



SAFETIN **AUDIT**

BOGGED FINANCE

April 7th, 2022



TABLE OF CONTENTS

- I. SUMMARY**
- II. OVERVIEW**
- III. FINDINGS**
- IV. CONCLUSION**
- V. DISCLAIMER**

SUMMARY

This report was written for [Bogged Finance \(\\$BOG\)](#) in order to find flaws and vulnerabilities in the [Bogged Finance](#) project's source code, as well as any contract dependencies that weren't part of an officially recognized library.

A comprehensive examination has been performed, utilizing Static Analysis, Manual Review, and [Bogged Finance](#) Deployment techniques. The auditing process pays special attention to the following considerations:

- ❖ Testing the smart contracts against both common and uncommon attack vectors
- ❖ Assessing the codebase to ensure compliance with current best practices and industry standards
- ❖ Ensuring contract logic meets the specifications and intentions of the client
- ❖ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- ❖ Through line-by-line manual review of the entire codebase by industry expert

AUDIT OVERVIEW

PROJECT SUMMARY

Project name	BOGGED FINANCE
Description	The bogged finance project aims to build a whole ecosystem, including a trading platform. In order to facilitate exchanges on this platform, the Bog Token is deployed as a primary payment method.
Platform	Binance Smart Chain
Language	Solidity
Codebase	https://bscscan.com/address/0xb09fe1613fe03e7361319d2a43edc17422f36b09

FINDINGS SUMMARY

Vulnerability	Total	Resolved
● Critical	0	0
● Major	0	0
● Medium	3	0
● Minor	2	0
● Informational	2	0

EXECUTIVE SUMMARY

Bogged finance has developed an entire decentralized exchange platform (DEX) for cryptocurrencies called BogSwap. The team also developed a token as an exchange token for the decentralized platform.

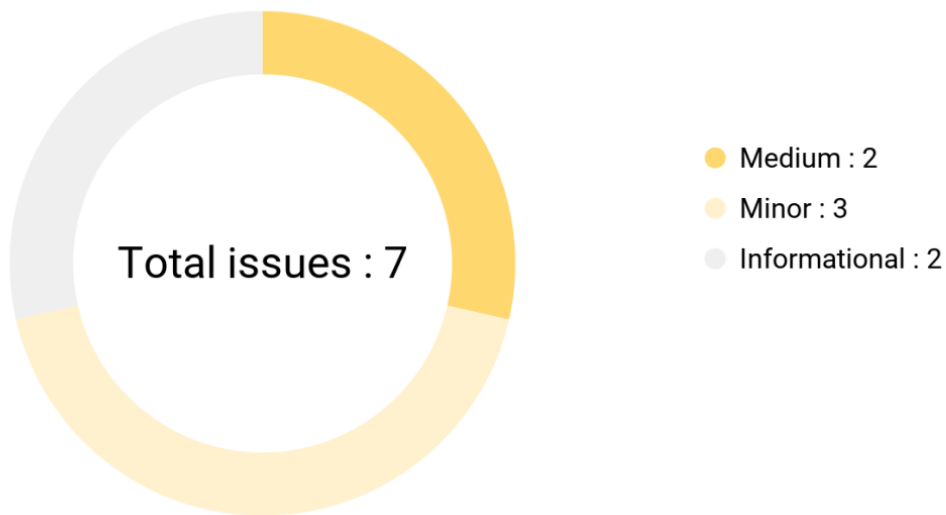
Holding Bog Token (BOG) allows users to access advanced features on the DEX, including stop-loss, portfolio tracking and more. All BOG transactions are taxed by 5% (fees are updatable under a 10% threshold):

- 1% is burned
- 2% is used for LP staking
- 1% is used in the auto-liquidity mechanism

The BOG Token is subject to buybacks and subsequent burns from the Bogged Finance platform, which helps to increase the scarcity and the value of the token.

There have been no major or critical issues related to the codebase and all findings listed here are minor or informational. The major security problems are the dependence on a decentralized exchange platform and the centralization of privileges.

AUDIT FINDINGS



Code	Title	Severity
CENT-1	Centralization of major privileges	● Medium
EXT-1	External protocol dependencies	● Medium
BLOC-1	Usage of block.timestamp	● Minor
BLOC-2	Usage of block.number	● Minor
COMP-1	Unfixed version of compiler	● Minor
MSG-2	Missing error message	● Informational
MSG-1	Unclear error message	● Informational

MSG-1 | Wrong error message

Description

Some of the error messages in the smart contract are unclear or badly formulated. In order to optimize error handling and project maintenance, we recommend using the clearest possible error messages. 3 errors of this type have been found in [bogged.sol](#).

Recommendation

We recommend replacing these ambiguous messages with clearer ones :

```
//Edited code with clearer error messages  
//L167  
require(paused || isAuthorized(msg.sender), "!PAUSED/You  
are not authorized"); _;  
//L171  
require(!paused || isAuthorized(msg.sender), "PAUSED/You  
are not authorized"); _;  
//L280  
require(!paused || isAuthorized(sender) ||  
isAuthorized(msg.sender), "PAUSED/You are not authorized"); _;
```

MSG-2 | Missing error message

Description

Some require statements of the smart contract guarantee validity of conditions but do not throw any error message if these conditions are not met. We recommend documenting errors as best as possible in order to improve their handling and the user interface. 1 issue of this type has been found in [bogged.sol](#).

Recommendation

We recommend adding an error message to the following require statement:

```
//Edited code containing missing error message  
//L402  
require(amount <= 1000 * (10 ** _decimals), "The amount is too  
high");
```

COMP-1 | Unfixed version of compiler

Description

[BOG token's](#) contract does not have locked compiler versions, meaning a range of compiler versions can be used. This can lead to differing bytecodes being produced depending on the compiler version, which can create confusion when debugging as bugs may be specific to a specific compiler version(s).

To rectify this, we recommend setting the compiler to a single version, the lowest version tested to be compatible with the code, an example of this change can be seen below.

1 error of this type has been found in [boggled.sol](#).

Recommendation

We recommend fixing the compiler version to the most recent one :

```
//Edited code containing fixed compiler version  
//L7  
pragma solidity 0.7.4;
```

BLOC-1 | Using block.timestamp

Description

The use of `block.timestamp` can be problematic. The timestamp can be partially manipulated by the miner (see <https://cryptomarketpool.com/block-timestamp-manipulation-attack/>). In this smart contract this is not critical as in the worst case an attacker could force the automatic liquify to run faster.

Recommendation

We fully understand the smart contract's logic of the **BOG token**. The use of `block.timestamp` is required to power the auto-liquify mechanism and we cannot replace it. Nevertheless, it is still useful to point out this kind of potential security problem.

BLOC-2 | Using block.number

Description

The use of `block.number` can be problematic. The timestamp can be partially manipulated by the miner (see <https://cryptomarketpool.com/block-timestamp-manipulation-attack/>). Since the timestamp of a block cannot be fully trusted, the exact block counting at an exact timestamp cannot be fully trusted. In this smart contract this is not critical as in the worst case an attacker could manipulate the information to modify the calculation of the liquidity fee.

Recommendation

We fully understand the smart contract's logic of the `BOG token`. The use of `block.number` is required to power the auto-liquify mechanism and we cannot replace it. Nevertheless, it is still useful to point out this kind of potential security problem.

CENT-1 | Centralization of major privileges

Description

The [authorized](#) modifier of the smart contract gives major privileges over it (authorized people can change the liquidity DEX, and launch the [rescueBNB](#) process)*. This can be a problem, in the case of a hack, an attacker who has taken possession of these privileged accounts could damage the project and the investors.

*This list is not exhaustive but presents the most sensitive points

Recommendation

We recommend at least to use a multi-sig wallet as owner address, and at best to establish a community governance protocol to avoid such centralization.

For more information, see :

<https://solidity-by-example.org /app/multi-sig-wallet/>

EXT-1 | Dependence to an external protocol

Description

The contract is serving as the underlying entity to interact with third party [Bogged Finance](#) protocols. The scope of the audit would treat this third party entity as black box and assume it is fully functional. However in the real world, third parties may be compromised and may have led to assets lost or stolen. We fully understand that the business logic of the [BOG](#) token is designed to work with the [BogSwap](#) DEX.

Recommendation

We encourage the team to constantly monitor the security level of the entire [Bogged Finance](#) project, as the security of the token is highly dependent on the security of the decentralized exchange platform.

GLOBAL SECURITY WARNINGS

These are safety issues for the whole project. They are not necessarily critical problems but they are inherent in the structure of the project itself. Potential attack vectors for these security problems should be monitored.

CENT-1 | Global SPOF (Single Point Of Failure)

Description

The project's smart contracts often have a problem of centralized privileges. The [owner](#) and [authorization](#) system in particular can be subject to attack.

Recommandation

To address this security issue we recommend using a multi-sig wallet, establishing secure project administration protocols and strengthening the security of project administrators.

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 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 without [Safetin'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 considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts [Safetin](#) 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 report should not be used in any way

Safetin security assessment 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.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Safetin's position is that each company and individual are responsible for their own due diligence and continuous security. Safetin'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 fun.