Biconomic

Smart Contract Audit Report

SYSFIXED

09 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Biconomic	BMB	Binance Smart Chain

Addresses

Contract address	0xFCaC04229D2b552fd77075E5bCae27C2D6E5b035
Contract deployer address	0x6442D1166A643132b7CE5274cA8ff4adBcc78ea1

Project Website

https://puzziland.com/

Codebase

https://bscscan.com/address/0xFCaC04229D2b552fd77075E5bCae27C2D6E5b035#code

SUMMARY

PuzZiland is the entry portal of puzzle gaming based on blockchain. PuzZiland has a play-to-earn ecosystem where anyone can experience trade NFTs, collect, daily p2p, and group gaming. PuzZiland game allows users to upgrade their puzzle NFT to more parts or create any kind of NFT as a puzzle. BMB$ is our native token on Binance Smart Chain play.

Contract Summary

Documentation Quality

Biconomic 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.

- Standart solidity basecode and rules are already followed with Biconomic 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 163, 163, 279, 319, 343, 344, 388, 390, 422, 436, 451, 452, 465, 477, 492, 506, 520, 534, 550, 573, 596, 622, 747 and 835.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 402.
- SWC-110 | It is recommended to use use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 837 and 838.

CONCLUSION

We have audited the Biconomic released on January 2023 to discover issues and identify potential security vulnerabilities in Biconomic Project. This process finds bugs, technical issues, and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with a low risk issue on the contract project.

The most common issue in writing code on contracts that do not pose a big risk is that writing on contracts is close to the standard of writing contracts in general. Some of the common issues that were found stated variable visibility are not set and a floating pragma is set. We recommended specifying 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.

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	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
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	Sunday Jan 08 2023 02:07:58 GMT+0000 (Coordinated Universal Time)
Finished	Monday Jan 09 2023 01:29:42 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	BMBToken.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

SYSFIXED

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-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 163

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
162   uint8 private constant DECIMALS = 18;
163   uint256 private TOTAL_SUPPLY = 1000000000 * 10**DECIMALS;
164
165   // set the value owner for Ownable contract
166   constructor(address owner) Ownable(owner) {
167
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 279

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
278
279   _approve(sender, msg.sender, currentAllowance - amount);
280   return true;
281   }
282
283
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 296

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
295   function increaseAllowance(address spender, uint256 addedValue) public returns
(bool) {
296   _approve(msg.sender, spender, _allowances[msg.sender][spender] + addedValue);
297   return true;
298   }
299
300
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 319

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
318
319   _approve(msg.sender, spender, currentAllowance - subtractedValue);
320   return true;
321   }
322
323
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 343

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
require(senderBalance >= amount, "BEP20: transfer amount exceeds balance");
_balances[sender] = senderBalance - amount;
_balances[recipient] += amount;

emit Transfer(sender, recipient, amount);

```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 344

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
343   _balances[sender] = senderBalance - amount;
344   _balances[recipient] += amount;
345
346   emit Transfer(sender, recipient, amount);
347   }
348
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 388

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
387   unchecked {
388   _balances[account] = accountBalance - amount;
389   // Overflow not possible: amount <= accountBalance <= totalSupply.
390   TOTAL_SUPPLY -= amount;
391   }
392
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 390

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
389   // Overflow not possible: amount <= accountBalance <= totalSupply.
390   TOTAL_SUPPLY -= amount;
391   }
392
393   emit Transfer(account, address(0), amount);
394
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 422

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
421   unchecked {
422   uint256 c = a + b;
423   if (c < a) return (false, 0);
424   return (true, c);
425   }
426
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 436

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
435   if (b > a) return (false, 0);
436   return (true, a - b);
437   }
438   }
439
440
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 451

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
450   if (a == 0) return (true, 0);
451   uint256 c = a * b;
452   if (c / a != b) return (false, 0);
453   return (true, c);
454   }
455
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 452

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
451   uint256 c = a * b;
452   if (c / a != b) return (false, 0);
453   return (true, c);
454   }
455   }
456
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 465

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
464   if (b == 0) return (false, 0);
465   return (true, a / b);
466   }
467   }
468
469
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 477

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
476   if (b == 0) return (false, 0);
477   return (true, a % b);
478   }
479   }
480
481
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 492

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
491   function add(uint256 a, uint256 b) internal pure returns (uint256) {
492   return a + b;
493   }
494
495   /**
496
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 506

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
505   function sub(uint256 a, uint256 b) internal pure returns (uint256) {
506   return a - b;
507   }
508
509   /**
510
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 520

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
519   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
520   return a * b;
521   }
522
523   /**
524
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 534

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
533   function div(uint256 a, uint256 b) internal pure returns (uint256) {
534   return a / b;
535   }
536
537   /**
538
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 550

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
549   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
550   return a % b;
551   }
552
553   /**
554
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 573

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
572   require(b <= a, errorMessage);
573   return a - b;
574   }
575   }
576
577
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 596

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
595   require(b > 0, errorMessage);
596   return a / b;
597   }
598   }
599
600
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 622

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
621   require(b > 0, errorMessage);
622   return a % b;
623   }
624   }
625   }
626
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 747

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
746
747   return amount - taxAmount;
748   }
749
750   // Obtaining percentage of fee (purchase or sale or transfer) and return this
751
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 835

low SEVERITY

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

Source File

- BMBToken.sol

Locations

```
834
835   for (uint256 i = 0; i < count; i++)
836   {
837   BEP20._transfer(msg.sender, _address [i], _amount);
838   emit AirDrop (_address[i], _amount);
839
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 402

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

- BMBToken.sol

Locations

```
401
402   pragma solidity ^0.8.0;
403
404   // CAUTION
405   // This version of SafeMath should only be used with Solidity 0.8 or later,
406
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 837

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BMBToken.sol

Locations

```
836   {
837   BEP20._transfer(msg.sender, _address [i], _amount);
838   emit AirDrop (_address[i], _amount);
839   }
840
841
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 838

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BMBToken.sol

Locations

```
837   BEP20._transfer(msg.sender, _address [i], _amount);
838   emit AirDrop (_address[i], _amount);
839   }
840
841   return true;
842
```

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.