HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Powerbomb Finance
Date: March 10th, 2022

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for Powerbomb Finance.
Approved by	Andrew Matiukhin | CTO Hacken OU Evgeniy Bezuglyi | SC Department Head at Hacken OU
Type	Vault Curve Staking
Platform	EVM
Language	Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/Powerbomb-Finance/powerbomb-lite
Commit	DE68879C3326452FA3FB8A87FDAC3E91965F541C
Technical Documentation	YES (https://github.com/Powerbomb-Finance/powerbomb-lite/blob/0f86ff1eecdd723be733a9b33ff4ffa3bbdadcee/hardhat/README.md)
JS tests	NO
Website	https://www.powerbomb.finance
Timeline	23 FEBRUARY 2022 - 10 MARCH 2022
Changelog	07 MARCH 2022 - INITIAL AUDIT 10 MARCH 2022 — REMEDIATIONS CHECKS

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by Powerbomb Finance (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contracts.

Scope

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

Repository:
https://github.com/Powerbomb-Finance/powerbomb-lite
Commit:
de68879c3326452fa3fb8a87fdac3e91965f541c
Technical Documentation: Yes
https://github.com/Powerbomb-Finance/powerbomb-lite/blob/0f86ff1eecdd723be733a9b33ff4ffa3bbdadcee/hardhat/README.md
JS tests: No
Contracts:
hardhat/contracts/PowerBombFtmCurveGeist.sol
hardhat/contracts/PowerBombOneCurve.sol
hardhat/contracts/PowerBombAvaxCurve.sol
hardhat/contracts/PowerBombAvaxCurve33.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	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ EIP standards violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency ▪ Kill-Switch Mechanism

Executive Summary

Score measurements details can be found in the corresponding section of the methodology.

Documentation quality

The customer provided average functional requirements and no technical requirements. Total Documentation Quality score is **3** out of **10**.

Code quality

Total CodeQuality score is **3** out of **10**. Code duplications. No unit tests were provided. Lots of commented-out code. Many state readings.

Architecture quality

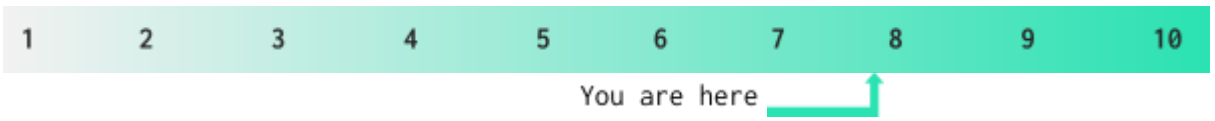
Architecture quality score is **3** out of **10**. All the logic is implemented in one file. Functions are overwhelmed with template code that could be moved to separate functions and be reused.

Security score

As a result of the audit, security engineers found **1** high and **2** low severity issues. High and one low severity issues were fixed before the first remediations check. After the first remediations check there is still **1** low severity issue. The security score is **10** out of **10**. All found issues are displayed in the “Issues overview” section of the report.

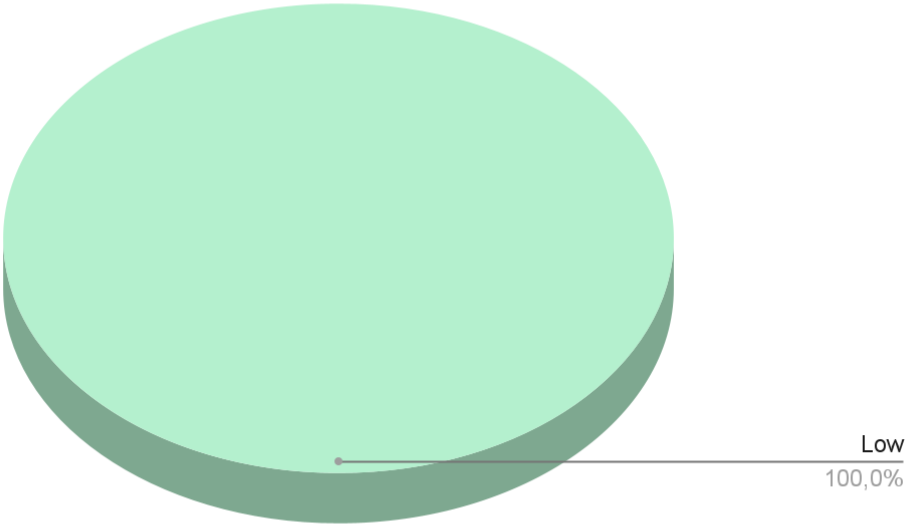
Summary

According to the assessment, the Customer's smart contract has the following score: **7.9**



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Graph 1. The distribution of vulnerabilities after the audit.



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

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

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Audit overview

Critical

No critical issues were found.

High

Possible reentrancy

While the contract calls external contracts whose functionality is out of the audit scope, we couldn't know if there are some protections from reentrancy. Like the "claimReward" function calls "lendingPool.withdraw" whose implementation is unknown, we could not be sure, that the "lendingPool" contract will not call back the user while withdrawing.

Contracts: PowerBombAvaxCurve.sol

Function: claimReward

Recommendation: Please add "nonReentrant" modifier to functions which could be possibly reentered.

Status: Fixed

Medium

No medium severity issues were found.

Low

1. Redundant Statements

   "isDeposit" is a redundant statement because it performs no action, so no code will be generated for such statement and it could be removed.

   Contracts: PowerBombFtmCurveGeist.sol

   Function: _harvest

   Recommendation: Remove redundant statement. To eliminate "warnings" you may declare your function with no named arguments, which will eliminate unneeded memory allocation for a local variable, like the following:

```
function _harvest(bool) internal override {
```

   Status: Still Present

2. The unused state variable

   The public state variable "ibRewardTokenBaseAmt" is never used in the contract. It neither ever initialized nor read.

Contracts: PowerBombOneCurve.sol

Variable: ibRewardTokenBaseAmt

Recommendation: Remove redundant state variable.

Status: Fixed

Recommendations

1. Please make sure all contracts you’re depending on are safe.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

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 the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bug-free 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 the security of smart contracts.

Technical Disclaimer

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