Smart Contract Security Audit Report

[2021]

Solyard Finance

Table Of Contents

1 Executive Summary

On 2021.07.12, the SlowMist security team received the SolYard team's security audit application for SolYard, 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
- Short Address 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

3 Project Overview

3.1 Project Introduction

SolYard is a new Yield Farming Aggregator for the Solana Ecosystem.

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	Invoke_signed cross-contract call does not capture the return value	Design Logic Audit	Suggestion	Ignored
N3	Not check whether lamport is sufficient	Design Logic Audit	Low	Fixed
N4	Code redundancy	Others	Suggestion	Fixed
N5	The authenticity of the system account is not checked	Authority Control Vulnerability	Suggestion	Ignored
N6	The input parameters are not fully checked	Authority Control Vulnerability	High	Fixed
N7	Trustlessness absence	Authority Control Vulnerability	Low	Confirmed

4 Code Overview

4.1 Contracts Description

The main network address of the contract is as follows:

https://explorer.solana.com/address/FGJzGGoxpbxv3DPxABykx5KFG91uT2agbNmNsexds6y4

Audit version:

https://github.com/SolyardFinance/Vaults-program2

Commit:45d69c24a4731443df4ed5556d4bb220b907d0f9

Review commit:4b976055a131d48df817ba4e03ab5987211bd43c

4.2 Visibility Description

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

entrypoint				
Function Name	Lamport check	Account check	Signer check	Program_id check
process_initialize_farm_v3_pool	x	2/12	x	x
process_deposit_raydium	x	15/24	x	x
process_withdraw_raydium	x	14/23	x	x
process_deposit_raydium_v4	x	17/22	x	x
process_withdraw_raydium_v4	x	16/21	x	x
process_deposit_ray	x	9/19	x	x
process_withdraw_ray	x	6/18	x	x
process	-	-	-	-
process_with_constraints	-	-	-	-

process_initialize_farm_v3_pool		
index	Account Name	Check

process_initialize_farm_v3_pool		
0	token_program_id	x
1	farm_info	o
2	authority_info	o
3	pool_farm_address	x
4	pool_lp_address	x
5	pool_fee_lp_address	x
6	clock_info	x
7	pool_ray_address	x
8	pool_id	x
9	pool_user_info	x
10	pool_other_address	x
11	pool_ray_info	x

process_deposit_raydium		
index	Account Name	Check
0	farm_info	o
1	authority_info	o
2	user_pool	-
3	user_farm_address	o
4	user_lp_address	o

process_deposit_raydium		
5	pool_lp_address	o
6	user_transfer_authority_info	x
7	token_program_info	o
8	pool_farm_address	o
9	clock_info	x
10	pool_fee_lp_account	o
11	pool_ray_account	x
12	user_ray_address	o
13	raydium_auth	x
14	raydium_lp_address	x
15	raydium_ray_address	x
16	pool_raydium_info_address	o
17	raydium_pool_id	o
18	raydium_token_program_id	x
19	ray_pool_id	x
20	ray_auth	x
21	ray_lp_address	x
22	ray_ray_address	x
23	pool_ray_info_address	x

process_withdraw_raydium		
index	Account Name	Check
0	farm_info	o
1	authority_info	o
2	user_pool	x
3	user_farm_address	o
4	user_lp_address	o
5	pool_lp_address	o
6	token_program_info	o
7	pool_farm_address	o
8	clock_info	x
9	pool_fee_lp_address	o
10	pool_ray_account	x
11	user_ray_address	o
12	raydium_auth	x
13	raydium_lp_address	x
14	raydium_ray_address	x
15	pool_raydium_info_address	o
16	raydium_pool_id	o
17	raydium_token_program_id	x
18	ray_pool_id	x

process_withdraw_raydium		
19	ray_auth	x
20	ray_lp_address	x
21	ray_ray_address	x
22	pool_ray_info_address	x

process_deposit_raydium_v4		
index	Account Name	Check
0	farm_info	o
1	authority_info	o
2	user_pool	x
3	user_farm_address	o
4	user_lp_address	o
5	pool_lp_address	o
6	user_transfer_authority_info	x
7	token_program_info	o
8	pool_farm_address	o
9	clock_info	x
10	pool_fee_lp_account	o
11	pool_ray_account	x
12	user_ray_address	o

process_deposit_raydium_v4		
13	raydium_auth	x
14	raydium_lp_address	x
15	raydium_ray_address	x
16	pool_raydium_info_address	o
17	raydium_pool_id	o
18	raydium_token_program_id	x
19	pool_other_address	o
20	raydium_other_address	x
21	user_other_address	o

process_withdraw_raydium_v4		
index	Account Name	Check
0	farm_info	o
1	authority_info	o
2	user_pool	x
3	user_farm_address	o
4	user_lp_address	o
5	pool_lp_address	o
6	token_program_info	o
7	pool_farm_address	o

process_withdraw_raydium_v4		
8	clock_info	x
9	pool_fee_lp_address	o
10	pool_ray_account	x
11	user_ray_address	o
12	raydium_auth	x
13	raydium_lp_address	x
14	raydium_ray_address	x
15	pool_raydium_info_address	o
16	raydium_pool_id	o
17	raydium_token_program_id	x
18	pool_other_address	o
19	raydium_other_address	x
20	user_other_address	o

process_deposit_ray		
index	Account Name	Check
0	farm_info	x
1	authority_info	x
2	user_pool	x
3	user_farm_address	o

process_deposit_ray		
4	user_lp_address	o
5	pool_lp_address	x
6	user_transfer_authority_info	x
7	token_program_info	o
8	pool_farm_address	x
9	clock_info	x
10	pool_fee_lp_account	x
11	pool_ray_account	x
12	user_ray_address	o
13	raydium_auth	x
14	raydium_lp_address	x
15	raydium_ray_address	x
16	pool_raydium_info_address	x
17	raydium_pool_id	o
18	raydium_token_program_id	o

process_withdraw_ray		
index	Account Name	Check
0	farm_info	x
1	authority_info	x

process_withdraw_ray		
2	user_pool	x
3	user_farm_address	o
4	user_lp_address	o
5	pool_lp_address	x
6	token_program_info	o
7	pool_farm_address	x
8	clock_info	x
9	pool_fee_lp_address	x
10	pool_ray_account	x
11	user_ray_address	o
12	raydium_auth	o
13	raydium_lp_address	x
14	raydium_ray_address	x
15	pool_raydium_info_address	x
16	raydium_pool_id	o
17	raydium_token_program_id	x

4.3 Vulnerability Summary

[N1] [Suggestion] Invoke_signed cross-contract call does not capture the return value

Category: Design Logic Audit

Content

Code location: vaultsv3/program/src/processor.rs

`invoke_signed` call in the `token_transfer` function;

`invoke_signed` call in `deposit_raydium` function;

`invoke_signed` call in `withdraw_raydium` function;

`invoke_signed` call in `deposit_raydium_v4` function;

`invoke_signed` call in `withdraw_raydium_v4` function;

invoke_signed cross-contract call does not capture the return value.

Solution

Status

Ignored; The program will catch ProgramError automatically.

[N3] [Low] Not check whether lamport is sufficient

Category: Design Logic Audit

Content

`process_initialize_farm_v3_pool` do not check whether lamport is sufficient;

`process_deposit_raydium` do not check whether the lamport is sufficient;

`process_withdraw_raydium` do not check whether the lamport is sufficient;

`process_deposit_raydium_v4` do not check whether the lamport is sufficient;

`process_withdraw_raydium_v4` do not check whether the lamport is sufficient;

`process_deposit_ray` do not check whether the lamport is sufficient;

`process_withdraw_ray` do not check whether the lamport is sufficient;

Solution

Use `rent.is_exempt` to compare lamports and data length

Status

Fixed

[N4] [Suggestion] Code redundancy

Category: Others

Content

Code location: vaultsv3/program/src/processor.rs

`process_withdraw_raydium` function

`process_withdraw_raydium_v4` function

`process_deposit_ray` function

`process_withdraw_ray` function

```
if pending> pool_farm_account.amount{
     pending = pool_farm_account.amount;
}
if pending> 0 && pool_farm_account.amount >= pending {
```

`pool_farm_account.amount >= pending` is always established

Solution

Can be deleted

Status

Fixed

[N5] [Suggestion] The authenticity of the system account is not checked

Category: Authority Control Vulnerability

Content

The attacker may use maliciously constructed fake accounts to replace the system clock

`SysvarClock11111111111111111111111111111111`

Solution

This vulnerability cannot be successfully exploited at present. It was found that Clock::from_account_info returned an error in a simulated attack:

```
Error: failed to send transaction: Transaction simulation failed: Error processing
Instruction 0: invalid program argument
```

But as a weakness, it is recommended to enhance it.

Status

Ignored; It is not necessary to change the code.

[N6] [High] The input parameters are not fully checked

Category: Authority Control Vulnerability

Content

`process_initialize_farm_v3_pool` does not fully check the `owner`, `key` or `is_signer` of the input account;

`process_deposit_raydium` does not fully check the `owner`, `key` or `is_signer` of the input account;

`process_withdraw_raydium` does not fully check the `owner`, `key` or `is_signer` of the input account;

`process_deposit_raydium_v4` does not fully check the `owner`, `key` or `is_signer` of the input account;

`process_withdraw_raydium_v4` does not fully check the `owner`, `key` or `is_signer` of the input account;

`process_deposit_ray` does not fully check the `owner`, `key` or `is_signer` of the input account;

`process_withdraw_ray` does not fully check the `owner`, `key` or `is_signer` of the input account;

An attacker may use a maliciously constructed account as an input parameter to make malicious calls to the contract.

For details, please refer to section 4.2 of this report.

Solution

Use the initial account, or compare the input account with the initial configuration account.

Status

Fixed; Verify most accounts in the latest version, and parts of accounts verify in Radium program instead of verify in Solyard.

[N7] [Low] Trustlessness absence

Category: Authority Control Vulnerability

Content

The mainnet program account is upgradeable, the project party can upgrade the main network contract code and change the logic. Users need to have enough trust in the project party.

Solution

Status

Confirmed; It is the feature of Solana.

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0x002107230001	SlowMist Security Team	2021.07.12 - 2021.07.23	Low Risk

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 high risk, 2 low risk, 3 suggestion vulnerabilities. And 1 low risk vulnerabilities were confirmed and being fixed; 2 suggestion vulnerabilities were ignored; All other findings were fixed.

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