

HAECHI AUDIT

Boomco

Smart Contract Security Analysis

Published on :Aug 01, 2022

Version v2.0





HAECHI AUDIT

Smart Contract Audit Certificate



Boomco

Security Report Published by HAECHI AUDIT

v1.0 July 22, 2022

v2.0 Aug 01, 2022

Auditor : Felix Kim

Executive Summary

Severity of Issues	Findings	Resolved	Unresolved	Acknowledged	Comment
Critical	1	1	-	-	-
Major	1	1	-	-	-
Minor	1	-	-	-	-
Tips	-	-	-	-	-

TABLE OF CONTENTS

3 Issues (1 Critical, 1 Major, 1 Minor) Found

[TABLE OF CONTENTS](#)

[ABOUT US](#)

[INTRODUCTION](#)

[SUMMARY](#)

[OVERVIEW](#)

[FINDINGS](#)

[owner가 임의의 유저의 토큰을 전송할 수 있습니다. \(Found - v.1.0\) \(Resolved - v.2.0\)](#)

[chainId를 임의로 변경할 수 있습니다. \(Found - v.1.0\) \(Resolved - v.2.0\)](#)

[ERC165의 interface id가 잘못된 값을 나타내고 있습니다. \(Found - v.1.0\)](#)

[DISCLAIMER](#)

[Appendix A. Test Results](#)

ABOUT US

HAECHI AUDIT은 디지털 자산이 가져올 금융 혁신을 믿습니다. 디지털 자산을 쉽고 안전하게 만들기 위해 HAECHI AUDIT은 '보안'과 '신뢰'라는 가치를 제공합니다. 그로써 모든 사람이 디지털 자산을 부담없이 활용할 수 있는 세상을 꿈꿉니다.

HAECHI AUDIT은 글로벌 블록체인 업계를 선도하는 HAECHI LABS의 대표 서비스 중 하나로, 스마트 컨트랙트 보안 감사 및 개발을 전문적으로 제공합니다.

다년간 블록체인 기술 연구 개발 경험을 보유하고 있는 전문가들로 구성되어 있으며, 그 전문성을 인정받아 블록체인 기술 기업으로는 유일하게 삼성전자 스타트업 육성 프로그램에 선정된 바 있습니다. 또한, 이더리움 재단과 이더리움 커뮤니티 펀드로부터 기술 장려금을 수여받기도 하였습니다.

대표적인 클라이언트 및 파트너사로는 카카오 자회사인 Ground X, LG, 한화, 신한은행 등이 있으며, Sushiswap, 1inch, Klaytn, Badger와 같은 글로벌 블록체인 프로젝트와도 협업한 바 있습니다. 지금까지 약 300여곳 이상의 클라이언트를 대상으로 가장 신뢰할 수 있는 스마트 컨트랙트 보안감사 및 개발 서비스를 제공하였습니다.

문의 : audit@haechi.io

웹사이트 : audit.haechi.io

INTRODUCTION

본 보고서는 Boomco 팀이 제작한 스마트 컨트랙트의 보안을 감사하기 위해 작성되었습니다. HAECHI AUDIT 은 Boomco 팀이 제작한 스마트 컨트랙트의 구현 및 설계가 공개된 자료에 명시한 것처럼 잘 구현이 되어있고, 보안상 안전한지에 중점을 맞춰 감사를 진행했습니다.

CRITICAL

Critical 이슈는 광범위한 사용자가 피해를 볼 수 있는 치명적인 보안 결점으로 반드시 해결해야 하는 사항입니다.

MAJOR

Major 이슈는 보안상에 문제가 있거나 의도와 다른 구현으로 수정이 필요한 사항입니다.

MINOR

Minor 이슈는 잠재적으로 문제를 발생시킬 수 있으므로 수정이 요구되는 사항입니다.

TIPS

Tips 이슈는 수정했을 때 코드의 사용성이나 효율성이 더 좋아질 수 있는 사항입니다.

HAECHI AUDIT은 Boomco 팀이 발견된 모든 이슈에 대하여 개선하는 것을 권장합니다. 이어지는 이슈 설명에서는 코드를 세부적으로 지칭하기 위해서 {파일 이름}#{줄 번호}, {컨트랙트 이름}#{함수/변수 이름} 포맷을 사용합니다. 예를 들면, *Sample.sol:20*은 Sample.sol 파일의 20번째 줄을 지칭하며, *Sample#fallback()* 는 Sample 컨트랙트의 fallback() 함수를 가리킵니다. 보고서 작성을 위해 진행된 모든 테스트 결과는 Appendix에서 확인 하실 수 있습니다.


SUMMARY

Audit에 사용된 코드는 GitHub

(<https://github.com/HAECHI-LABS/audit-boomco/blob/800fe446e421cfe389da22ccbf4c8876967d43be/contracts>)에서 찾아볼 수 있습니다. Audit에 사용된 코드의 마지막 커밋은 “800fe446e421cfe389da22ccbf4c8876967d43be”입니다.

Issues HAECHI AUDIT에서는 Critical 이슈 1개, Major 이슈 1개, Minor 이슈 1개를 발견하였으며 수정했을 때 코드의 사용성이나 효율성이 더 좋아질 수 있는 사항들을 0개의 Tips 카테고리로 나누어 서술하였습니다.

Update [v2.0] - 새로운 커밋 “097bd2968cc7e0ea39ac1a1b47240755da08e974” 에서 Critical 이슈 1개, Major 이슈 1개가 수정되었습니다.

Severity	Issue	Status
 CRITICAL	owner가 임의의 유저의 토큰을 전송할 수 있습니다.	(Found - v1.0) (Resolved - v2.0)
 MAJOR	chainId를 임의로 변경할 수 있습니다.	(Found - v1.0) (Resolved - v2.0)
 MINOR	ERC165의 interface id가 잘못된 값을 나타내고 있습니다.	(Found - v1.0)

OVERVIEW

Contracts subject to audit

- ❖ Gov
- ❖ Commitable
- ❖ Interface

FINDINGS

CRITICAL

owner가 임의의 유저의 토큰을 전송할 수 있습니다. (Found - v.1.0) (Resolved - v.2.0)

```
function approveForSpending(address owner, uint rawAmount) onlyCommitter
isNotDestroyed external{
    uint96 amount;
    if (rawAmount == uint(-1)) {
        amount = uint96(-1);
    } else {
        amount = safe96(rawAmount, "Comp::approve: amount exceeds 96 bits");
    }

    allowances[owner][msg.sender] = amount;

    emit Approval(owner, msg.sender, amount);
}
```

[<https://github.com/HAECHI-LABS/audit-boomco/blob/800fe446e421cfe389da22ccbf4c8876967d43be/contracts/DAO/Gov.sol#L106-L117>]

Issue

Gov#approveForSpending() 함수는 committer가 Commitable#isDestroyed 변수가 true가 아닌 경우에 임의의 유저의 allowance를 변경할 수 있습니다. committer는 owner가 임의로 추가하거나 제거할 수 있으므로, 결국 owner가 임의의 유저의 토큰을 전송할 수 있습니다.

Recommendation

Gov#approveForSpending() 함수를 제거하시는 것을 추천드립니다.

Update

[v2.0] - 새로운 커밋 "097bd2968cc7e0ea39ac1a1b47240755da08e974"에서 해당 함수가 삭제되어 이슈가 해결되었습니다.

⚠ MAJOR

chainId를 임의로 변경할 수 있습니다. (Found - v.1.0) (Resolved - v.2.0)

```
function setChainId(uint _chainId) external {  
    chainId = _chainId;  
}
```

[<https://github.com/HAECHI-LABS/audit-boomco/blob/800fe446e421cfe389da22ccbf4c8876967d43be/contracts/DAO/Gov.sol#L320-L322>]

Issue

`Gov#setChainId()` 함수는 누구나 호출 할 수 있으며, 컨트랙트의 chainId 변수를 변경하는 함수입니다. 누구나 호출 할 수 있으므로, 컨트랙트의 chainId 변수의 일관성을 보장 할 수 없습니다.

이 경우, 해당 컨트랙트가 실제로는 동일한 체인에도 동작하고 있더라도 `Gov#setChainId()` 함수를 통하여 chainId를 변경하여 `Gov#delegateBySig()` 함수가 정상적으로 동작하지 않도록 하는 DoS 공격이 가능합니다. 극단적인 경우, 다른 체인에서 발행된 signature를 이용하는 replay 공격도 가능합니다.

Recommendation

chainId 값의 경우 EVM assembly로 받아올 수 있기에, `Gov#setChainId()` 함수는 삭제하시는 것을 추천드립니다.

Update

[v2.0] - 새로운 커밋 “097bd2968cc7e0ea39ac1a1b47240755da08e974”에서 해당 함수가 삭제되어 이슈가 해결되었습니다.

○ MINOR

ERC165의 interface id가 잘못된 값을 나타내고 있습니다. (Found - v.1.0)

```
contract Interface {  
    /*  
     * bytes4(keccak256('supportsInterface(bytes4)')) = 0x01ffc9a7  
     */  
    bytes4 private constant _INTERFACE_ID_ERC165 = 0x36372b07;  
  
    /**  
     * @dev 지원 여부에 대한 인터페이스 ID의 매핑  
     Mapping of interface ids to whether or not it's supported.  
     */  
    mapping(bytes4 => bool) private _supportedInterfaces;
```

[<https://github.com/HAECHI-LABS/audit-boomco/blob/800fe446e421cfe389da22ccbf4c8876967d43be/contracts/Utils/Interface.sol#L5>]

Issue

Interface 컨트랙트에 정의된 _INTERFACE_ID_ERC165 변수는 실제로 ERC20의 interface id 를 가리키고 있습니다. 따라서 외부에서 Interface 컨트랙트의 interface 지원 여부를 확인하고자 할 때, 잘못된 정보를 제공할 수 있습니다.

Recommendation

_INTERFACE_ID_ERC165 변수를 적절한 값으로 변경하시는 것을 추천드립니다.

DISCLAIMER

해당 리포트는 투자에 대한 조언, 비즈니스 모델의 적합성, 버그 없이 안전한 코드를 보증하지 않습니다. 해당 리포트는 알려진 기술 문제들에 대한 논의의 목적으로만 사용됩니다. 리포트에 기술된 문제 외에도 메인넷상의 결함 등 발견되지 않은 문제들이 있을 수 있습니다. 안전한 스마트 컨트랙트를 작성하기 위해서는 발견된 문제들에 대한 수정과 충분한 테스트가 필요합니다.

Appendix A. Test Results

아래 결과는, 보안 감사 대상인 스마트 컨트랙트의 주요 로직을 커버하는 unit test 결과입니다.
붉은색으로 표시된 부분은 이슈가 존재하여 테스트에 통과하지 못한 테스트 케이스입니다.

Gov

#constructor()

- ✓ should set name properly (69ms)
- ✓ should set symbol properly
- ✓ should set decimals properly
- ✓ should set initial supply properly
- ✓ supports ERC20 interfaces

1) supports ERC165 interfaces

ERC20 Spec

#transfer()

- ✓ should fail if recipient is ZERO_ADDRESS (62ms)
- ✓ should fail if sender's amount is lower than balance (41ms)

when succeeded

- ✓ sender's balance should decrease
- ✓ recipient's balance should increase
- ✓ should emit Transfer event

#transferFrom()

- ✓ should fail if sender is ZERO_ADDRESS
- ✓ should fail if recipient is ZERO_ADDRESS
- ✓ should fail if sender's amount is lower than transfer amount
- ✓ should fail if allowance is lower than transfer amount
- ✓ should fail even if try to transfer sender's token without approval process

when succeeded

- ✓ sender's balance should decrease
- ✓ recipient's balance should increase
- ✓ should emit Transfer event
- ✓ allowance should decrease
- ✓ should emit Approval event

#approve()

valid case

- ✓ allowance should set appropriately
- ✓ should emit Approval event

#approveForSpending()

- ✓ should fail if msg.sender is not committer
- ✓ should fail if contract is destroyed (38ms)

valid case

- ✓ allowance should set appropriately

- ✓ should emit Approval event

Gov spec

#delegate()

Delegation: address(0) -> address(0)

- ✓ Nothing happens

Delegation: address(0) -> delegatee

- ✓ delegatee Votes changes
- ✓ should emit DelegateChanged Event
- ✓ should emit DelegateVotesChanged Event

Delegation: nonzero delegatee -> new nonzero delegatee

- ✓ delegatee Votes change
- ✓ should emit DelegateChanged Event
- ✓ should emit DelegateVotesChanged Event - previous delegatee
- ✓ should emit DelegateVotesChanged Event - new delegatee

Delegation: nonzero delegatee -> address(0)

- ✓ delegatee Votes change
- ✓ should emit DelegateChanged Event
- ✓ should emit DelegateVotesChanged Event

Delegation: multiple delegator set same delegatee

- ✓ delegatee Votes change
- ✓ should emit DelegateChanged Event
- ✓ should emit DelegateVotesChanged Event

#getPriorVotes()

- ✓ should fail if blockNumber is not finalized
- ✓ should return 0 if account does not have any checkpoint

valid case - numCheckpoints == 1

- ✓ return the appropriate number of votes (141ms)

valid case - numCheckpoints > 1

- ✓ return the appropriate number of votes (2984ms)

#delegateBySig()

- ✓ should fail if nonce mismatch
- ✓ should fail if signature expired

valid case

- ✓ delegate success
- ✓ should emit DelegateChanged event
- ✓ should emit DelegateVotesChanged event

End of Document