QuillAudits

Audit Report
April, 2022

For

GUNFUNNY

Contents

Scope of the Audit

The scope of this audit was to analyze and document the GunFunny smart contract codebase for quality, security, and correctness.

Checked Vulnerabilities

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

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis
In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis
Static analysis of smart contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis
Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption
In this step, we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.

Issue Categories

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

Risk-level	Description
High	A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.
Medium	The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.
Low	Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.
Informational	These are severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	0	0
Acknowledged	0	0	0	0
Closed	0	1	0	2

Introduction

During the period of **April 4, 2022 to April 08 , 2022** - QuillAudits Team performed a security audit for GunFunny smart contracts.

The code for the audit was taken from the Auditee:
GunFunny Contract: https://github.com/GunfunnyHQ/TokenContracts

Branch: main
Commit: 86ab029

Fixed In: c30dc9d

Contract Address - 0xDED4c8adE892A81D65333DFc3b96a6F9a126E884

Issues Found – Code Review / Manual Testing

A. Contract - GunFunny

High severity issues

No issues found

Medium severity issues

1. Unknown interface implementation,we are not sure about the IBPContract interface, as it was not provided during the audit, which might be a malicious contract .

```
82  interface IBPContract {
83      function protect(address sender, address receiver, uint256 amount) external;
84  }
```

Recommendation: The Implementation smart contract should be provided for audit.

Status: Resolved

Low severity issues

No issues found

Informational Issues

2. **Multiple pragma directives have been used**

Recommendation: Contracts should be deployed using the same compiler version/flags with which they have been tested. Locking the pragma (for e.g. by not using ^ in pragma solidity 0.8.2) ensures that contracts do not accidentally get deployed using an older compiler version with unfixed bugs. Ref: Security Pitfall 2

Status: Resolved

3. **No check or guard For renounceOwnership**

   Recommendation: Contracts should have a double check for renounceOwnership function , because this will make the contract to have address ox000 as the admin, when Admin mistakenly calls the function.

   Status: Resolved

Functional Tests

Complete functional testing report has been attached *here*

Some of the tests performed are mentioned below:

- should be able to mint 5m token at deployment	PASS
- should be able to burn	PASS
- should revert to burnFrom	PASS
- should be able to pause , transfer	PASS
- should be able to approve a contract to spend token	PASS
- should be able transfer	PASS
- should be able to transferFrom	PASS
- should be able to transferOwnership	PASS
- should be able to unpause	PASS
- should be able to acceptOwnership	PASS
- should revert if paused and transfer is called	PASS

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of the GunFunny Smart Contract. We performed our audit according to the procedure described above.

The audit showed one medium and two informational severity issues, which the Auditee has fixed.

Disclaimer

QuillAudits smart contract audit is not a security warranty, investment advice, or an endorsement of the GunFunny Platform. This audit does not provide a security or correctness guarantee of the audited smart contracts.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the GunFunny Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Audit Report
April, 2022

For

QuillAudits

Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com