HALBORN

Pontem Network - Liquidswap Update 0.3.1

Move Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **September 30th, 2022 - October 5th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	09/30/2022	Lukasz Mikula
0.2	Draft Version	10/05/2022	Lukasz Mikula
0.3	Draft Review	10/06/2022	Gabi Urrutia
1.0	Remediation Plan	10/10/2022	Lukasz Mikula
1.1	Remediation Plan Review	10/10/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Luis Quispe Gonzales	Halborn	Luis.QuispeGonzales@halborn.com
Lukasz Mikula	Halborn	Lukasz.Mikula@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Pontem Network engaged Halborn to conduct a security audit on their smart contracts beginning on September 30th, 2022 and ending on October 5th, 2022 . The security assessment was scoped to the smart contracts provided in the GitHub repository Liquidswap, commit hashes and further details can be found in the Scope section of this report.

1.2 AUDIT SUMMARY

The team at Halborn was provided five days for the engagement and assigned one full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn found that the contract followed secure development best practices, resulting in low findings with negligible impact on security.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual review of the code and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of the smart contract audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of smart contracts and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into the architecture, purpose, and use of the platform.
- Smart contract manual code review and walk-through to identify any logic issue.
- Thorough assessment of safety and usage of critical Rust variables and functions in scope that could lead to arithmetic related vulnerabilities.
- Test coverage review (aptos move test).

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.

1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

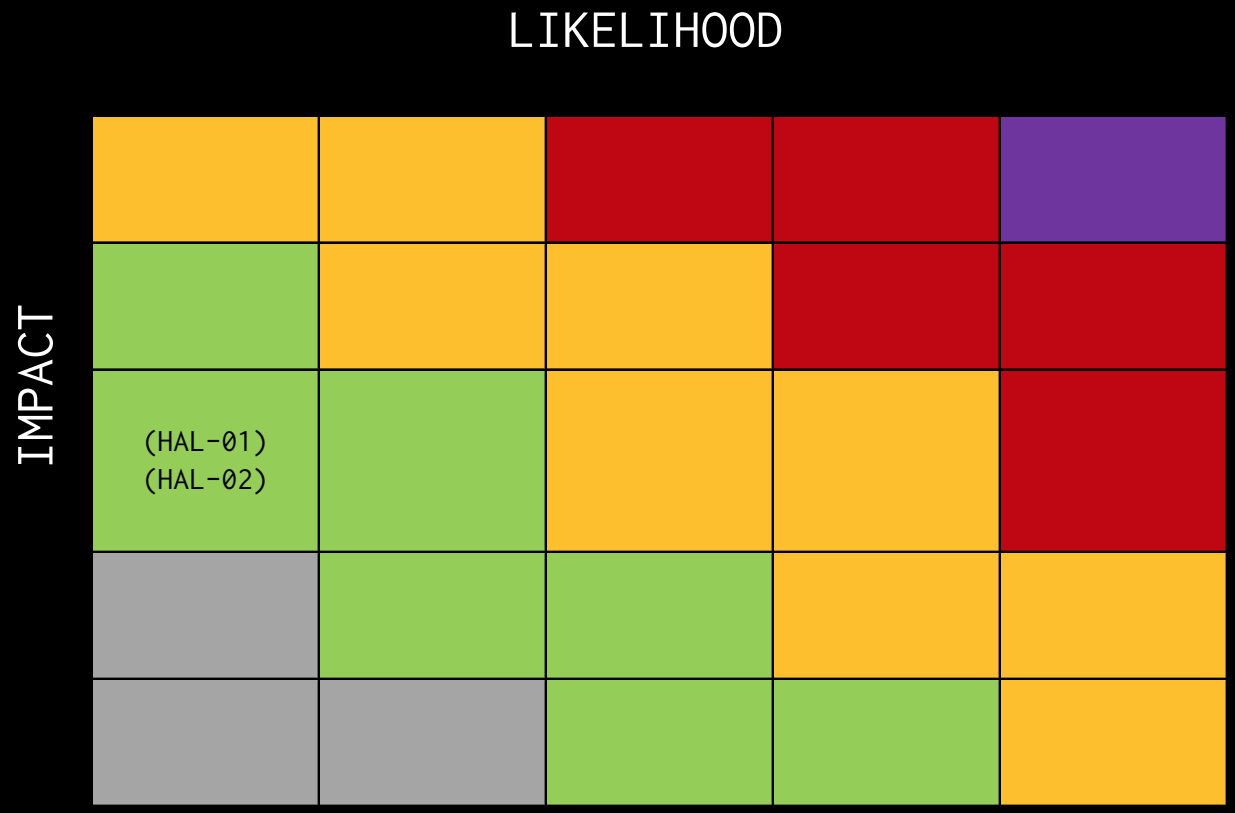
1.4 SCOPE

1. Move Smart Contract
   (a) Repository: liquidswap
   (b) Commit ID: 13f54d85c253e3c8c9c610703b38db40b580c66f
   (c) Contracts in scope:
      - global_config.move
      - liquidity_pool.move

Out-of-scope: External libraries and financial related attacks.

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	2	0



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
PRIVILEGED ADDRESS CAN BE TRANSFERRED WITHOUT CONFIRMATION	Low	RISK ACCEPTED
NO LIMITATIONS OF MAXIMAL DAO FEE	Low	RISK ACCEPTED

FINDINGS & TECH DETAILS

3.1 (HAL-01) PRIVILEGED ADDRESS CAN BE TRANSFERRED WITHOUT CONFIRMATION - LOW

Description:

Incorrect use of the ownership transfer functions: set_dao_admin, set_emergency_admin and set_fee_admin could set the OWNER to an invalid address, unintentionally losing control of the contract, which cannot be undone in any way. Currently, the OWNER of the contracts can change their address using the aforementioned function in a single transaction and without confirmation from the new address.

Code Location:

Below snippets show tree functions: set_dao_admin, set_emergency_admin and set_fee_admin in respective order.

Listing 1: sources/swap/global_config.move (Line 94)

```
88        public entry fun set_dao_admin(dao_admin: &signer, new_addr
↳ : address) acquires GlobalConfig {
89            assert!(exists<GlobalConfig>(@liquidswap),
↳ ERR_CONFIG_DOES_NOT_EXIST);
90
91            let config = borrow_global_mut<GlobalConfig>(@liquidswap);
92            assert!(config.dao_admin_address == signer::address_of(
↳ dao_admin), ERR_NOT_ADMIN);
93
94            config.dao_admin_address = new_addr;
95        }
```

Listing 2: sources/swap/global_config.move (Line 112)

```
106     public entry fun set_emergency_admin(emergency_admin: &signer,
↳  new_addr: address) acquires GlobalConfig {
107           assert!(exists<GlobalConfig>(@liquidswap),
↳ ERR_CONFIG_DOES_NOT_EXIST);
```

```
108
109         let config = borrow_global_mut<GlobalConfig>(@liquidswap);
110         assert!(config.emergency_admin_address == signer::
 ↳ address_of(emergency_admin), ERR_NOT_ADMIN);
111
112         config.emergency_admin_address = new_addr;
113     }
```

Listing 3: sources/swap/global_config.move (Line 130)

```
124     /// Set fee admin account.
125     public entry fun set_fee_admin(fee_admin: &signer, new_addr:
 ↳ address) acquires GlobalConfig {
126         assert!(exists<GlobalConfig>(@liquidswap),
 ↳ ERR_CONFIG_DOES_NOT_EXIST);
127
128         let config = borrow_global_mut<GlobalConfig>(@liquidswap);
129         assert!(config.fee_admin_address == signer::address_of(
 ↳ fee_admin), ERR_NOT_ADMIN);
130
131         config.fee_admin_address = new_addr;
132     }
```

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

Each of the aforementioned functions should follow a two-step process, splitting into set_owner and accept_owner functions. The latter requires the transfer to be completed by the recipient, effectively protecting the contract against potential typing errors compared to OWNER's one-step transfer mechanisms.

Remediation plan:

RISK ACCEPTED: The \client team accepted the risk of this finding.

3.2 (HAL-02) NO LIMITATIONS OF MAXIMAL DAO FEE - LOW

Description:

The MAX_DAO_FEE has no upper percentage limit. That means that if a malicious actor has access to the fee_admin role, there is a possibility to increase that fee to 100%, which could be used to perform further attacks related to draining increased funds from the DAO. While this alone is not a direct security threat, it could be a favorable circumstance if an unauthorized actor has access to the fee_admin role.

Code Location:

Listing 4: sources/swap/global_config.move (Line 38)

```
34        /// Minimum value of dao fee, 0%
35    const MIN_DAO_FEE: u64 = 0;
36
37    /// Maximum value of dao fee, 100%
38    const MAX_DAO_FEE: u64 = 100;
```

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

It is recommended to limit the MAX_DAO_FEE between stricter bands, e.g., a maximum of 20%, depending on business / tokenomics needs..

Remediation plan:

RISK ACCEPTED: The \client team accepted the risk of this finding.

THANK YOU FOR CHOOSING

// HALBORN