#HashEx
BLOCKCHAIN SECURITY

# Moonstarter

## smart contracts final audit report

December 2021

hashex.org

contact@hashex.org

# Contents

# 1. Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties, or other terms of any kind except as set out in this disclaimer, and HashEx hereby excludes all representations, warranties, conditions, and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of the use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed. HashEx owns all copyright rights to the text, images, photographs, and other content provided in the following document. When using or sharing partly or in full, third parties must provide a direct link to the original document mentioning the author (hashex.org).

# 2. Overview

HashEx was commissioned by the Moonstarter team to perform an audit of their smart contract. The audit was conducted between November 19 and November 23, 2021.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts
- Formally check the logic behind given smart contracts.

Information in this report should be used for understanding the risk exposure of smart contracts, and as a guide to improving the security posture of smart contracts by remediating the issues that were identified.

The code is available at @MoonStarter/PoolContractTemplate GitHub repository after e9acd4eb commit. Only the StakingV2 contract was in the scope of this audit. The contract was rechecked after the 9ed3a5f commit.

The updated code is deployed to the Binance Smart Chain:

0x31a1bfCc65aa22F015d4b0a9743fAD860386AA66 StakingV2.

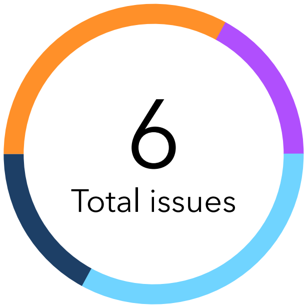
## 2.1 Summary

| Project name | Moonstarter |
|---|---|
| URL | https://moonstarter.net |
| Platform | Binance Smart Chain |
| Language | Solidity |

## 2.2 Contracts

| Name | Address |
|---|---|
| StakingV2 | `0x31a1bfCc65aa22F015d4b0a9743fAD860386AA66` |

# 3. Found issues



| | | |
|---|---|---|
| ● | High | 2 (33%) |
| ● | Medium | 1 (17%) |
| ● | Low | 2 (33%) |
| ● | Info | 1 (17%) |

## Cc5. StakingV2

| ID | Severity | Title | Status |
|---|---|---|---|
| Cc5l0e | ● High | Unrestricted coolDownDuration | ✓ Resolved |
| Cc5l10 | ● High | Gas block limit possibility | ✓ Resolved |
| Cc5l0f | ● Medium | End of the Cool Down period depends on value of coolDownDuration | ✓ Resolved |
| Cc5lea | ● Low | Excessive conditions | ✓ Acknowledged |
| Cc5leb | ● Low | Extra check | ✓ Resolved |
| Cc5l11 | ● Info | Unclear significance of cosmosDurationStatus | ✓ Resolved |

# 4. Contracts

## Cc5. StakingV2

### Overview

Staking contract without a rewarding model.

### Issues

#### Cc5I0e Unrestricted coolDownDuration ● High ✓ Resolved

The Owner is able to change coolDownDuration to an arbitrary value without any restriction.

#### Recommendation

It's recommended to set a range of appropriate values which the coolDownDuration can be changed within. Otherwise, the owner is able to halt all the unstake() function calls by setting the cooldown period in the distant future.

#### Cc5I10 Gas block limit possibility ● High ✓ Resolved

Removing an item from the array `_userList` can be a reason for the "exceeded block gas limit" error in case of a big amount of items within the array.

#### Recommendation

In case storing the list of users is necessary, consider using OpenZeppelin's EnumerableSet. It provides cheap operation for removing and reading items.

## Cc5l0f End of the Cool Down period depends on value of coolDownDuration ● Medium ✓ Resolved

It's considered an issue since the users' withdrawal availability strictly depends on it. When a user has set their coolDownInitTimestamp, a change of coolDownDuration by the Owner will concern the user.

### Recommendation

Instead of storing a timestamp when the user's cooldown starts, we suggest storing a timestamp when it ends.

## Cc5lea Excessive conditions ● Low ✓ Acknowledged

Excessive conditions in `getUserStakingData()` at L192,216,227. The second inequality is stricter than the first one.

## Cc5leb Extra check ● Low ✓ Resolved

In L172-173 there is a condition that has been already checked above in L167 in the function `_getCoolDownFinished`.

## Cc5l11 Unclear significance of cosmosDurationStatus ● Info ✓ Resolved

The meaning of the Cosmos Status is needed to be clarified since it's probably used out of audited scope.

### Team response

The cosmos duration status is the duration to have the cosmos state in our IDOs.

# 5. Conclusion

2 high severity issues were found during the initial audit and resolved after the recheck of the updated code. The contract is strongly dependent on the owner's account. If the owner's account is compromised, the staking contract may be severely broken. We recommend securing the owner account by putting it behind a Timelock contract and/or using a multi-sig.

This audit includes recommendations on code improvement.

The updated code is deployed to the Binance Smart Chain:

0x31a1bfCc65aa22F015d4b0a9743fAD860386AA66 StakingV2.

# Appendix A. Issues' severity classification

- **Critical.** Issues that may cause an unlimited loss of funds or entirely break the contract workflow.  Malicious code (including malicious modification of libraries) is also treated as a critical severity issue. These issues must be fixed before deployments or fixed in already running projects as soon as possible.
- **High.** Issues that may lead to a limited loss of funds, break interaction with users, or other contracts under specific conditions. Also, issues in a smart contract, that allow a privileged account the ability to steal or block other users' funds.
- **Medium.** Issues that do not lead to a loss of funds directly, but break the contract logic. May lead to failures in contracts operation.
- **Low.** Issues that are of a non-optimal code character, for instance, gas optimization tips, unused variables, errors in messages.
- **Informational.** Issues that do not impact the contract operation. Usually, informational severity issues are related to code best practices, e.g. style guide.

# Appendix B. List of examined issue types

- Business logic overview
- Functionality checks
- Following best practices
- Access control and authorization
- Reentrancy attacks
- Front-run attacks
- DoS with (unexpected) revert
- DoS with block gas limit
- Transaction-ordering dependence
- ERC/BEP and other standards violation
- Unchecked math
- Implicit visibility levels
- Excessive gas usage
- Timestamp dependence
- Forcibly sending ether to a contract
- Weak sources of randomness
- Shadowing state variables
- Usage of deprecated code

✉ contact@hashex.org

@hashex_manager

blog.hashex.org

linkedin

github

twitter

# HashEx

BLOCKCHAIN SECURITY