Baobey

Smart Contract Audit Report

SYSFIXED

07 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Baobey	BeBe	Binance Smart Chain

Addresses

Contract address	0xcc42315f99979fC051Eb7Bee82E869ac41eC514F
Contract deployer address	0x34C739C55301B4bE510aEc9b0F933fbaAa5EA8Fa

Project Website

https://baobeytoken.com/

Codebase

https://bscscan.com/address/0xcc42315f99979fC051Eb7Bee82E869ac41eC514F#code

SUMMARY

BaoBey Token is a utility token, whose vision is to create a Web3 ecosystem of payments by creating a decentralized DApps, so that all holders can use it as a payment method, and will also have a partnership with Visa or MasterCard to grant a debit card. The CEO's business strategies will support the token to inject liquidity and be in constant evolution and development. Certik: audited, KYC and Skynet, to give more security to investors. AMA on Binance Live

Contract Summary

Documentation Quality

Baobey 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 Baobey 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 486, 515, 551, 554, 576, 579, 605, 607, 660, 800, 818, 818, 818, 818, 901, 901, 902, 902, 903 and 903.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 2.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 860 and 863.

CONCLUSION

We have audited the Baobey project released on January 2023 to discover issues and identify potential security vulnerabilities in Baobey 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 Baobey 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, a floating pragma is set, a state variable visibility is not set and weak sources of randomness which is recommended to use external sources of randomness via oracles.

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.	ISSUE FOUND
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	Friday Jan 06 2023 10:04:41 GMT+0000 (Coordinated Universal Time)
Finished	Saturday Jan 07 2023 19:11:43 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	BaoBeyContractToken.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-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-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
485   address owner = _msgSender();
486   _approve(owner, spender, allowance(owner, spender) + addedValue);
487   return true;
488   }
489
490
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 515

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
514   unchecked {
515   _approve(owner, spender, currentAllowance - subtractedValue);
516   }
517
518   return true;
519
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 551

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
550   unchecked {
551   _balances[from] = fromBalance - amount;
552   // Overflow not possible: the sum of all balances is capped by totalSupply, and the
sum is preserved by
553   // decrementing then incrementing.
554   _balances[to] += amount;
555
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 554

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
553   // decrementing then incrementing.
554   _balances[to] += amount;
555   }
556
557   emit Transfer(from, to, amount);
558
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 576

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
575
576   _totalSupply += amount;
577   unchecked {
578   // Overflow not possible: balance + amount is at most totalSupply + amount, which
is checked above.
579   _balances[account] += amount;
580
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 579

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
578   // Overflow not possible: balance + amount is at most totalSupply + amount, which
is checked above.
579   _balances[account] += amount;
580   }
581   emit Transfer(address(0), account, amount);
582
583
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 605

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
604   unchecked {
605   _balances[account] = accountBalance - amount;
606   // Overflow not possible: amount <= accountBalance <= totalSupply.
607   _totalSupply -= amount;
608   }
609
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 607

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
606   // Overflow not possible: amount <= accountBalance <= totalSupply.
607   _totalSupply -= amount;
608   }
609
610   emit Transfer(account, address(0), amount);
611
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 660

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
659   unchecked {
660   _approve(owner, spender, currentAllowance - amount);
661   }
662   }
663   }
664
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 800

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
799   uint8 public constant marketing_amount_tax = 4;
800   uint256 private constant _gas_price_limit = 70 * 1 gwei;
801
802   modifier marketingOrOwner() {
803   require(
804
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 818

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
817   uniswapV2Router = _uniswapV2Router;
818   _mint(mint_wallet, (130 * 10 ** 9) * (10 ** uint256(decimals())));
819   uniswap_v2_pair = IUniswapV2Factory(_uniswapV2Router.factory())
820   .createPair(address(this), _uniswapV2Router.WETH());
821   _setAutomatedMarketMakerPair(uniswap_v2_pair, true);
822
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 818

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
817   uniswapV2Router = _uniswapV2Router;
818   _mint(mint_wallet, (130 * 10 ** 9) * (10 ** uint256(decimals())));
819   uniswap_v2_pair = IUniswapV2Factory(_uniswapV2Router.factory())
820   .createPair(address(this), _uniswapV2Router.WETH());
821   _setAutomatedMarketMakerPair(uniswap_v2_pair, true);
822
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 818

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
817   uniswapV2Router = _uniswapV2Router;
818   _mint(mint_wallet, (130 * 10 ** 9) * (10 ** uint256(decimals())));
819   uniswap_v2_pair = IUniswapV2Factory(_uniswapV2Router.factory())
820   .createPair(address(this), _uniswapV2Router.WETH());
821   _setAutomatedMarketMakerPair(uniswap_v2_pair, true);
822
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 818

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
817   uniswapV2Router = _uniswapV2Router;
818   _mint(mint_wallet, (130 * 10 ** 9) * (10 ** uint256(decimals())));
819   uniswap_v2_pair = IUniswapV2Factory(_uniswapV2Router.factory())
820   .createPair(address(this), _uniswapV2Router.WETH());
821   _setAutomatedMarketMakerPair(uniswap_v2_pair, true);
822
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 901

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
900   ) internal {
901   uint256 burn_fee = (_amount * burn_amount_tax) / 100;
902   uint256 marketing_fee = (_amount * marketing_amount_tax) / 100;
903   _amount = _amount - burn_fee - marketing_fee;
904
905
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 901

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
900   ) internal {
901   uint256 burn_fee = (_amount * burn_amount_tax) / 100;
902   uint256 marketing_fee = (_amount * marketing_amount_tax) / 100;
903   _amount = _amount - burn_fee - marketing_fee;
904
905
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 902

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
901   uint256 burn_fee = (_amount * burn_amount_tax) / 100;
902   uint256 marketing_fee = (_amount * marketing_amount_tax) / 100;
903   _amount = _amount - burn_fee - marketing_fee;
904
905   super._transfer(from, _dead_wallet, burn_fee);
906
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 902

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
901   uint256 burn_fee = (_amount * burn_amount_tax) / 100;
902   uint256 marketing_fee = (_amount * marketing_amount_tax) / 100;
903   _amount = _amount - burn_fee - marketing_fee;
904
905   super._transfer(from, _dead_wallet, burn_fee);
906
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 903

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
902   uint256 marketing_fee = (_amount * marketing_amount_tax) / 100;
903   _amount = _amount - burn_fee - marketing_fee;
904
905   super._transfer(from, _dead_wallet, burn_fee);
906   super._transfer(from, marketing_wallet, marketing_fee);
907
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 903

low SEVERITY

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

Source File

- BaoBeyContractToken.sol

Locations

```
902   uint256 marketing_fee = (_amount * marketing_amount_tax) / 100;
903   _amount = _amount - burn_fee - marketing_fee;
904
905   super._transfer(from, _dead_wallet, burn_fee);
906   super._transfer(from, marketing_wallet, marketing_fee);
907
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 2

low SEVERITY

The current pragma Solidity directive is ""^0.8.15"". 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

- BaoBeyContractToken.sol

Locations

```
1   // SPDX-License-Identifier: MIT
2   pragma solidity ^0.8.15;
3
4   interface IUniswapV2Router01 {
5   function factory() external pure returns (address);
6
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 860

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BaoBeyContractToken.sol

Locations

```
859   _holder_last_transfer_timestamp[_msgSender()] <
860   block.number,
861   "_transfer: Transfer Delay is enabled"
862   );
863   _holder_last_transfer_timestamp[_msgSender()] = block
864
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 863

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BaoBeyContractToken.sol

Locations

```
862   );
863   _holder_last_transfer_timestamp[_msgSender()] = block
864   .number;
865   }
866   }
867
```

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.