

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



This document may contain confidential information about IT systems and the intellectual property of the Customer as well as information about potential vulnerabilities and methods of their exploitation.

The report containing confidential information can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities fixed – upon a decision of the Customer.

Document

| | |
|--------------------|--|
| Name | Smart Contract Code Review and Security Analysis Report for PureFi. |
| Approved by | Andrew Matiukhin CTO Hacken OU |
| Type | Vesting, Farming, Token |
| Platform | Ethereum / Solidity |
| Methods | Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review |
| Repository | 1. https://github.com/purefiprotocol/token 2. https://github.com/purefiprotocol/eth-bsc-swap-contracts |
| Commit | 1. bbb66a17e4f452d3aa999fabb82a432e9b56d0be 2. d010f53f859a589a31a7d9b55104f77ab0df87d1 |
| Timeline | 22 JULY 2021 - 27 JULY 2021 |
| Changelog | 27 JULY 2021 - INITIAL AUDIT |



Table of contents

| | |
|----------------------------|---|
| Introduction | 4 |
| Scope | 4 |
| Executive Summary | 5 |
| Severity Definitions | 6 |
| Audit overview | 7 |
| Conclusion | 8 |
| Disclaimers | 9 |

Introduction

Hacken OÜ (Consultant) was contracted by PureFi (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of Customer's smart contract and its code review conducted between July 22nd, 2021 - July 27th, 2021.

Scope

The scope of the project is smart contracts in the repository:

Repository 1: <https://github.com/purefiprotocol/token>

Commit 1: bbb66a17e4f452d3aa999fabb82a432e9b56d0be

Files:

```
contracts/PureFiPaymentPlan.sol
contracts/PureFiLinearPaymentPlan.sol
contracts/PureFiFixedDatePaymentPlan.sol
contracts/PureFiFarming.sol
contracts/PureFiToken.sol
contracts/PureFiBotProtection.sol
```

Repository 2: <https://github.com/purefiprotocol/eth-bsc-swap-contracts>

Commit 2: d010f53f859a589a31a7d9b55104f77ab0df87d1

Files:

```
contracts/bep20/BEP20TokenImplementation.sol
contracts/bep20/PureFiBotProtection.sol
```

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

| Category | Check Item |
|-------------|---|
| Code review | <ul style="list-style-type: none">ReentrancyOwnership TakeoverTimestamp DependenceGas Limit and LoopsDoS with (Unexpected) ThrowDoS with Block Gas LimitTransaction-Ordering DependenceStyle guide violationCostly LoopERC20 API violationUnchecked external callUnchecked mathUnsafe type inferenceImplicit visibility levelDeployment ConsistencyRepository ConsistencyData Consistency |

| | |
|-------------------|---|
| Functional review | <ul style="list-style-type: none"> ■ Business Logics Review ■ Functionality Checks ■ Access Control & Authorization ■ Escrow manipulation ■ Token Supply manipulation ■ Assets integrity ■ User Balances manipulation ■ Data Consistency manipulation ■ Kill-Switch Mechanism ■ Operation Trails & Event Generation |
|-------------------|---|

Executive Summary

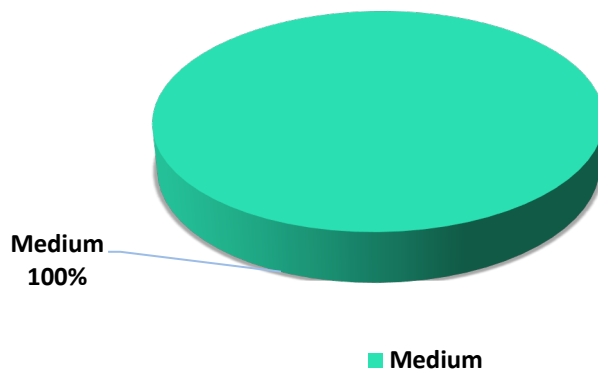
According to the assessment, the Customer's smart contracts are well-secured but contains some edge cases that are recommended to fix.



Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. All found issues can be found in the Audit overview section.

As a result of the audit, security engineers found **2** medium issues.

Graph 1. The distribution of vulnerabilities after the audit.



Severity Definitions

| Risk Level | Description |
|-----------------|---|
| Critical | Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations. |
| High | High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions |
| Medium | Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations. |
| Low | Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution |

Audit overview

■ ■ ■ ■ Critical

No critical issues were found.

■ ■ ■ High

No high severity issues were found.

■ ■ Medium

1. An `_lpTokenAddress` parameter is not validated and reward balances can be messed up.

Contracts: PureFiFarming.sol

Function: addPool

Recommendation: ensure that lp token does not yet exist.

2. When a reward token and the LP token are essentially the same token, the reward and LP tokens staked by users are mixed. Which may lead to the fact that that user cannot withdraw LP tokens in case farming contract lacks reward tokens (or on case of miscalculations when reward tokens sent to contract).

Contracts: PureFiFarming.sol

Function: addPool

Recommendation: forbid setting reward token as LP token.

Customer notice: Such a case will not happen if calculations are done properly and contract is fully funded with reward tokens expected to be claimed by users.

■ Low

No low severity issues were found.



Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

As a result of the audit, security engineers found **2** medium issues.



Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only – we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. Thus, the audit can't guarantee the explicit security of the audited smart contracts.