CLAIMSWAP SECURITY ASSESSMENT REPORT

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

시작하기 전에

- 본 문서는 블록체인 보안 전문업체 SOOHO에서 진행한 취약점 검사를 바탕으로 작성한 문서로, 보안 취약점의 발견에 초점을 두고 있습니다. 추가적으로 코드 퀄리티 및 코드 라이센스 위반 사항 등에 대해서도 논의합니다.
- 본 문서는 코드의 유용성, 코드의 안정성, 비즈니스 모델의 적합성, 비즈니스의 법적인 규제, 계약의 적합성, 버그 없는 상태에 대해 보장하거나 서술하지 않습니다. 감사 문서는 논의 목적으로만 사용됩니다.
- SOOHO는 회사 정보가 대외비 이상의 성격을 가짐을 인지하고 사전 승인 없이 이를 공개하지 않습니다.
- SOOHO는 업무 수행 과정에서 취득한 일체의 회사 정보를 누설하거나 별도의 매체를 통해 소장하지 않습니다.
- SOOHO는 스마트 컨트랙트 분석에 최선을 다하였음을 밝히는 바입니다.

SOOHO 소개

SOOHO는 Audit Everything, Automatically란 슬로건으로 지속적인 보안을 위해 필요한 기술을 연구하고 서비스합니다. 자체 취약점 분석기들과 오픈소스 분석기들을 기반으로 모든 개발 생애 주기에 걸쳐 취약점들을 검사합니다. SOOHO는 자동화 도구를 연구, 개발하는 보안 분야 박사 연구원들과 탐지 결과와 컨트랙트 코드를 깊게 분석하는 화이트 해커들로 구성되어 있습니다. 보안 분야 전문성을 바탕으로 파트너 사의 컨트랙트를 알려진 취약점과 Zero-day 취약점의 위협으로부터 안전하게 만들어줍니다.

개 요

2021년 12월 2일에서 12월 15일, 12월 16일에서 20일, 12월 29일에서 22년 1월 7일, 1월 19일부터 1월 21일까지 Claimswap팀이 개발한 Claimswap DEX와 CLA/CLS 토큰, 그리고 분배 로직과 LP 토큰에 대한 취약점 분석을 진행하였습니다. 감사 기간 동안 아래의 작업을 수행했습니다.

- SOOHO의 자체 취약점 검사기를 통한 취약점 탐지 및 결과 분석
- 보안 취약점 의심 지점에 대한 익스플로잇(Exploit) 코드 작성
- 컨트랙트 코드 모범 사례와 시큐어 코딩 가이드를 바탕으로 코드의 수정 권고 사항 작성

총 2명의 보안 전문가가 컨트랙트의 취약점을 분석하였습니다. 참여한 보안 전문가는 Defcon, Nuit du Hack, 화이트햇, SamsungCTF 등 국내외의 해킹 대회에서 수상을 하고 보안분야 박사 학위의 학문적 배경을 가지는 등 우수한 해킹 실력과 경험을 가지고 있습니다.

SOOHO를 통해 알려진 취약 코드 시그니쳐를 해당 컨트랙트에서 스캐닝하였습니다. 또한, SWC registry를 비롯한 이슈들이 존재하는지에 대해 주로 확인하였습니다.

분석 결과 **발견된 이슈는 총 2개로 Note 2개 입니다. 모든 이슈가 해결되었음을 확인하였습니다.** 꾸준한 코드 감사를 통해 서비스의 안정을 도모하고 잠재적인 취약점에 대한 분석을 하는 것을 추천 드립니다`.

분석 대상

분석 기간 동안 아래의 프로젝트를 분석하였습니다.

Project	claimswap-contracts-audit
# of Files	36
# of Target	31
# of Lines	6,124

Project	claimswap-contracts-audit-12.14
# of Files	41
# of Target	40
# of Lines	6,477

Project	claimswap audit 2021-12-29
# of Files	41
# of Target	40
# of Lines	6,690

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

주요 감사 포인트 및 프로세스

클레임스왑 팀에서 개발한 Claimswap은 AMM 기반 DEX로 클레이튼 기반의 새로운 대안 DEX입니다. 기존의 DEX 서비스를 사용하는 사람이 쉽게 마이그레이션 가능하고 CLA 토큰과 CLS 토큰을 통한 탈중앙화된 DEX를 구현하였습니다. 이에 따라 KIPC20 토큰 컨트랙트와 마이그레이션 컨트랙트, 분배 컨트랙트에서 주로 발생하는 이슈를 확인하였습니다. 예를 들어, DoS가 발생하는지, 접근 권한은 문제가 없는지, 전송이 올바른지, 정확한 비율에 맞게 스왑이 되는지 등이 포함됩니다.

단, 관리자의 내부 해킹을 비롯해 컨트랙트 외부 서버 환경의 안정성은 고려하지 않았습니다. 본 보고서에서는 언급하지 않았지만 노드 자체에 대한 보안 검증과 외부 연동되는 서비스에 대해서도 검토하기를 제안합니다. 분석은 대상 프로젝트에 포함된 컨트랙트의 기능 안정성에 관한 것입니다.

1

취약점 자동 분석
소프트웨어 검사

2

보안 전문가의
코드 직접 분석

다음을 고려합니다.

- 보다 큰 리스크를 내포하는 코드를 중점적으로 분석합니다.
- 접근 권한의 관리가 잘 되고 있는지 검사합니다.
- 파트너 사가 의도한대로 코드가 작성되었는지 분석합니다.

3

Exploit 분석과
Poc 작성

Exploit 분석과 PoC 작성은 다음을 목표로 합니다.

- 코드 실행을 통한 동적 분석
- 발견된 취약점을 악용해 서비스에 악영향을 끼칠 수 있는지 살펴봅니다.

취약점의 심각성 척도

발견된 취약점은 심각성 척도를 기준으로 나열해서 설명합니다.

Critical High Medium Low Note

심각성 척도는 우측 OWASP의 Impact & Likelihood 기반 리스크 평가 모델을 기반으로 정해졌습니다. 해당 모델과 별개로 심각도가 부여된 이슈는 해당 결과에서 그 이유를 서술합니다.

Impact	Likelihood: Low	Likelihood: Medium	Likelihood: High
High	Medium	High	Critical
Medium	Low	Medium	High
Low	Note	Low	Medium

Severity

분석 결과

분석 결과는 심각도에 따라 Critical, High, Medium, Low, Note로 표현됩니다. Sooho는 발견된 모든 이슈에 대해서 개선하는 것을 권장합니다.

(NON-ISSUE) NULL VALIDATION IS RECOMMENDED ✓

분석 결과에 대한 추가적인 자료 및 코멘트

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 Null 값에 대한 처리가 부재한 변수 설정 함수들이 존재합니다. 해당 값들은 문제없이 다시 갱신할 수 있어 큰 문제로 이어지지 않음을 확인하였습니다. 하지만 Null Validation을 추가하는 것을 권장합니다.

(RESOLVED) EMIT EVENTS AFTER CONFIGURATION ✓

분석 결과에 대한 추가적인 자료 및 코멘트

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 토큰을 분배받는 개발팀의 지갑 주소를 의미하는 변수나 디파이 서비스에 영향을 줄 수 있는 변수값의 갱신에는 event를 발생시키는 것을 권장합니다. 12.29 버전에서 이벤트가 emit 되는 것을 확인하였습니다.

(VERIFIED) SWC-100 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 스마트 컨트랙트 상에서 발생할 수 있는 함수 접근 범위에 대한 부분에 대한 검증을 완료하였습니다.

(VERIFIED) SWC-101 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 정수형 데이터에 대한 오버플로우와 언더플로우 문제는 발생하지 않는 것을 확인하였습니다.

분석 결과

분석 결과는 심각도에 따라 Critical, High, Medium, Low, Note로 표현됩니다. Sooho는 발견된 모든 이슈에 대해서 개선하는 것을 권장합니다.

(VERIFIED) SWC-104 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 호출 데이터에 대한 예외 처리는 SWC의 스펙과 상이하지만 핵심이라고 할 수 있는 결과 값에 대한 분석 구문이 포함되어 있습니다.

(VERIFIED) SWC-107 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details Reentrancy 취약점이 발생하지 않음을 확인하였습니다.

(VERIFIED) SWC-108 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 컨트랙트 클래스에 대한 설계가 잘 되어 있습니다.

(VERIFIED) SWC-113 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details DoS 발생 시나리오에 대한 분석을 완료하였습니다.

(VERIFIED) SWC-116 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 블록타임과 관련된 연산이 있지만 관련해서 충분히 고려되어 설계되었음을 확인하였습니다.

(VERIFIED) SWC-118 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 생성자 이름은 모두 온전하게 작성되어 있습니다.

(VERIFIED) SWC-131 ✓

분석 결과에 대한 추가적인 자료 및 코멘트

Details 분석기를 통해 사용되지 않는 변수는 없음을 확인하였습니다.

분석 결과

분석 결과는 심각도에 따라 Critical, High, Medium, Low, Note로 표현됩니다. Sooho는 발견된 모든 이슈에 대해서 개선하는 것을 권장합니다.

(VERIFIED) CLA DISTRIBUTION CALCULATES CORRECTLY ✓

분석 결과에 대한 추가적인 자료 및 코멘트

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 CLA 토큰의 분배를 위해 특정 블록 기간 동안에 분배되어야 하는 토큰의 양을 계산하는 함수에 대한 검증이 완료되었습니다.

(VERIFIED) UPGRADABILITY IMPLEMENTED WELL ✓

분석 결과에 대한 추가적인 자료 및 코멘트

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

Details CLS 토큰 컨트랙트에 대한 업그레이드 가능한 컨트랙트 상속과 쓰임에 대한 검증을 완료하였습니다.

(VERIFIED) MIGRATOR IMPLEMENTED CORRECTLY ✓

분석 결과에 대한 추가적인 자료 및 코멘트

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

Details 유저 편의를 위해 만들어진 Migrator는 잘 설계되고 구현되었습니다.

(VERIFIED) HARVEST IMPLEMENTED CORRECTLY ✓

분석 결과에 대한 추가적인 자료 및 코멘트

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

Details 각 Pool에 쌓여있는 Fee를 모으고 유저와 개발팀에게 분배하는 구현체에 대한 검증을 완료하였습니다.

검사 결과 요약 및 결론

클레임스왑 팀에서 개발한 CLA에 대한 배치 옥션 컨트랙트와 마이그레이션 코드, CLA와 CLS 토큰 컨트랙트는 이해하기 쉽게 명명되고 용도와 쓰임에 따라 잘 설계되어 있습니다. 대부분 모범 사례를 따르고 있습니다. 컨트랙트들은 매우 잘 설계되었고 그 구현 또한 훌륭하였음을 강조하고 싶습니다.

세번째 분석 대상의 경우, 클레이스왑에서의 마이그레이션에서 발생하는 decimal 차이에 대한 고려와 주요 환경 변수에 대한 업데이트 함수, 프론트러닝이나 mev와 같은 상황에 대해 고려한 코드가 추가되었음을 확인하였습니다. Claimswap이 동작하는 클레이튼이 블록타임이 1초이고 Governance Council 만이 블록 생성에 참여하기에 발생하지 않는다고 판단하였습니다. 하지만 이용자를 위해 팀에서 보다 안전하고 좋은 설계와 구현을 적용한 것을 확인할 수 있었습니다.

네번째 분석 대상의 경우, 클레임스왑팀의 자체 오딧을 통해 발견된 precision 및 startIdx 이슈가 해결된 코드를 확인하였습니다. 클레임스왑팀의 클레이튼 기반 DeFi 서비스들에 대한 이해도를 엿볼 수 있었습니다.

코드 검사 결과 **발견된 이슈는 2개로 Note 2개 입니다. 모든 이슈가 해결되었음을 확인하였습니다.** 꾸준한 코드 감사를 통해 컨트랙트의 안정을 도모하고 잠재적인 취약점에 대한 분석을 하는 것을 추천드립니다.

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
```