Smart Contract Security Audit Report

[2021]

Table Of Contents

1 Executive Summary

2 Audit Methodology

3 Project Overview

3.1 Project Introduction

3.2 Vulnerability Information

4 Code Overview

4.1 Contracts Description

4.2 Visibility Description

4.3 Vulnerability Summary

5 Audit Result

6 Statement

1 Executive Summary

On 2021.12.06, the SlowMist security team received the PortFinance team's security audit application for Port Finance Lending, developed the audit plan according to the agreement of both parties and the characteristics of the project, and finally issued the security audit report.

The SlowMist security team adopts the strategy of "white box lead, black, grey box assists" to conduct a complete security test on the project in the way closest to the real attack.

The test method information:

Test method	Description
Black box testing	Conduct security tests from an attacker's perspective externally.
Grey box testing	Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses.
White box testing	Based on the open source code, non-open source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc.

The vulnerability severity level information:

Level	Description
Critical	Critical severity vulnerabilities will have a significant impact on the security of the DeFi project, and it is strongly recommended to fix the critical vulnerabilities.
High	High severity vulnerabilities will affect the normal operation of the DeFi project. It is strongly recommended to fix high-risk vulnerabilities.
Medium	Medium severity vulnerability will affect the operation of the DeFi project. It is recommended to fix medium-risk vulnerabilities.
Low	Low severity vulnerabilities may affect the operation of the DeFi project in certain scenarios. It is suggested that the project party should evaluate and consider whether these vulnerabilities need to be fixed.
Weakness	There are safety risks theoretically, but it is extremely difficult to reproduce in engineering.

Level	Description
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for smart contract includes two steps:

Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.

Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Reentrancy Vulnerability
- Replay Vulnerability
- Reordering Vulnerability
- Denial of Service Vulnerability
- Transaction Ordering Dependence Vulnerability
- Race Conditions Vulnerability
- Authority Control Vulnerability
- Integer Overflow and Underflow Vulnerability
- TimeStamp Dependence Vulnerability
- Unsafe External Call Audit
- Design Logic Audit
- Scoping and Declarations Audit
- Forged account attack Audit

3 Project Overview

3.1 Project Introduction

Port Finance is a lending protocol that aims to provide an entire suite of fixed income products including variable rate lending, fixed rate lending and interest rate swaps.

3.2 Vulnerability Information

The following is the status of the vulnerabilities found in this audit:

NO	Title	Category	Level	Status
N1	Anyone can initialize a staking pool	Forged account attack	Suggestion	Ignored
N2	Missing to check oracle status	Arithmetic Accuracy Deviation Vulnerability	Medium	Fixed
N3	Rounding decimal	Arithmetic Accuracy Deviation Vulnerability	Suggestion	Ignored
N4	Anyone can initialize a lending market	Forged account attack	Suggestion	Ignored
N5	Missing check `host_fee_receiver_info`	Forged account attack	Suggestion	Ignored
N6	Reentrancy risk	Reentrancy Vulnerability	Suggestion	Ignored

4 Code Overview

4.1 Contracts Description

The main network address of the contract is as follows:

The code was not deployed to the mainnet.

Audit version:

https://github.com/port-finance/lending

879064a3f09ee3433b235d1a2623cc08c752e34a

4.2 Visibility Description

The SlowMist Security team analyzed the visibility of major contracts during the audit, the result as follows:

staking				
Function Name	Lamport check	Owner/Key check	Signer check	ProgramID check
process_change_owner	-	2/2	x	o
process_change_admin	-	3/3	o	o
process_change_duration	-	2/2	x	o
process_change_reward_supply	-	6/8	o	o
process_init_staking_pool	o	8/8	o	o
process_create_stake_account	o	3/4	x	o
process_deposit	-	4/4	o	o
process_withdraw	-	4/4	o	o
process_claim_reward	-	8/8	o	o

staking				
process_update_earliest_reward_claim_time	-	2/2	o	o

lending				
Function Name	Lamport check	Owner/Key check	Signer check	ProgramID check
process_init_lending_market	o	2/3	x	o
process_set_lending_market_owner	o	2/2	o	o
process_init_reserve	o	11/16	o	o
process_refresh_reserve	-	3/3	x	o
process_deposit_reserve_liquidity	-	9/10	x	o
process_redeem_reserve_collateral	-	9/10	x	o
process_init_obligation	o	4/5	o	o
process_refresh_obligation	-	4/4	x	o
process_deposit_obligation_collateral	-	12/13	x	o
process_withdraw_obligation_collateral	-	9/9	o	o
process_borrow_obligation_liquidity	-	10/10	o	o
process_repay_obligation_liquidity	-	7/8	x	o
process_liquidate_obligation	-	14/15	x	o

lending				
process_flash_loan	-	7/9	x	o
process_deposit_reserve_liquidity_and_obligation_collateral	-	14/16	x	o
process_update_reserve	o	6/6	o	o
process_withdraw_fee	o	7/8	o	o

4.3 Vulnerability Summary

[N1] [Suggestion] Anyone can initialize a staking pool

Category: Forged account attack

Content

Missing check the signer's key, anyone can initialize a instance with faking info.

- Code location: lending/staking/program/src/processor.rs

```
fn process_init_staking_pool
```

Solution

Only administrator can invoke initialize function.

Status

Ignored; This is a feature.

[N2] [Medium] Missing to check oracle status

Category: Arithmetic Accuracy Deviation Vulnerability

Content

- Code location: lending/token-lending/program/src/processor.rs

```
fn get_pyth_price(pyth_price_info: &AccountInfo, clock: &Clock) -> Result<Decimal,
ProgramError> {









    const STALE_AFTER_SLOTS_ELAPSED: u64 = 10;

    let pyth_price_data = pyth_price_info.try_borrow_data()?;
    let pyth_price = pyth::load::<pyth::Price>(&pyth_price_data)
        .map_err(|_| ProgramError::InvalidAccountData)?;

    if pyth_price.ptype != pyth::PriceType::Price {
        msg!("Oracle price type is invalid");
        return Err(LendingError::InvalidOracleConfig.into());
    }
//SlowMist//...skip code...
}
```

Solution

`pyth_price.agg.status` should be equal to `pyth::PriceStatus::Trading`.

Status

Fixed; patch: https://github.com/port-finance/variable-rate-lending/pull/73

[N3] [Suggestion] Rounding decimal

Category: Arithmetic Accuracy Deviation Vulnerability

Content

`try_round_u64` will cause the result to be greater or less than expected, and `borrow_fee/host_fee` will be less than expected in this place.

- Code location: lending/token-lending/program/src/state/reserve.rs

```
fn calculate_fees(
    &self,
    amount: Decimal,
    fee_wad: u64,
    fee_calculation: FeeCalculation,
) -> Result<(u64, u64), ProgramError> {
//SlowMist//...skip code...

        let borrow_fee = borrow_fee_decimal.try_round_u64()?;
        let host_fee = if need_to_assess_host_fee {
            borrow_fee_decimal
                .try_mul(host_fee_rate)?
                .try_round_u64()?
                .max(1u64)
        } else {
            0
        };

        Ok((borrow_fee, host_fee))
    } else {
        Ok((0, 0))
    }
}
```

Solution

Ceil fee.

Status

Ignored; This is acceptable.

[N4] [Suggestion] Anyone can initialize a lending market

Category: Forged account attack

Content

- Code location: lending/token-lending/program/src/processor.rs

```
fn process_init_lending_market
```

```
fn process_init_reserve

fn process_init_obligation
```

Since `lending_market.owner/obligation.lending_market/reserve.lending_market` may missing check in related logical, the initialize permission should be limited.

Solution

Only administrator can invoke initialize function.

Status

Ignored; This is a feature.

[N5] [Suggestion] Missing check `host_fee_receiver_info`

Category: Forged account attack

Content

Missing check `host_fee_receiver_info`, users can steal `host_fee` by specifying `host_fee_receiver_info` to their own addresses.

- Code location: lending/token-lending/program/src/processor.rs

```
fn process_flash_loan(
    program_id: &Pubkey,
    liquidity_amount: u64,
    accounts: &[AccountInfo],
) -> ProgramResult {
//SlowMist// ...skip code...

    let host_fee_receiver_info = next_account_info(account_info_iter)?;
```

Solution

Check `host_fee_receiver_info` key.

Status

Ignored; This is a feature.

[N6] [Suggestion] Reentrancy risk

Category: Reentrancy Vulnerability

Content

After the attacker calls `process_flash_loan` to borrow, he uses the borrowed funds to deposit to the contract. In this way, the flash loan will detect that the funds have been returned during the repayment check, which leads to the success of the flash loan, but the funds are not actually returned. The attacker can withdraw this deposit at any time, thereby stealing all the funds in the fund pool.

- Code location: lending/token-lending/program/src/processor.rs

```
fn process_flash_loan
```

Solution

Prohibition of contract re-entry.

Status

Ignored; Since in Solana Reentrancy is forbidden except for direct recursion.

Refer to solana documentation: https://docs.solana.com/developing/programming-model/calling-between-programs#reentrancy

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002112230002	SlowMist Security Team	2021.12.06 - 2021.12.23	Passed

Summary conclusion: The SlowMist security team use a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 1 medium risk, 5 suggestion vulnerabilities. And 5 suggestion vulnerabilities

were ignored; All other findings were fixed. The code was not deployed to the mainnet.

6 Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.

For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

Official Website

www.slowmist.com

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist