29th JANUARY 2021

SMART CONTRACT AUDIT REPORT

-

version v2.0

Smart Contract Security Audit and General Analysis

HAECHI AUDIT

Table of Contents

0 Issues (0 Critical, 0 Major, 2 Minor) Found

About HAECHI AUDIT

HAECHI AUDIT is a global leading smart contract security audit and development firm operated by HAECHI LABS. HAECHI AUDIT consists of professionals with years of experience in blockchain R&D and provides the most reliable smart contract security audit and development services.

So far, based on the HAECHI AUDIT's security audit report, our clients have been successfully listed on the global cryptocurrency exchanges such as Huobi, Upbit, OKEX, and others.

Our notable portfolios include SK Telecom, Ground X by Kakao, and Carry Protocol while HAECHI AUDIT has conducted security audits for the world's top projects and enterprises.

Trusted by the industry leaders, we have been incubated by Samsung Electronics and awarded the Ethereum Foundation Grants and Ethereum Community Fund.

Contact : audit@haechi.io
Website : audit.haechi.io

01. Introduction

This report was written to provide a security audit for the Valor smart contract. HAECHI AUDIT conducted the audit focusing on whether Valor smart contract is designed and implemented in accordance with publicly released information and whether it has any security vulnerabilities.

The issues found are classified as CRITICAL, MAJOR, MINOR or TIPS according to their severity.

CRITICAL	Critical issues are security vulnerabilities that MUST be addressed in order to prevent widespread and massive damage.
MAJOR	Major issues contain security vulnerabilities or have faulty implementation issues and need to be fixed.
MINOR	Minor issues are some potential risks that require some degree of modification.
TIPS	Tips could help improve the code's usability and efficiency

HAECHI AUDIT advises addressing all the issues found in this report.

02. Summary

The code used for the audit can be found at GitHub (https://github.com/Valor-finance/Valor). The last commit for the code audited is at "b4501d84e60954b0d645e3e54aa3488f6c675221".

Issues

HAECHI AUDIT has 0 Critical Issues, 0 Major Issues, and 2 Minor Issue.

Severity	Issue	Status
MINOR	**ValorToken#delegateBySig() can be called with (nonce, expiry) which was not used to generate the (v, r, s).**	(Found v1.0)
MINOR	**ValorPool,VaultFlipToCake#notifyRewardAmount() can decrease rewardRate**	(Found v1.0) (Intended Behavior - v2.0)
NOTICE	**Non-audited, changeable external contract**	(Found v1.0)

03. Overview

Contracts Subject to Audit

- ValorMinter.sol
- ValorToken.sol
- ValorPool.sol
- ValorBNBPool.sol
- VaultCakeToCake.sol
- VaultFlipToFlip.sol
- VaultFlipToCake.sol

Roles

The Valor Smart contract has the following authorizations:

- **Owner**
- **Minter**
- **Keeper**
- **RewardDistribution**

The features accessible by each level of authorization is as follows:

Role	Functions
Owner	● ValorToken ○ mint ● VaultFlipToCake ○ setMinter ○ setRewardsToken ○ setRewardsDuration ○ recoverToken ● ValorMinter ○ transferValorOwner ○ setWithdrawalFee ○ setPerformanceFee ○ setWithdrawalFeeFreePeriod ○ setMinter ○ setValorPerProfitBNB ○ setValorPerValorBNBFlip

	○ setHelper ● VaultController ○ setMinter ○ recoverToken ● VaultFlipToFlip ○ recoverToken ● ValorPool ○ setRewardsToken ○ setHelper ○ recoverBEP20 ○ setRewardsDuration ● ValorBNBPool ○ setFlipToken ○ setMinter ○ setHelper
Minter	● ValorMinter ○ mintFor ○ mintForValorBNB
Keeper	● VaultController ○ setKeeper ● VaultFlipToFlip ○ harvest
RewardDistribution	● VaultFlipToCake ○ notifyRewardAmont ● ValorPool ○ notifyRewardAmount

Notice

- **Non-audited, changeable external contract**

  Haechi has found that Valor protocol used “StrategyHelper” contract in several parts for getting tvl, apy. Although this helper is used to get data, not writing data which is much safer. But this “helper” can be changed by the owner and this can lead to price manipulation. Also, these view functions which depend on

helper contracts, are weak to flash loan attacks. Do not trust these functions as a price table when building on top of this protocol.

If you are an end-user, this issue will not expose much issue/security risk to you. But if you build something on top of this code, be aware of the above issue.

04. Issues Found

MMINOR : ValorToken#delegateBySig() can be called with (nonce, expiry) which was not used to generate the (v, r, s). (Found - v1.0)

MINOR

Problem Statement

*ValorToken#delegateBySIg()*is designed to delegate the voting right with meta-transaction. In this case, the delegator generates (v, r, s) by signing (nonce,expiry) with its privateKey. Delegatee(or any other one who can call this function) will call delegateBySig() with these parameter. Then the function should verify if the (v,r,s) is signed by delegator, but in ValorToken, it does not check if the signer is the delegator. Which will result in delegating another address's voting power instead of the delegator if (nonce, expiry) is not the one that was signed.

Recommendation

Add one more parameter in delegateBySig() function that gets delegator's address and verify if ecrecovered address is delegator.

Disclaimer

- **This issue has been found on most well-known governance tokens, since these "delegate" parts are copy and pasted which results in almost the same codes being used over and over again.**
- **Although this looks very malicious, it is nearly impossible to find (nonce,expiry) that will result in making another user with voting power to delegate instead of the actual signer.**
- **If you are an end-user, this issue will not expose much issue/security risk to you. But if you are going to build on top of this code, such as opening a website to receive voting power using delegateBySig, or some kind of reward system contract that rewards users when delegate, be aware that this "delegateBySig" won't guarantee the signature is valid.**

MINOR : ValorPool,VaultFlipToCake#notifyRewardAmount() can decrease rewardRate (Found - v1.0) (Intended Behavior - v2.0)

MINOR

Problem Statement

notifyRewardAmount() does not check if the rewardRate decreases after notification.

Since it updates rate to be *(leftoverRate + notified reward)/duration*when previous reward is not finished, if admin keeps notifying with zero reward, it can lead to continuous decrease on reward rate,

Since this function is designed to be only called by admin, this error can only be done by admin.

Recommendation

Check if rewardRate increases after notifying reward.

Update

Valor team has confirmed that this is a intended behavior

05. Disclaimer

This report is not an advice on investment, nor does it guarantee adequacy of a business model and/or a bug-free code. This report should be used only to discuss known technical problems. The code may include problems on Ethereum that are not included in this report. It will be necessary to resolve addressed issues and conduct thorough tests to ensure the safety of the smart contract.