator & accelerator program with market management & cohort staking pools. Ten million supply 0 tax token, with a 3% utility accrued fee for token development. FomoBSC is developed as an improved version of the Ingress App.

Contract Summary

Documentation Quality

This project has a standard of documentation.

- Technical description provided.

Code Quality

The quality of the code in this project is up to standard.

- The official Solidity style guide is followed.

Test Scope

Project test coverage is 100% (Via Codebase).

Audit Findings Summary

Issues Found

- SWC-101 | Arithmetic operation issues discovered on lines 330, 353, 386, 389, 411, 414, 440, 442, and 492.
- SWC-103 | A floating pragma is set on lines 12, 97, 126, 153, 543, 626, 643, 663, 689, and 709, the current pragma Solidity directive is ""^0.8.0"".

CONCLUSION

We have audited the FomoBSC project which has released on December 2022, to discover issues and identify potential security vulnerabilities in FomoBSC Project. This process is used to find technical issues and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with low-risk issues.

The most common issue found in writing code on contracts that do not pose a big risk, writing on contracts is close to the standard of writing contracts in general. The low-level issues found are some arithmetic operation issues and a floating pragma is set in multiple lines.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	PASS
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	ISSUE FOUND
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Check-Effect Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	PASS
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	PASS
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Caller	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS

Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	PASS
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique Id	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS

SMART CONTRACT ANALYSIS

Started	Thu Dec 22 2022 23:18:11 GMT+0000 (Coordinated Universal Time)
Finished	Fri Dec 23 2022 23:18:11 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	StandardBEP20.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged

SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 330

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
329   address owner = _msgSender();
330   _approve(owner, spender, allowance(owner, spender) + addedValue);
331   return true;
332   }
333
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 353

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
352   unchecked {
353   _approve(owner, spender, currentAllowance - subtractedValue);
354   }
355
356   return true;
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 386

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
385   unchecked {
386   _balances[from] = fromBalance - amount;
387   // Overflow not possible: the sum of all balances is capped by totalSupply, and the
sum is preserved by
388   // decrementing then incrementing.
389   _balances[to] += amount;
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 389

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
388   // decrementing then incrementing.
389   _balances[to] += amount;
390   }
391
392   emit Transfer(from, to, amount);
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 411

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
410
411   _totalSupply += amount;
412   unchecked {
413   // Overflow not possible: balance + amount is at most totalSupply + amount, which
is checked above.
414   _balances[account] += amount;
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 414

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
413   // Overflow not possible: balance + amount is at most totalSupply + amount, which
is checked above.
414   _balances[account] += amount;
415   }
416   emit Transfer(address(0), account, amount);
417
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 440

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
439   unchecked {
440   _balances[account] = accountBalance - amount;
441   // Overflow not possible: amount <= accountBalance <= totalSupply.
442   _totalSupply -= amount;
443   }
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 442

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
441   // Overflow not possible: amount <= accountBalance <= totalSupply.
442   _totalSupply -= amount;
443   }
444
445   emit Transfer(account, address(0), amount);
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 492

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- StandardBEP20.sol

Locations

```
491   unchecked {
492   _approve(owner, spender, currentAllowance - amount);
493   }
494   }
495   }
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 12

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
11
12  pragma solidity ^0.8.0;
13
14  /**
15  * @dev Interface of the ERC20 standard as defined in the EIP.
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 97

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
96
97   pragma solidity ^0.8.0;
98
99   /**
100   * @dev Interface for the optional metadata functions from the ERC20 standard.
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 126

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
125
126   pragma solidity ^0.8.0;
127
128   /**
129   * @dev Provides information about the current execution context, including the
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 153

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
152
153   pragma solidity ^0.8.0;
154
155
156
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 543

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
542
543   pragma solidity ^0.8.0;
544
545   /**
546   * @dev Contract module which provides a basic access control mechanism, where
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 626

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
625
626   pragma solidity ^0.8.0;
627
628   /**
629   * @title IBEP20
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 643

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
642
643   pragma solidity ^0.8.0;
644
645
646   /**
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 663

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
662
663   pragma solidity ^0.8.0;
664
665   /**
666   * @title ERC20Decimals
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 689

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
688
689   pragma solidity ^0.8.0;
690
691   interface IPayable {
692   function pay(string memory serviceName, bytes memory signature, address wallet)
external payable;
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 709

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- StandardBEP20.sol

Locations

```
708
709   pragma solidity ^0.8.0;
710
711
712
```

DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. 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 Sysfixed's prior written consent in each instance.

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 considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Sysfixed to perform a security assessment. This report 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 is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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.

This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.

ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.