Quicksnap Audit Report

Dec 22, 2023

Table of Contents

Summary

This report has been prepared for Quicksnap smart contract, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Overview

Project Summary

Project Name	**Quicksnap**
Codebase	**https://github.com/quicksnap-io/contracts/**
Commit	**4b61cb732f8e574b3b753fd95c7b37ab14f5fb7d**
Language	**Solidity**

Audit Summary

Delivery Date	**Dec 22, 2023**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**3**

[WP-S1] Consider adding a `version` check to avoid mistakenly setting the same `_merkleRoot` twice.

Issue Description

The current implementation would allow the same calldata to `updateMerkleRoot()` multiple times, which will result in the rewards for one update/version to be claimed multiple times.

While this requires misbehavior of the owner, given the severe impact, it's recommended to add a check to disallow such a mistake from happening.

https://github.com/quicksnap-io/contracts/blob/c684f1d79b506ea5813e08ffbbb4657a82c1ce9e/contracts/MultiMerkleStash.sol#L69-L77

```
69  function updateMerkleRoot(address token, bytes32 _merkleRoot) public onlyOwner {
70
71      // Increment the update (simulates the clearing of the claimedBitMap)
72      update[token] += 1;
73      // Set the new merkle root
74      merkleRoot[token] = _merkleRoot;
75
76      emit MerkleRootUpdated(token, _merkleRoot, update[token]);
77  }
```

Recommendation

```
69  function updateMerkleRoot(address token, bytes32 _merkleRoot, uint256 version)
    public onlyOwner {
70      // Increment the update (simulates the clearing of the claimedBitMap)
71      update[token] += 1;
72
73      require(update[token] == version, "version mismatch");
74
75      // Set the new merkle root
76      merkleRoot[token] = _merkleRoot;
77
78      emit MerkleRootUpdated(token, _merkleRoot, update[token]);
79  }
```

Status

✓ Fixed

[WP-L2] `claimMulti()` Wrong implementation of `claims.length` limit.

Low

Issue Description

https://github.com/quicksnap-io/contracts/blob/c684f1d79b506ea5813e08ffbbb4657a82c1ce9e/contracts/MultiMerkleStash.sol#L60-L65

```
60      function claimMulti(address account, claimParam[] calldata claims) external {
61          require(claims.length < 20, "Can't claim more than 20 tokens at once");
62          for (uint256 i = 0; i < claims.length; i++) {
63              claim(claims[i].token, claims[i].index, account, claims[i].amount,
    claims[i].merkleProof);
64          }
65      }
```

Based on the error message and context, it is inferred that the expected implementation only allows a maximum of ”claiming 20 tokens at once”.

However, in the current implementation, the maximum is ”claiming 19 tokens at once”.

Recommendation

```
60      function claimMulti(address account, claimParam[] calldata claims) external {
61          require(claims.length <= 20, "Can't claim more than 20 tokens at once");
62          for (uint256 i = 0; i < claims.length; i++) {
63              claim(claims[i].token, claims[i].index, account, claims[i].amount,
    claims[i].merkleProof);
64          }
65      }
```

Status

✓ Fixed

[WP-L3] Missing error messages in require statements

Low

Issue Description

https://github.com/quicksnap-io/contracts/blob/c684f1d79b506ea5813e08ffbbb4657a82c1ce9e/contracts/BribeV3Snapshot.sol#L38-L49

```
38  function add_reward_amount(string memory proposal, uint256 option, address
    reward_token, uint256 amount, uint256 startTime, uint256 endTime) nonReentrant
    external {
39      require(reward_token != address(0));
40      require(amount > 0, "no reward to add");
41
42      uint256 fee = calculate_fee(amount);
43      amount = amount - fee;
44
45      IERC20(reward_token).safeTransferFrom(msg.sender, feeAddress, fee);
46      IERC20(reward_token).safeTransferFrom(msg.sender, distributionAddress,
    amount);
47
48      emit Bribe(block.timestamp, msg.sender, proposal, option, reward_token,
    amount, startTime, endTime);
49  }
```

Status

✓ Fixed

Appendix

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by WatchPug; however, WatchPug does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Smart Contract technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.