CLAIMSWAP SECURITY ASSESSMENT REPORT

DEC. 02 ~ DEC. 15, 2021
DEC. 16 ~ DEC. 20, 2021
DEC. 29 ~ JAN. 07, 2022
JAN. 19 ~ JAN. 21, 2022

DISCLAIMER

• This document is based on a security assessment conducted by a blockchain security company SOOHO. This document describes the detected security vulnerabilities and also discusses the code quality and code license violations.

• This security assessment does not guarantee nor describe the usefulness of the code, the stability of the code, the suitability of the business model, the legal regulation of the business, the suitability of the contract, and the bug-free status. Audit document is used for discussion purposes only.

• SOOHO does not disclose any business information obtained during the review or save it through a separate media.

• SOOHO presents its best endeavors in smart contract security assessment.

SOOHO

SOOHO with the motto of “Audit Everything, Automatically” researches and provides technology for reliable blockchain ecosystem. SOOHO verifies vulnerabilities through entire development life-cycle with Aegis, a vulnerability analyzer created by SOOHO, and open source analyzers. SOOHO is composed of experts including Ph.D researchers in the field of automated security tools and white-hackers verifying contract codes and detected vulnerabilities in depth. Professional experts in SOOHO secure partners’ contracts from known to zero-day vulnerabilities.

INTRODUCTION

SOOHO conducted a security assessment of Claimswap's DEX, CLS/CLA Token, Distributors and Migrator smart contract from Dec. 2 until Dec. 15, 2021, Dec. 16 until Dec. 20, 2021, Dec. 29 until Jan. 07, 2022, and Jan. 19 until Jan. 21, 2022. The following tasks were performed during the audit period:

• Performing and analyzing the results of Odin, a static analyzer of SOOHO.

• Writing Exploit codes on suspected vulnerability in the contract.

• Recommendations on codes based on best practices and the Secure Coding Guide.

Our security experts participated in a vulnerability analysis of the contract. The experts are professional hackers with Ph.D. academic backgrounds and experiences of receiving awards from national/international hacking competitions such as Defcon, Nuit du Hack, White Hat, SamsungCTF, and etc.

The detected vulnerabilities are as follows: Note 2. We have confirmed that all the issues are resolved. It is recommended to promote the stability of service through continuous code audit and analyze potential vulnerabilities.

ANALYSIS TARGET

The following projects were analyzed during period.

Project	claimswap-contracts-audit	Project	claimswap-contracts-audit-12.14
# of Files	36	# of Files	41
# of Target	31	# of Target	40
# of Lines	6,124	# of Lines	6,477
Project	claimswap audit 2021-12-29	Project	claimswap audit 2022-01-11
# of Files	41	# of Files	41
# of Target	40	# of Target	40
# of Lines	6,690	# of Lines	6,728

KEY AUDIT POINTS & PROCESS

Claimswap is an AMM-based DEX developed by the Claimswap team, a new alternative DEX based on Klaytn. It aims to build decentralized DEX through CLA tokens and CLS tokens, and it supports easy migration for existing DEX users. Accordingly, we have identified issues that mainly occur in the KIPC20 token contract, migration contract, and distribution contract. For example, whether there is any problem in managing access controls and whether DoS occurs.

However, we did not take any internal hackings by administrators into account (e.g., Rug Pull). In addition, the design of protocol is also excluded from technical review.

1 **Automated Vulnerability Analysis**

2 **Manual Code Analysis**

The followings are considered:

- Preferential analysis of codes with greater risks.
- Supervision of Access Control management.
- Analyze whether the code is written under the client's intention.

3 **Review of Exploitability and PoC Code**

The followings are considered:

- Dynamic analysis through code execution.
- Examine possible financial gain by misusing detected vulnerabilities. (e.g., infinite withdrawal)
- Examine the possibility of adverse effect to the token service by misusing detected vulnerabilities. (e.g., Mint)

RISK RATING OF VULNERABILITY

Detected vulnerabilities are listed on the basis of the risk rating of vulnerability.

Critical | High | Medium | Low | Note

The risk rating of vulnerability is set based on OWASP's Impact & Likelihood Risk Rating Methodology as seen on the right. Some issues were rated vulnerable aside from the corresponding model and the reasons are explained in the following results.

Impact \ Likelihood	Low	Medium	High
High	Medium	High	Critical
Medium	Low	Medium	High
Low	Note	Low	Medium

Severity

OVERVIEW

Analysis results based on SWC are categorized into Critical, High, Medium, Low, and Note. SOOHO recommends upgrades on every detected issue.

(NON-ISSUE) NULL VALIDATION IS RECOMMENDED ✓

Additional resources and comments

File Name : `ClaDistributor.sol`
File Location : `contracts/CLS/ClaDistributor.sol`
MD5 : 6c1551034a0e5e94d91a5414691946b1

```
275     /// @notice set Rewarder (airdroper)
276     function setRewarder(IRewarder rewarder_) public onlyOwner {
277         rewarder = rewarder_;
278         emit SetRewarder(address(rewarder));
279     }
```

Details There are configuration functions that skip checking null values. Though the value is set as null, it can be updated without major problems. So we confirmed it as a note severity. But we recommend adding Null validation.

(RESOLVED) EMIT EVENTS AFTER CONFIGURATION ✓

Additional resources and comments

File Name : `FeeDistributor.sol`
File Location : `contracts/CLS/FeeDistributor.sol`
MD5 : 245ff5c38b22017aa6b56bc31e1e66af

```
284     // Update dev address.
285     function setDev(address dev_) public {
286         require(msg.sender == owner() || msg.sender == dev, "dev: wut?");
287         dev = dev_;
288     }
```

Details It is recommended to emit an event for updates of configurations such as the team wallet address or any variables that can affect the service. We have confirmed that the issue is resolved by the last project.

(VERIFIED) SWC-100 ✓

Additional resources and comments

Details We have confirmed the function visibility is appropriate.

(VERIFIED) SWC-101 ✓

Additional resources and comments

Details We have confirmed that the arithmetic operations are safe.

ANALYSIS RESULTS

Analysis results are categorized into Critical, High, Medium, Low, and Note. SOOHO recommends upgrades on every detected issue.

(VERIFIED) SWC-104 ✓

Additional resources and comments

Details We have confirmed that the result of the execution is properly validated.

(VERIFIED) SWC-107 ✓

Additional resources and comments

Details We have confirmed that the reentrancy vulnerability will not work.

(VERIFIED) SWC-108 ✓

Additional resources and comments

Details We have confirmed that the contract is well architected.

(VERIFIED) SWC-113 ✓

Additional resources and comments

Details We have confirmed that the DoS will not work.

(VERIFIED) SWC-116 ✓

Additional resources and comments

Details We have confirmed that the block time issues are none.

(VERIFIED) SWC-118 ✓

Additional resources and comments

Details We have confirmed that the constructor is well developed.

(VERIFIED) SWC-131 ✓

Additional resources and comments

Details We have confirmed that the unused variables are none.

ANALYSIS RESULTS

Analysis results are categorized into Critical, High, Medium, Low, and Note. SOOHO recommends upgrades on every detected issue.

(VERIFIED) CLA DISTRIBUTION CALCULATES CORRECTLY ✓

Additional resources and comments

File Name : `ClaDistributor.sol`
File Location : `contracts/CLS/ClaDistributor.sol`
MD5 : 0f74ff572cfddd7a3776bff6163e0d4b

```
109     /// @notice Return reward multiplier over the given _from to _to block.
110     function claPerBlocks(uint256 from, uint256 to)
111         public
112         view
113         returns (uint256)
```

Details We have verified the distribution functions that calculate the number of CLA tokens during the specific block period.

(VERIFIED) UPGRADABILITY IMPLEMENTED WELL ✓

Additional resources and comments

File Name : `ClsToken.sol`
File Location : `contracts/CLS/ClsToken.sol`
MD5 : 9d85128d0b02ae48d30a35cccc800182

Details We have verified the inheritance of the upgradeable contract used in the CLS token contract.

(VERIFIED) MIGRATOR IMPLEMENTED CORRECTLY ✓

Additional resources and comments

File Name : `Migrator.sol`
File Location : `contracts/Migrator.sol`
MD5 : f684c28823e432bcdcfc8811cde79107

Details We have verified the Migrator that is built for fine user experience.

(VERIFIED) HARVEST IMPLEMENTED CORRECTLY ✓

Additional resources and comments

File Name : `FeeDistributor.sol`
File Location : `contracts/CLS/FeeDistributor.sol`
MD5 : 54daf342ed850c9066b95157081b0dbe

Details We have verified the implementation that collects the fees in each LP pool and distributes them to users and development teams.

CONCLUSION

The source code of the batch auction, migration code, CLA/CLS token contract developed by Claimswap is easy to read and very well organized. We have to remark that contracts are well architected and all the additional features are implemented.

The third target project includes the changes that decimal handling logics while migrating Klayswap to Claimswap, setter function of config variables, guarded logic for Frontrunning or the MEV. Although we verified that the MEV and Frontrunning are not applicable in the Klaytn network since the only Governance Council can participate in consensus and it is 1-sec block time, we have to remark on Claimswap's efforts to enhance user security.

The fourth target project includes the changes that handle the precision and startIdx issues found by Claimswap teams themselves. We have remark their expertise on the Klaytn DeFi ecosystem.

The detected vulnerabilities are as follows: Note 2. We have confirmed that all the issues are resolved. Most of the codes are found out to be compliant with all the best practices. It is recommended to promote the stability of service through continuous code audit and analyze potential vulnerabilities.

SWC/CWE			
	SWC-100 ✓	SWC-107 ✓	SWC-116 ✓
	SWC-101 ✓	SWC-108 ✓	SWC-118 ✓
	SWC-104 ✓	SWC-113 ✓	SWC-131 ✓

Project	claimswap audit 2022-01-11
# of Files	41
# of Target	40
# of Lines	6,728

File Tree — claimswap audit 2022-01-11/contracts

```
├── CLA
│   ├── ClaimToken.sol
│   ├── MasterChef.sol
│   ├── MiningTreasury.sol
│   └── Treasury.sol
├── CLS
│   ├── ClaDistributor.sol
│   ├── ClsToken.sol
│   └── FeeDistributor.sol
├── Migrator.sol
├── WKLAY.sol
├── auction
│   └── AuctionSwap.sol
├── codes
│   ├── AccessControl.sol
│   ├── ERC20.sol
│   ├── ERC20Upgradeable.sol
│   ├── Ownable.sol
│   └── OwnableUpgradeable.sol
├── interfaces/
├── libraries/
├── proxy
│   └── Initializable.sol
└── swap
    ├── UniswapV2ERC20.sol
    ├── UniswapV2Factory.sol
    ├── UniswapV2Pair.sol
    └── UniswapV2Router02.sol
```