QuillAudits

Audit Report
July, 2021

Silicon Finance

Contents

Scope of Audit

The scope of this audit was to analyze and document the Silicon Finance Token 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 analyzed 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.
SmartCheck.

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 security of smart contracts.

Code Review / Manual Analysis
Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of 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 possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

Issue Categories

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

High severity issues

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 to be fixed before moving to a live environment.

Medium level severity issues

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 level severity issues

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 four issues which 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	1	0
Acknowledged	0	0	0	0
Closed	0	0	0	0

Introduction

During the period of **July 27, 2021 to July 29, 2021** - QuillAudits Team performed a security audit for Silicon Finance Token smart contracts.

The code for the audit was taken from following the official link:

Note	Date	Commit hash
Version 1	29th July	https://bscscan.com/address/0x532aC600f284E2b048B26ed6112b92b8de1A7a74#code

Issues Found – Code Review / Manual Testing

High severity issues

No issues were found

Medium severity issues

No issues were found

Low level severity issues

1. **Dead code**

   Description
   It was discovered that the function _burn() and _burnFrom() are not used in this contract.

   Unused code is allowed in Solidity and they do not pose a direct security issue. It is best practice though to avoid them as they can:
   cause an increase in computations (and unnecessary gas consumption)
   indicate bugs or malformed data structures and they are generally a sign of poor code quality
   cause code noise and decrease readability of the code

   Remediation
   We recommend removing all unused variables/code from the codebase.

Informational

No issues were found

Functional test

Function Names	Testing results
approve	Passed
decreaseAllowance	Passed
increaseAllowance	Passed
transferOwnerShip	Passed
renounceOwnership	Passed
transfer	Passed
transferFrom	Passed

Automated Testing

Slither

```
INFO:Detectors:
SiliconFinanceToken.allowance(address,address).owner (silicon.sol#425) shadows:
    - Ownable.owner() (silicon.sol#303-305) (function)
SiliconFinanceToken._approve(address,address,uint256).owner (silicon.sol#550) shadows:
    - Ownable.owner() (silicon.sol#303-305) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Context._msgData() (silicon.sol#119-122) is never used and should be removed
SafeMath.div(uint256,uint256) (silicon.sol#218-220) is never used and should be removed
SafeMath.div(uint256,uint256,string) (silicon.sol#233-240) is never used and should be removed
SafeMath.mod(uint256,uint256) (silicon.sol#253-255) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (silicon.sol#268-271) is never used and should be removed
SafeMath.mul(uint256,uint256) (silicon.sol#193-205) is never used and should be removed
SafeMath.sub(uint256,uint256) (silicon.sol#164-166) is never used and should be removed
SiliconFinanceToken._burn(address,uint256) (silicon.sol#529-535) is never used and should be removed
SiliconFinanceToken._burnFrom(address,uint256) (silicon.sol#564-567) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Redundant expression "this (silicon.sol#120)" inContext (silicon.sol#110-123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
SiliconFinanceToken.constructor() (silicon.sol#357-365) uses literals with too many digits:
    - _totalSupply = 100000000 * 1e18 (silicon.sol#361)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
renounceOwnership() should be declared external:
    - Ownable.renounceOwnership() (silicon.sol#322-325)
transferOwnership(address) should be declared external:
    - Ownable.transferOwnership(address) (silicon.sol#331-333)
increaseAllowance(address,uint256) should be declared external:
    - SiliconFinanceToken.increaseAllowance(address,uint256) (silicon.sol#471-474)
decreaseAllowance(address,uint256) should be declared external:
    - SiliconFinanceToken.decreaseAllowance(address,uint256) (silicon.sol#490-493)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

Mythril

```
myth a silicon.sol
The analysis was completed successfully. No issues were detected.
```

THEO

```
theo --rpc-http 127.0.0.1:8545
The account's private key (input hidden)
>
Contract to interact with
> 0x0177f335aE7598b54bc6280A0248681c32D44418
Scanning for exploits in contract: 0x0177f335aE7598b54bc6280A0248681c32D44418
IPC / RPC error: encoding with 'idna' codec failed (UnicodeError: label empty or too long)
No exploits found. You're going to need to load some exploits.

Tools available in the console:
- `exploits` is an array of loaded exploits found by Mythril or read from a file
- `w3` an initialized instance of web3py for the provided HTTP RPC endpoint
- `dump()` writing a json representation of an object to a local file

Check the readme for more info:
https://github.com/cleanunicorn/theo

Theo version v0.8.2.
```

SOLHINT LINTER

```
silicon.sol
  113:27  warning  Code contains empty blocks          no-empty-blocks
  202:5   warning  Error message for require is too long  reason-string
  339:5   warning  Error message for require is too long  reason-string
  510:5   warning  Error message for require is too long  reason-string
  511:5   warning  Error message for require is too long  reason-string
  530:5   warning  Error message for require is too long  reason-string
  551:5   warning  Error message for require is too long  reason-string
  552:5   warning  Error message for require is too long  reason-string

✖ 8 problems (0 errors, 8 warnings)
```

Results

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 Silicon Finance Token platform. We performed our audit according to the procedure described above.

No High, Medium or Informational issues were found during the audit, there is one low severity issue was discovered. We highly recommend addressing them. Also, we recommend analyzing the latest version of the platform.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the Silicon Finance Token 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 Silicon Finance Token Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Silicon Finance

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com