4th OCTOBER 2021

SMART CONTRACT AUDIT REPORT

-

version v1.0

ERC20 Security Audit and General Analysis

HAECHI AUDIT

Table of Contents

0 Issues (0 Critical, 0 Major, 0 Minor) Found

About HAECHI AUDIT

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.

HAECHI AUDIT의 보안감사 보고서는 전세계 암호화폐 거래소들의 신뢰를 받고 있습니다. 실제로 많은 클라이언트들이 HAECHI AUDIT 스마트 컨트랙트 보안감사를 거친 후에, Huobi, OKEX, Upbit, Bithumb 등에 성공적으로 상장하였습니다.

대표적인 클라이언트 및 파트너사로는 글로벌 블록체인 프로젝트와 포춘 글로벌 500대 기업들이 있으며, 카카오의 자회사인 Ground X, Carry 프로토콜, Metadium, LG, 한화, 신한은행 등이 있습니다. 지금까지 약 60여곳 이상의 클라이언트를 대상으로 가장 신뢰할 수 있는 스마트 컨트랙트 보안감사 및 개발 서비스를 제공하였습니다.

문의 : audit@haechi.io
웹사이트 : audit.haechi.io

01. Introduction

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

발견된 이슈는 중요도 차이에 따라 CRITICAL, MAJOR, MINOR, TIPS 로 나누어집니다.

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 AUDIT는 FriendsGames 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.

02. Summary

Audit에 사용된 코드는 Github (https://github.com/HAECHI-LABS/audit-FriendsGames/blob/master/contracts/31.BORAToken-klaytn-remix-20210914.sol)에서 찾아볼 수 있습니다.

Audit에 사용된 코드의 마지막 커밋은 "d4bfe6c3ad7b315eeb4c7ec0ef95f4b2aed202ad"입니다

Issues

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

03. Overview

Contracts Subject to Audit

- Context
- SafeMath
- IERC20
- ERC20
- ERC20Detailed
- Roles
- BurnerRole
- ERC20Burnable
- PauserRole
- Pausable
- ERC20Pausable
- Ownable
- WhitelistAdminRole
- WhitelistedRole
- BlacklistedRole
- Address
- SafeERC20
- LockedToken
- BoraToken

Roles

BoraToken Smart contract에는 다음과 같은 권한이 있습니다.

- **Owner**
- **Pauser**
- **Burner**
- **WhitelistAdmin**

각 권한의 제어에 대한 명세는 다음과 같습니다.

Role	MAX	Addable	Deletable	Transferable	Renouncable
Owner	1	X	X	O	X
Pauser	∞	O	O	X	O (except owner)
Burner	∞	O	O	X	O (except owner)
WhitelistAdmin	∞	O	O	X	O (except owner)

각 권한으로 접근 할 수 있는 기능은 다음과 같습니다.

Role	Functions
Owner	*BoraToken#transferOwnership()* *BoraToken#renounceOwnership()*
Pauser	*PauserRole#addPauser()* *Pausable#pause()* *Pasuable#unpause()* *BoraToken#lock()* *BoraToken#addAdmin()* *BoraToken#renounceAdmin()* *BoraToken#renouncePauser()*
Burner	*BurnerRole#addBurner()* *BoraToken#burn()* *BoraToken#burnFrom()* *BoraToken#addAdmin()*

	BoraToken#renounceAdmin() *BoraToken#renouncePauser()*
WhitelistAdmin	*WhitelistAdminRole#addWhitelistAdmin()* *WhitelistedRole#addWhitelisted()* *WhitelistedRole#removeWhitelisted()* *BlacklistedRole#addBlacklisted()* *BlacklistedRole#removeBlacklisted()* *BoraToken#addAdmin()* *BoraToken#renounceAdmin()* *BoraToken#renounceWhitelistAdmin()*

04. Issues Found

발견된 이슈가 없습니다.

05. 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 클레이튼 상의 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.

```
Address
  #isContract()
    ✓ returns false for account address
    ✓ returns true for contract address
  #sendValue()
    ✓ should fail if try to transfer more than sender contract amounts
    when sender contract has ethers
      ✓ sends 0 wei
      ✓ sends non-zero amounts
      ✓ sends the whole balance
      ✓ should fail if try to send more than the amounts
    with contract recipient
      ✓ sends ether
      ✓ should fail if recipient reverts

 BlacklistedRole
  #isBlacklisted()
    ✓ should return true if account exists in blacklisted
    ✓ should return false if account doesn't exist in blacklisted
  #addBlacklisted()
    ✓ should fail if msg.sender is not in blacklisted
    ✓ should fail if account is already blacklisted
    valid case
      ✓ account should added to blacklisted successfully
      ✓ should emit BlacklistedAdded event
  #removeBlacklisted()
    ✓ should fail if msg.sender is not whitelistAdmin
    ✓ should fail if msg.sender is not in blacklisted
    valid case
      ✓ account should removed from blacklisted successfully
      ✓ should emit BlacklistedRemoved event

 BoraToken
  #constructor()
    ✓ should set name properly
```

```
  ✓ should set symbol properly
  ✓ should set decimals properly
  ✓ should set initial supply properly
  ✓ contract deployer granted a ownerRole
  ✓ contract deployer granted a burnerRole
  ✓ contract deployer granted a whitelistAdminRole
ERC20 Spec
 #transfer()
   ✓ should fail if recipient is ZERO_ADDRESS
   ✓ should fail if sender's amount is lower than balance
   ✓ should fail if paused
   ✓ should fail if msg.sender is blacklisted
   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
   ✓ should fail if paused
   ✓ should fail if msg.sender is blacklisted
   ✓ should fail if sender is blacklisted
   when succeeded
     ✓ sender's balance should decrease
     ✓ recipient's balance should increase
     ✓ should emit Transfer event
     ✓ allowance should decrease
     ✓ should emit Approval event
 #approve()
   ✓ should fail if spender is ZERO_ADDRESS
   valid case
     ✓ allowance should set appropriately
     ✓ should emit Approval event
 #increaseAllowance()
   ✓ should fail if spender is ZERO_ADDRESS
   ✓ should fail if overflows
   valid case
     ✓ allowance should set appropriately
     ✓ should emit Approval event
```

```
  #decreaseAllowance()
    ✓ should fail if spender is ZERO_ADDRESS
    ✓ should fail if overflows
    valid case
      ✓ allowance should set appropriately
      ✓ should emit Approval event
ERC20Burnable spec
  #burn()
    ✓ should fail if msg.sender is not burner
    ✓ should fail if try to burn more than burner's balance
    valid case
      ✓ totalSupply should decrease
      ✓ account's balance should decrease
      ✓ should emit Transfer event
      ✓ should emit Burn event
  #burnFrom()
    ✓ should fail if msg.sender is not burner
    ✓ should fail if account is ZERO_ADDRESS
    ✓ should fail if account's amount is lower than burn amount
    ✓ should fail if allowance is lower than burn amount
    ✓ should fail even if try to burn account's this.token without approve process
    valid case
      ✓ totalSupply should decrease
      ✓ account's balance should decrease
      ✓ allowance should decrease
      ✓ should emit Transfer event
      ✓ should emit Burn event
      ✓ should emit Approval event
BoraToken Role
  #addAdmin()
    ✓ should fail if msg.sender is not burner, pauser nor witelistAdmin
    ✓ should fail if try to add admin to ZERO_ADDRESS
    valid case
      ✓ new admin granted burnerRole
      ✓ new admin granted pauserRole
      ✓ new admin granted whiltelistAdminRole
      ✓ should emit AdminAded event
  #renounceAdmin()
    ✓ should fail if msg.sender is not burner, pauser nor witelistAdmin
    ✓ should fail if msg.sender is owner
    valid case
      ✓ admin renounced burnerRole
```

```
    ✓ admin renounced pauserRole
    ✓ admin renounced whiltelistAdminRole
    ✓ should emit AdminRemoved event
  #transferOwnership()
   ✓ should fail if msg.sender is not owner
   valid case
    ✓ new owner granted ownerRole
    ✓ new owner granted burnerRole
    ✓ new owner granted pauserRole
    ✓ new owner granted whiltelistAdminRole
    ✓ should emit AdminAdded event
    ✓ should emit OwnershiptTransferred event
  #renounceOwnership()
   ✓ renounceOwnership is disabled
  #renounceBurner()
   ✓ should fail if msg.sender is not burner
   ✓ should fail if msg.sender is owner
   valid case
    ✓ burner renounced burnerRole
    ✓ should emit BurnerRemoved event
  #renouncePauser()
   ✓ should fail if msg.sender is not pauser
   ✓ should fail if msg.sender is owner
   valid case
    ✓ pauser renounced pauserRole
    ✓ should emit PauserRemoved event
  #renounceWhitelistAdmin()
   ✓ should fail if msg.sender is not whitelistAdmin
   ✓ should fail if msg.sender is owner
   valid case
    ✓ whitelistAdmin renounced whitelistAdminRole
    ✓ should emit WhitelistAdminRemoved event
 #lock()
  ✓ should fail if msg.sender is not pauser
  ✓ should fail if msg.sender does not have enough balance
  valid case
   ✓ token locked
   ✓ after duration, beneficiary can claim token
   ✓ should emit Lock event

BurnerRole
 #isBurner()
```

```
    ✓ should return true if account exists in burner
    ✓ should return false if account doesn't exist in burner
  #addBurner()
    ✓ should fail if msg.sender is not in burner
    ✓ should fail if account is already burner
    valid case
      ✓ account should added to burner successfully
      ✓ should emit BurnerAdded event
  #renounceBurner()
    ✓ should fail if msg.sender is not in burner
    valid case
      ✓ account should removed from burner successfully
      ✓ should emit BurnerRemoved event

 LockedToken
  #constructor()
    ✓ should fail if donor is ZERO_ADDRESS
    ✓ should fail if beneficiary is ZERO_ADDRESS
    ✓ should fail if release time is before current time
  #revoke()
    ✓ should fail if tokens are not revocable
    ✓ should fail if msg.sender is not donor
    ✓ should fail if no tokens to revoke
    valid case
      ✓ donor token balance should increase
      ✓ should emit Revoke event
  #claim()
    ✓ should fail if current time is before release time
    ✓ should fail if no tokens to claim
    valid case
      ✓ beneficiary earns token
      ✓ should emit Claim event

Ownable
  #constructor()
    ✓ should set msg.sender to owner
  #transferOwnership()
    ✓ should fail if msg.sender is not owner
    ✓ should fail if try to transfer ownership to AddressZero
    valid case
      ✓ should change owner to newOwner
      ✓ should emit OwnershipTransferred event
```

```
  #renounceOwnership()
    ✓ should fail when msg.sender is not owner
    valid case
      ✓ owner release ownership
      ✓ should emit OwnershipTransferred event

Pausable
  #constructor()
    ✓ paused set properly
  #pause()
    ✓ should fail if msg.sender is not owner
    ✓ should fail when already paused
    valid case
      ✓ should change paused to true
      ✓ should emit Pause event
  #unpause()
    ✓ should fail if msg.sender is not owner
    ✓ should fail when already unpaused
    valid case
      ✓ should change paused to false
      ✓ should emit Unpause event

PauserRole
  #isPauser()
    ✓ should return true if account exists in pauser
    ✓ should return false if account doesn't exist in pauser
  #addPauser()
    ✓ should fail if msg.sender is not in pauser
    ✓ should fail if account is already pauser
    valid case
      ✓ account should added to pauser successfully
      ✓ should emit PauserAdded event
  #renouncePauser()
    ✓ should fail if msg.sender is not in pauser
    valid case
      ✓ account should removed from pauser successfully
      ✓ should emit PauserRemoved event

Roles
  #add()
    ✓ should fail if account already has role
    ✓ should fail if account is AddressZero
```

```
  valid case
    ✓ account should added to role
 #remove()
  ✓ should fail if account is AddressZero
  ✓ should fail if account does not have any role
  valid case
    ✓ account should removed from role
 #has()
  ✓ should fail if account is AddressZero
  valid case
    ✓ should return true if account has role
    ✓ should return false if account does not have role

SafeERC20
 with address that has no contract code
  ✓ reverts on transfer
  ✓ reverts on transferFrom
  ✓ reverts on approve
  ✓ reverts on increaseAllowance
  ✓ reverts on decreaseAllowance
 with token that returns false on all calls
  ✓ reverts on transfer
  ✓ reverts on transferFrom
  ✓ reverts on approve
  ✓ reverts on increaseAllowance
  ✓ reverts on decreaseAllowance
 with token that returns true on all calls
  ✓ doesn't revert on transfer
  ✓ doesn't revert on transferFrom
  approvals
    with zero allowance
      ✓ doesn't revert when approving a non-zero allowance
      ✓ doesn't revert when approving a zero allowance
      ✓ doesn't revert when increasing the allowance
      ✓ reverts when decreasing the allowance
    with non-zero allowance
      ✓ reverts when approving a non-zero allowance
      ✓ doesn't revert when approving a zero allowance
      ✓ doesn't revert when increasing the allowance
      ✓ doesn't revert when decreasing the allowance to a positive value
      ✓ reverts when decreasing the allowance to a negative value
 with token that returns no boolean values
```

```
    ✓ doesn't revert on transfer
    ✓ doesn't revert on transferFrom
   approvals
     with zero allowance
       ✓ doesn't revert when approving a non-zero allowance
       ✓ doesn't revert when approving a zero allowance
       ✓ doesn't revert when increasing the allowance
       ✓ reverts when decreasing the allowance
     with non-zero allowance
       ✓ reverts when approving a non-zero allowance
       ✓ doesn't revert when approving a zero allowance
       ✓ doesn't revert when increasing the allowance
       ✓ doesn't revert when decreasing the allowance to a positive value
       ✓ reverts when decreasing the allowance to a negative value

SafeMath
  #add()
    ✓ adds correctly
    ✓ reverts on addition overflow
  #sub()
    ✓ subtracts correctly
    ✓ reverts if subtraction result would be negative
  #mul()
    ✓ multiplies correctly
    ✓ multiplies by zero correctly
    ✓ reverts on multiplication overflow
  #div()
    ✓ divides correctly
    ✓ divides zero correctly
    ✓ returns complete number result on non-even division
    ✓ reverts on division by zero
  #mod()
    ✓ reverts with a 0 divisor
   modulos correctly
     ✓ when the dividend is smaller than the divisor
     ✓ when the dividend is equal to the divisor
     ✓ when the dividend is larger than the divisor
     ✓ when the dividend is a multiple of the divisor

WhitelistAdminRole
  #isWhitelistAdmin()
    ✓ should return true if account exists in WhitelistAdmin
```

```
    ✓ should return false if account doesn't exist in WhitelistAdmin
  #addWhitelistAdmin()
    ✓ should fail if msg.sender is not in WhitelistAdmin
    ✓ should fail if account is already WhitelistAdmin
    valid case
      ✓ account should added to WhitelistAdmin successfully
      ✓ should emit WhitelistAdminAdded event
  #renounceWhitelistAdmin()
    ✓ should fail if msg.sender is not in WhitelistAdmin
    valid case
      ✓ account should removed from WhitelistAdmin successfully
      ✓ should emit WhitelistAdminRemoved event

 WhitelistedRole
  #isWhitelisted()
    ✓ should return true if account exists in whitelisted
    ✓ should return false if account doesn't exist in whitelisted
  #addWhitelisted()
    ✓ should fail if msg.sender is not in whitelistedAdmin
    ✓ should fail if account is already whitelisted
    valid case
      ✓ account should added to whitelisted successfully
      ✓ should emit WhitelistedAdded event
  #removeWhitelisted()
    ✓ should fail if msg.sender is not whitelistAdmin
    ✓ should fail if msg.sender is not in whitelisted
    valid case
      ✓ account should removed from whitelisted successfully
      ✓ should emit WhitelistedRemoved event
  #renounceWhitelisted()
    ✓ should fail if msg.sender is not in whitelisted
    valid case
      ✓ account should removed from whitelisted successfully
      ✓ should emit WhitelistedRemoved event
```

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/					
BoraTokenI.sol	**100**	**100**	**100**	**100**	

[표 1] Test Case Coverage