Alaska Gold Rush Security Review

March 20, 2024 - March 26, 2024

Date: March 26, 2024

Conducted by: **KeySecurity**
gkrastenov, Lead Security Researcher

Table of Contents

1 About KeySecurity

KeySecurity is a innovative Web3 security company that hires top-talented security researchers for your project. We have conducted over 40+ security reviews for various projects, collectively holding over $300,000,000 in TVL. For security audit inquiries, you can reach out to us on X or Telegram `@gkrastenov` or check our previous work `here`.

2 About Alaska Gold Rush

`Alaska Gold Rush` is the first WEB3 native game offering an open world with an exciting plot, NFTs, Play2Earn mechanics, and adventures within the Metaverse.

3 Disclaimer

Audits are a time, resource, and expertise bound effort where trained experts evaluate smart contracts using a combination of automated and manual techniques to identify as many vulnerabilities as possible. Audits can show the presence of vulnerabilities **but not their absence**.

4 Risk classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4.1 Impact

- **High** - leads to a significant loss of assets in the protocol or significantly harms a group of users.
- **Medium** - only a small amount of funds can be lost or a functionality of the protocol is affected.
- **Low** - any kind of unexpected behaviour that's not so critical.

4.2 Likelihood

- **High** - direct attack vector; the cost is relatively low to the amount of funds that can be lost.
- **Medium** - only conditionally incentivized attack vector, but still relatively likely.
- **Low** - too many or too unlikely assumptions; provides little or no incentive.

4.3 Actions required by severity level

- **Critical** - client **must** fix the issue.
- **High** - client **must** fix the issue.
- **Medium** - client **should** fix the issue.
- **Low** - client **could** fix the issue.

5 Executive summary

Overview

Project Name	Alaska Gold Rush
Repository	N/A, private codebase
Commit hash	N/A
Review Commit hash	N/A
Documentation	N/A
Methods	Manual review

Scope

ERC20CcipExtension
MultichainTransferBase
TokenTransferor
TokenWrapper
CaratBlocker

Timeline

March 20, 2024	Audit kick-off
March 26, 2024	Preliminary report
March 26, 2024	Mitigation review

Issues Found

Severity	**Count**
High	0
Medium	0
Low	1
Information	2
Total	**3**

6 System overview

The scope of this security audit consists of 5 smart contracts. The `ERC20CcipExtension` contract handles the burning of ERC20 tokens on the source blockchain and the minting of tokens on the destination blockchain. The `MultichainTransferBase` and `TokenTransferor` contracts are forks of the guide contracts in the CCIP documentation for sending arbitrary data and transferring tokens.

The `TokenWrapper` contract has one instance for all supported networks, as the main idea is to hold the total supply of tokens which will be transferred by the users.

The `CaratBlocke`r contract is used to generate signatures, which are then signed off-chain. These signatures are used to block and unblock `CARAT` tokens during the buying and selling process of CARAT tokens for GOLD.

7 Findings

7.1 Low

7.1.1 Wrong implementation of EIP-712

Severity: *Low*

Context: Global

Description: The hash structures `BLOCK_FUNCTION_HASH` and `UNBLOCK_FUNCTION_HASH` are incorrect implemented because they contain an additional argument bytes signature, which is not used during hashing. The only arguments that the structs should have are bytes32 operationId, uint256 chainId, address user, uint256 amount and uint256 blockDeadline. Everything else is not included during hashing.

Additionally, the hash structures and domain separator should use double apostrophes `"` instead of single ones `'` by definition of EIP-712.

Recommendation: Make the following changes:

```
"EIP712Domain(string name,string version,uint256 chainId,address verifyingContract)"

"blockCarat(bytes32 operationId_,uint256 chainId_,address user_,uint256 amount_,
    uint256 blockDeadline_)"

"unblockCarat(bytes32 operationId_,uint256 chainId_,address user_,uint256 amount_,
    uint256 blockDeadline_)"
```

Resolution and Client comment: Resolved.

7.2 Information

7.2.1 Lack of validation of receiver address

Severity: *Information*

Context: MultichainTransferBase

Description: The `MultichainTransferBase` lacks validation of the receiver address. Users may mistakenly set the receiver address to be equal to `address(this)` or `address(0)`. Most popular bridges have implemented checks for the receiver address to avoid sending tokens to the wrong address.

Recommendation: Add proper validation for the receiver address.

Resolution and Client comment: Resolved.

7.2.2 Redundant error

Severity: *Information*

Context: CaratBlocker

Description: The `CaratBlocker` contract contains redundant error `ZeroAddress()`.

Recommendation: Remove redundant error.

Resolution and Client comment: Resolved.