Shibium

Smart Contract Audit Report

SYSFIXED

12 Jan 2023

TABLE OF CONTENTS

Audited Details

- Audited Project
- Blockchain
- Addresses
- Project Website
- Codebase

Summary

- Contract Summary
- Audit Findings Summary
- Vulnerabilities Summary

Conclusion

Audit Results

Smart Contract Analysis

- Detected Vulnerabilities

Disclaimer

About Us

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Shibium	SHB	Binance Smart Chain

Addresses

Contract address	0x5642833f097880763F0118C6C50dAfF02Cc6EEc2
Contract deployer address	0x1e22248747c3Ac96b0504957417C52967A68506F

Project Website

https://www.shibium.finance/

Codebase

https://bscscan.com/address/0x5642833f097880763F0118C6C50dAfF02Cc6EEc2#code

SUMMARY

Continuing the success of the Axie Doge, we wants to continue bringing users high-quality products that can be applied in practice, solving the problems entangled in the Blockchain world. And this project is called SHIBIUM FINANCE.

Contract Summary

Documentation Quality

Shibium provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by Shibium with the discovery of several low issues.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 522, 536, 551, 552, 565, 577, 592, 606, 620, 634, 650, 673, 696, 722, 1148, 1148 and 1148.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 11, 31, 129, 176, 277, 502, 732, 759 and 835.

CONCLUSION

We have audited the Shibium project released on January 2023 to discover issues and identify potential security vulnerabilities in Shibium Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides a satisfactory result with some low-risk issues.

The issues found in the Shibium smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues and floating pragmas set on 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
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Assert Violation	SWC-110 SWC-123	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 Callee	SWC-112	Delegate calls 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	Wednesday Jan 11 2023 11:17:00 GMT+0000 (Coordinated Universal Time)
Finished	Thursday Jan 12 2023 11:14:36 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Shibium.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-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-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 522

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
521   unchecked {
522   uint256 c = a + b;
523   if (c < a) return (false, 0);
524   return (true, c);
525   }
526
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 536

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
535   if (b > a) return (false, 0);
536   return (true, a - b);
537   }
538   }
539
540
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 551

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
550   if (a == 0) return (true, 0);
551   uint256 c = a * b;
552   if (c / a != b) return (false, 0);
553   return (true, c);
554   }
555
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 552

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
551   uint256 c = a * b;
552   if (c / a != b) return (false, 0);
553   return (true, c);
554   }
555   }
556
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 565

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
564   if (b == 0) return (false, 0);
565   return (true, a / b);
566   }
567   }
568
569
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 577

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
576   if (b == 0) return (false, 0);
577   return (true, a % b);
578   }
579   }
580
581
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 592

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
591   function add(uint256 a, uint256 b) internal pure returns (uint256) {
592   return a + b;
593   }
594
595   /**
596
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 606

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
605   function sub(uint256 a, uint256 b) internal pure returns (uint256) {
606   return a - b;
607   }
608
609   /**
610
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 620

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
619   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
620   return a * b;
621   }
622
623   /**
624
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 634

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
633   function div(uint256 a, uint256 b) internal pure returns (uint256) {
634   return a / b;
635   }
636
637   /**
638
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 650

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
649   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
650   return a % b;
651   }
652
653   /**
654
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 673

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
672   require(b <= a, errorMessage);
673   return a - b;
674   }
675   }
676
677
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 696

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
695   require(b > 0, errorMessage);
696   return a / b;
697   }
698   }
699
700
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 722

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
721   require(b > 0, errorMessage);
722   return a % b;
723   }
724   }
725   }
726
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1148

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
1147   using Address for address;
1148   uint256 public constant maxSupply = 10**11 * 10**18;
1149   IUniswapV2Router02 public uniswapV2Router;
1150   address public uniswapV2Pair;
1151   uint256 public constant marketingSellFee = 4;
1152
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1148

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
1147   using Address for address;
1148   uint256 public constant maxSupply = 10**11 * 10**18;
1149   IUniswapV2Router02 public uniswapV2Router;
1150   address public uniswapV2Pair;
1151   uint256 public constant marketingSellFee = 4;
1152
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1148

low SEVERITY

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

Source File

- Shibium.sol

Locations

```
1147   using Address for address;
1148   uint256 public constant maxSupply = 10**11 * 10**18;
1149   IUniswapV2Router02 public uniswapV2Router;
1150   address public uniswapV2Pair;
1151   uint256 public constant marketingSellFee = 4;
1152
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 11

low SEVERITY

The current pragma Solidity directive is "">=0.5.0"". 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

- Shibium.sol

Locations

```
10
11   pragma solidity >=0.5.0;
12
13   interface IUniswapV2Factory {
14   event PairCreated(address indexed token0, address indexed token1, address pair,
uint);
15
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 31

low SEVERITY

The current pragma Solidity directive is "">=0.6.2"". 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

- Shibium.sol

Locations

```
30
31   pragma solidity >=0.6.2;
32
33   interface IUniswapV2Router01 {
34   function factory() external pure returns (address);
35
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 129

low SEVERITY

The current pragma Solidity directive is "">=0.6.2"". 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

- Shibium.sol

Locations

```
128
129   pragma solidity >=0.6.2;
130
131
132   interface IUniswapV2Router02 is IUniswapV2Router01 {
133
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 176

low SEVERITY

The current pragma Solidity directive is "">=0.4.0"". 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

- Shibium.sol

Locations

```
175   //SPDX-License-Identifier: MIT
176   pragma solidity >=0.4.0;
177
178   interface IERC20 {
179   /**
180
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 277

low SEVERITY

The current pragma Solidity directive is ""^0.8.1"". 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

- Shibium.sol

Locations

```
276
277   pragma solidity ^0.8.1;
278
279   /**
280   * @dev Collection of functions related to the address type
281
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 502

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

- Shibium.sol

Locations

```
501
502   pragma solidity ^0.8.0;
503
504   // CAUTION
505   // This version of SafeMath should only be used with Solidity 0.8 or later,
506
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 732

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

- Shibium.sol

Locations

```
731
732   pragma solidity ^0.8.0;
733
734   /**
735   * @dev Provides information about the current execution context, including the
736
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 759

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

- Shibium.sol

Locations

```
758
759   pragma solidity ^0.8.0;
760
761
762   /**
763
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 835

low SEVERITY

The current pragma Solidity directive is "">=0.4.0"". 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

- Shibium.sol

Locations

```
834
835   pragma solidity >=0.4.0;
836
837   /**
838   * @dev Implementation of the {IERC20} interface.
839
```

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.