BLOCKSEC

Security Audit
Report for 1LBP program of 1Intro

Date: April 29th, 2024 **Version:** 1.0

Contact: contact@blocksec.com

Contents

Report Manifest

Item	Description
Client	1Intro
Target	1LBP program of 1Intro

Version History

Version	Date	Description
1.0	April 29th, 2024	First release

Signature

About BlockSec BlockSec focuses on the security of the blockchain ecosystem and collaborates with leading DeFi projects to secure their products. BlockSec is founded by top-notch security researchers and experienced experts from both academia and industry. They have published multiple blockchain security papers in prestigious conferences, reported several zero-day attacks of DeFi applications, and successfully protected digital assets that are worth more than 14 million dollars by blocking multiple attacks. They can be reached at Email, Twitter and Medium.

Chapter 1 Introduction

1.1 About Target Contracts

Information	Description
Type	Smart Contract
Language	Rust
Approach	Semi-automatic and manual verification

The focus of this audit is the `programs/one-intro` within the 1LBP program of 1Intro [1]. Please note that other external dependencies in the repository, including the solana development framework `Anchor` [2], are considered reliable in terms of both functionality and security, these files are not included in the scope of the audit.

The auditing process is iterative. Specifically, we would audit the commits that fix the discovered issues. If there are new issues, we will continue this process. The commit SHA values during the audit are shown in the following table. Our audit report is responsible for the code in the initial version (`Version 1`), as well as new code (in the following versions) to fix issues in the audit report.

Project	Version	Commit Hash
1LBP program of 1Intro	`Version 1`	`696236080ab3926d88b5760e9c92cd8192cb43da`
	`Version 2`	`317225207312e958a227172186a169e923890302`

1.2 Disclaimer

This audit report does not constitute investment advice or a personal recommendation. It does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Any entity should not rely on this report in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset.

This audit report is not an endorsement of any particular project or team, and the report does not guarantee the security of any particular project. This audit does not give any warranties on discovering all security issues of the smart contracts, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues. As one audit cannot be considered comprehensive, we always recommend proceeding with independent audits and a public bug bounty program to ensure the security of smart contracts.

The scope of this audit is limited to the code mentioned in Section 1.1. Unless explicitly specified, the security of the language itself (e.g., the solidity language), the underlying compiling toolchain and the computing infrastructure are out of the scope.

[1] https://github.com/1intro/1intro-programs

[2] https://www.anchor-lang.com/

1.3 Procedure of Auditing

We perform the audit according to the following procedure.

- **Vulnerability Detection** We first scan smart contracts with automatic code analyzers, and then manually verify (reject or confirm) the issues reported by them.
- **Semantic Analysis** We study the business logic of smart contracts and conduct further investigation on the possible vulnerabilities using an automatic fuzzing tool (developed by our research team). We also manually analyze possible attack scenarios with independent auditors to cross-check the result.
- **Recommendation** We provide some useful advice to developers from the perspective of good programming practice, including gas optimization, code style, and etc.

We show the main concrete checkpoints in the following.

1.3.1 Software Security

* Reentrancy
* DoS
* Access control
* Data handling and data flow
* Exception handling
* Untrusted external call and control flow
* Initialization consistency
* Events operation
* Error-prone randomness
* Improper use of the proxy system

1.3.2 DeFi Security

* Semantic consistency
* Functionality consistency
* Permission management
* Business logic
* Token operation
* Emergency mechanism
* Oracle security
* Whitelist and blacklist
* Economic impact
* Batch transfer

1.3.3 NFT Security

* Duplicated item
* Verification of the token receiver
* Off-chain metadata security

1.3.4 Additional Recommendation

* Gas optimization
* Code quality and style

Note *The previous checkpoints are the main ones. We may use more checkpoints during the auditing process according to the functionality of the project.*

1.4 Security Model

To evaluate the risk, we follow the standards or suggestions that are widely adopted by both industry and academy, including OWASP Risk Rating Methodology [3] and Common Weakness Enumeration [4]. The overall *severity* of the risk is determined by *likelihood* and *impact*. Specifically, likelihood is used to estimate how likely a particular vulnerability can be uncovered and exploited by an attacker, while impact is used to measure the consequences of a successful exploit.

In this report, both likelihood and impact are categorized into two ratings, i.e., *high* and *low* respectively, and their combinations are shown in Table 1.1.

Table 1.1: Vulnerability Severity Classification

Impact \ Likelihood	*High*	*Low*
High	High	Medium
Low	Medium	Low

Accordingly, the severity measured in this report are classified into three categories: **High**, **Medium**, **Low**. For the sake of completeness, **Undetermined** is also used to cover circumstances when the risk cannot be well determined.

Furthermore, the status of a discovered item will fall into one of the following four categories:

- **Undetermined** No response yet.
- **Acknowledged** The item has been received by the client, but not confirmed yet.
- **Confirmed** The item has been recognized by the client, but not fixed yet.
- **Fixed** The item has been confirmed and fixed by the client.

[3] https://owasp.org/www-community/OWASP_Risk_Rating_Methodology

[4] https://cwe.mitre.org/

Chapter 2 Findings

In total, we find **two** potential security issues. Besides, we also have **one** recommendation and **two** notes.

- Low Risk: 2
- Recommendation: 1
- Note: 2

ID	Severity	Description	Category	Status
1	Low	Unexpected spot prices due to different token decimals	Software Security	Fixed
2	Low	Incorrect rounding direction	Software Security	Fixed
3	-	Remove redundant checks	Recommendation	Confirmed
4	-	Inconsistent swap results out of the LBP process	Note	-
5	-	Administrative risk of LBP Pool creator	Note	-

The details are provided in the following sections.

2.1 Software Security

2.1.1 Unexpected spot prices due to different token decimals

Severity Low

Status Fixed in `Version 2`

Introduced by `Version 1`

Description In the current implementation, the pool creators can add or withdraw liquidity with functions `join_pool()` and `exit_pool()` out of the LBP process if the swap functionality is enabled. In this scenario, if the decimals of the two tokens are different, the spot prices can be changed while adding or withdrawing the liquidity.

For example, when the decimal of ReprToken is much larger than PoolToken, the pool creator can invoke the function `exit_pool()` to withdraw the amount of ReprToken as 0 while the withdrawn amount of PoolToken is normal (i.e., larger than 0). This can decrease the price of ReprToken. Similarly, through the `join_pool()` function, pool creators can mint infinite ReprToken without providing the corresponding amount of PoolToken.

Impact Pool creators can change the token prices within the pool.

Suggestion Revise the code logic accordingly.

2.1.2 Incorrect rounding direction

Severity Low

Status Fixed in `Version 2`

Introduced by Version 1

Description Function `join_pool()` allows the pool creator to add liquidity to the pool, and the internal function `process()` is invoked to perform the relevant calculations and transfer logic. However, when calculating the required token quantities, rounding down is used, which is incorrect.

```
6   pub fn proportional(amount: u64, numerator: u64, denominator: u64) -> anchor_lang::Result<u64>
        {
7       if denominator == 0 {
8           return Ok(amount);
9       }
10      u64::try_from((amount as u128).checked_mul(numerator as u128).unwrap().checked_div(
            denominator as u128).unwrap()).map_err(|_| ErrorCode::CalculationFailure.into())
11  }
12
13   pub fn value_from_shares(shares: u64, total_value: u64, total_shares: u64) -> anchor_lang::
        Result<u64> {
14      proportional(shares, total_value, total_shares)
15  }
```

Listing 2.1: src/utils/calc.rs

Impact The required token amount for minting the same amount of LP tokens is less than expected.

Suggestion Round up when calculating the required token amount for minting LP.

2.2 Additional Recommendation

2.2.1 Remove redundant checks

Status Confirmed

Introduced by Version 1

Description There are some redundant checks in the constraints defined for the `Accounts` structure for the instructions. For example, in the following code segment, the `token_0_balance` is validated to be larger than zero, as well as larger than or equal to `MIN_BALANCE`.

```
371         constraint = params.token_0_balance > 0 && params.token_0_balance >= MIN_BALANCE &&
                params.token_0_balance <= MAX_BALANCE @ErrorCode::ConstraintInvalidTokenBalance,
372         constraint = params.token_0_weight >= MIN_WEIGHT && params.token_0_weight <= MAX_WEIGHT
                @ErrorCode::ConstraintInvalidTokenWeight,
```

Listing 2.2: src/lib.rs

Suggestion Remove these redundant checks.

Feedback from the Project Understood but this is to enforce the initial token balances should be greater than 0 and we don't want to enforce a min balance at the very beginning. the MIN_BALANCE setting will be reviewed and adjusted in a later version if needed.

2.3 Note

2.3.1 Inconsistent swap results out of the LBP process

Description Out of the LBP process, the pool creator can manually enable the swap functionality and change the weights of the tokens. Though the sudden changes of weights in the pools would not change the spot price, it can significantly change the invariant of the pool, resulting in inconsistent swap results for users before and after the change of weights.

```
94   pub fn update_token0_weight(ctx: Context<UpdateToken0Weight>, params: UpdateToken0WeightParams
         ) -> Result<()> {
95       validate_ctx(&ctx)?;
96       ctx.accounts.process(params)
97   }
98
99   pub fn update_token1_weight(ctx: Context<UpdateToken1Weight>, params: UpdateToken1WeightParams
         ) -> Result<()> {
100      validate_ctx(&ctx)?;
101      ctx.accounts.process(params)
102  }
```

Listing 2.3: src/lib.rs

Feedback from the Project As a LBP process, the weights are scheduled to be changed for a few days, usually from a higher project token weight to the lower project token weight finally, so weight mutation is needed during LBP process, and update token0 or token1 weights are also not allowed during LBP process.

If out of the LBP process, the pool swap flag is immediately set to FALSE once the LBP process ends, and the project team or the pool creator is supposed to remove all liquidity from the 1intro platform and 1intro platform doesn't support DEX trading directly after LBP.

Hence, I don't have too much concerns here and as a project team I don't think they will play with this set swap enabled transaction to cause trouble potentially.

2.3.2 Administrative risk of LBP Pool creator

Description The pool creator can change several configurations of the pool, which may cause administrative risks.

- Out of the LBP process, the pool creator is able to make sudden changes to the weights of the pool. Though this operation would not change the spot price, the depth and swap result of a pool would be significantly changed.
- The pool creator is able to pause the swap functionality inside the pool. This is designed to stop the LBP process. However, this feature can be abused if the pool creator is inherently malicious.

Feedback from the Project This is to provide a flexibility control to the project team, but this feature is not opened and integrated from 1intro website at the moment.

Potentially there may be some cases that the project team would like to pause the swaps (e.g. hacks or shutdown of the project, etc,) and the underlying program should be ready to

support when needed.


BOOST WEB3 THROUGH NEXT-GENERATION SECURITY & USABILITY INNOVATIONS