SECURITY AUDIT OF

NEMOSWAP SMART CONTRACTS

Public Report

Jun 09, 2023

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Solana	A decentralized blockchain built to enable scalable, user-friendly apps for the world.
SOL	A cryptocurrency whose blockchain is generated by the Solana platform.
Lamport	A fractional native token with the value of 0.000000001 sol.
Program	An app interacts with a Solana cluster by sending it transactions with one or more instructions. The Solana runtime passes those instructions to program.
Instruction	The smallest contiguous unit of execution logic in a program.
Cross-program invocation (CPI)	A call from one smart contract program to another.
Anchor	A framework for Solana's Sealevel runtime providing several convenient developer tools for writing smart contracts.
DAO	A digital Decentralized Autonomous Organization and a form of investor-directed venture capital fund.

EXECUTIVE SUMMARY

This Security Audit Report was prepared by Verichains Lab on Jun 09, 2023. We would like to thank the Renec Foundation for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the NemoSwap smart contracts. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified some vulnerable issues in the smart contracts code.

Renec Foundation fixed the code, according to Verichains's draft report, in commit `d5c6f84467d3a0f8536c844ef9af29f823a86ac0`.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About NemoSwap smart contracts

NemoSwap is the first #DEX on the RENEC blockchain. Trade #RENEC & #reUSD seamlessly with gasless fees and a low 0.01% trading fee.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the NemoSwap smart contracts. It was conducted on commit `44046ff44967e7c6a32c50317a8c93fd0f819b1d` from git repository link: *https://github.com/renec-chain/nemo-swap/*. Only the changes in *https://github.com/renec-chain/nemo-swap/compare/5b488e9226c6307966300e179d2abb367011d307...44046ff44967e7c6a32c50317a8c93fd0f819b1d* are in the scope of current audit context.

The latest version of the following file was made available in the course of the review:

SHA256 Sum	File
26408f7001d2a383daa1eb0d3cef952d81397f7f80a3dad7b1d3ea034cb0a604	instructions/set_pool_creator_authority.rs
e5cad663e169839987f90d57658470a966896faf59bd3dade180403a9a0064f8	instructions/initialize_reward.rs
e4cb016aa2f151bc16b51d8959a31be2044956213da25c8056b20a59b14d6aef	instructions/update_fees_and_rewards.rs
2827c070f0aa993784d858134ee7756edfab757079e8adf09eecdd8a81d199e1	instructions/set_enable_flag.rs
e757b61ebafd82fd20cdfda1446e0382da2e2f5220133959bf404d1dfa84a91f	instructions/initialize_pool.rs
7c3f2cad9465878c0bb912a075840e910055a7217f32f5a8c1fd0049a53dd68d	instructions/swap.rs
81b7555e296dbdaa70044b25555d45e52b5e0e3a32a6087749426d5bab7cbd3b	instructions/open_position_with_metadata.rs
5e1d2f7d05252d305d1284d843fd1b8e102eac9390f320aedf0aa3241774719a	instructions/decrease_liquidity.rs
f0d07ec9fac04afc84bec9a7f974cdc03fe0e5fb45df646b4a7fd7fb716278c2	instructions/collect_fees.rs

f29f8cae73e96f6839e9088d307fe91a58df0b0bfb0c4555f3e0c773d8699c05	instructions/increase_liquidity.rs
84db8e46a82ebc8eb965ee120745bd0d4a6881e074d45b36691619ade8310617	instructions/set_protocol_fee_rate.rs
6009c80bd59b492140be13dbdfe4342e873adce66af721160cbf067d9bde68b3	instructions/set_fee_rate.rs
951f12a7c2c6893f6f975f351a757b1f77ea9819a9c64a6adc392b20102e4a7c	instructions/set_reward_authority_by_super_authority.rs
e0170cb41de9fadae5653ec70465102a84eba429bc2e15724953511fcf7e9f9d	instructions/set_fee_authority.rs
972fae9297311f68986b83365b2bbabd25a7fc837ddea9cd4a80fed2939ba447	instructions/collect_reward.rs
00c6ec3f9daf42111fba70eb4fa7911cea216975e8be39e9a00bf0691df1d7bf	instructions/close_position.rs
f921791fc3106462ccb89e71626a5d9de58198362143a6f71368b81ccd27539e	instructions/collect_protocol_fees.rs
53e7c1dfaeb70d5f456dfb28d5c2fc90feef7f56ea2f659f165bc25cd063ee70	instructions/mod.rs
d029243f9b7b261d65e145dd47fa8cb30935d5f46b6b820efed0f90794e18825	instructions/set_default_fee_rate.rs
25f3423fcbb0d4b356332825a3d49cb202fc30772cc89d2cdb244fd115837769	instructions/set_default_protocol_fee_rate.rs
520d17f8a02f8b2b614047dc769959bd7b80e8e948045d591b1f1181b54938c3	instructions/set_collect_protocol_fees_authority.rs
e7acaf2192aa54f2ada0b7fb5dec615c4ac4b924eee4ba5112ec476a16415156	instructions/set_reward_emissions_super_authority.rs
7f4aae67f8f408c8aaea1538a4da31951e4a39bd86f479d49d0c60151cd4bb9b	instructions/set_reward_authority.rs
e039f9dfa4505c638512ccb303663b6766ea7f01e97650cf3d50340f8202c638	instructions/set_reward_emissions.rs
43b5112a09ba5ab626ba461c9a6589ef23b38f745809026074b938a73186f9cf	instructions/initialize_fee_tier.rs

4bd1560266e3dc3fc32bc98828113914ecc955c5e559880afd4c1206ab06e9a4	instructions/initialize_tick_array.rs
3ad0c95967d66c4c5088e2d0d709d7cf00c73645c065dbb91b6e3019de8f1b55	instructions/open_position.rs
256245f8f6915b25ec26f6316729e24f039ad86256b5f1e80a1cfe99df7c9263	instructions/initialize_config.rs
7ae90a53c31960fcff0e060e4574d0591d2c937f2f4e9a73d45e260493e819f4	util/util.rs
d9bc20505c60aef48f24c1846ebd1bb55c1846fd9fc2d057b8a7271e71dab554	util/test_utils/liquidity_test_fixture.rs
e986cee3936baca0d760714dbf1a07bf2d723d1493d044c699bd421666c0e3e7	util/test_utils/mod.rs
5233623ed906029c2fbac10b1366256679e8510d50d43d8c7a7f9a677724dfec	util/test_utils/swap_test_fixture.rs
6c77ea782e40f2703c2501cefe8c62d89254a732db96f8fc1c3390974c183e59	util/token.rs
2e1a702516e530119883052ec3f61db842ab41b82a15e26d1c3b77a0d6984d37	util/mod.rs
d23e36b008c6197535dbe7b51a129afceaedf5a7fe3b54d6a17ff164fac3dc35	util/swap_tick_sequence.rs
78362bffc4c68465c9ddc21be0c38e7abbcd04fcb285074d95cf9aeddb4bbec9	constants/test_constants.rs
cc4a69e0530b44fb97d32602bc2b82ba49bc7344fd3f09250b509d373c262594	constants/mod.rs
f60fd2a6dac059f4b72ff3470b30f34510f39ba7a6dcfef4bca53ed062113ccb	lib.rs
42c339d490e79303a0a43ba5045e141ef9e5c0d64988280b82a81e8b6c887e70	tests/swap_integration_tests.rs
7f881beb89967887fbba71accae54a785c54cc934c4e5eb1ccc50525922c705b	tests/swap_test_cases.json
b160d295b6905af260f9f0bdd12dc64e7da5c3251d70c88766cc25dc0971ec54	tests/mod.rs

7c1679352b750516ae540c6b9cc170f8256a03205b9f97c63e1946790870482e	state/config.rs
0b6d50e991ab2bd1ce872f3aa9f29c30b15711e341fd08877f96e9f81ca98761	state/whirlpool.rs
ec4ea46d5cbadd6e11976d28fe4ecf47a0e8fe92c4ff82b84aefc6fc452f9cb3	state/mod.rs
b7ab7b9b46ce8f3b1a67ffd82d0f930b778cd20504dd7c82c9ac09f7dfb59f91	state/fee_tier.rs
df13fce8b980cc5fdb4286b51acc62dd14a89822b44c1fe201bd860c7121d9e2	state/position.rs
8ca17bc0824d5db7e55469c049b56e477dc7b2d530449247b68e623a66bca519	state/tick.rs
3d539f397922de7bba77a84ecb90d6592333b3a6e480aaab862808d33b396d13	math/swap_math.rs
76e250b44b4c2814c6ab69eb71ade1bcfe7e7894412be4979904209b7ac10672	math/tick_math.rs
29686dec884e355ec52b78142e52cdf32c0d2ac23eaeac874b298ceccd584d51	math/token_math.rs
217a9138d346bee5c4563cc8711727b2888f3f513673325fe2609c79413dcfab	math/mod.rs
7ce29227bc73c003c0f282569fc5b327e0d7db886c8633973366abac152409fa	math/bit_math.rs
25d9afa48b422ac4a008836994655c5e38fa949a4e663cf01f0d916c5b637597	math/bn.rs
56e1781109a1f5139e30e59cd7e5e2aee66bbab9fa8667b8d3eb11b80beef713	math/u256_math.rs
45212362f05107b700c6df8f04395ee16eee95336f07fea87720c81531d53bf5	math/liquidity_math.rs
a8e93348c2796533fa57fad65985041d5f58f28b89cf5cd8511cb77554ca531e	manager/tick_manager.rs
c1e2be21d8237d935d356879e4cd040d71282286e56bf766de77bf22f7c8cd98	manager/liquidity_manager.rs

6d038b05526e3d8ec57e70a4e59521ba615957b1165d3b25847e6701601c06cd	manager/mod.rs
69bc96763bc7803a646edd54bba259b0b00342c3f090cfcce5adecd892ca2b3c	manager/whirlpool_manager.rs
da6c841a8040707ccf08f935fdb2658e5d6e5aa7dd19dd8cfd5de5a594102c81	manager/position_manager.rs
f6128413370a1f190d1871b7cd8bce9f0e5ef941db5c21f7175bf972401aae39	manager/swap_manager.rs
7d7d9d6aed19650ad1ecc6e6588e163ecc783e6030e6b25bd78b75f4b124713e	errors.rs

1.3. Audit methodology

Our security audit process for Solana smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using our in-house smart contract security analysis tool.
- 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 were considered during the audit of the Solana smart contract:

- Arithmetic Overflow and Underflow
- Signer checks
- Ownership checks
- Rent exemption checks
- Account confusions
- Bump seed canonicalization
- Closing account
- Signed invocation of unverified programs
- Numerical precision errors
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 1. Severity levels

1.4. Disclaimer

Please note that security auditing cannot uncover all existing vulnerabilities, and even an audit in which no vulnerabilities are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

2. AUDIT RESULT

2.1. Overview

The NemoSwap smart contracts was written in `Rust` programming language and `Anchor` framework. It is built based on Whirpools - an open-source concentrated liquidity AMM contract on the Solana blockchain.

2.2. Findings

During the audit process, the audit team had identified some vulnerable issues in the given version of NemoSwap smart contracts.

Renec Foundation fixed the code, according to Verichains's draft report, in commit `d5c6f84467d3a0f8536c844ef9af29f823a86ac0`.

2.2.1. initialize_pool.rs - Unsafe `UncheckedAccount` for `token_mint_a` MEDIUM

When `InitializePool`, the program use `UncheckedAccount` for `token_mint_a` but it only checks that the account can be deserialized to `Mint` account, but it does not check that the account's owner is `SPL token` (`Anchor` will do this check if we are using `Account<'info, Mint>`).

```
pub struct InitializePool<'info> {
    pub whirlpools_config: Box<Account<'info, WhirlpoolsConfig>>,

    /// CHECK: token_mint_a will be verified in handler,
    pub token_mint_a: UncheckedAccount<'info>,
    pub token_mint_b: Account<'info, Mint>,
    ...
}

pub fn handler(
    ctx: Context<InitializePool>,
    bumps: WhirlpoolBumps,
    tick_spacing: u16,
    initial_sqrt_price: u128,
) -> ProgramResult {
    ...
    // Only check Mint Info when token a is not a native mint.
    if !native_mint::check_id(&token_mint_a) {
        let mut data: &[u8] = &ctx.accounts.token_mint_a.try_borrow_data()?;
        Mint::try_deserialize(&mut data)?;

        if token_mint_a.ge(&token_mint_b) {
            return Err(ErrorCode::InvalidTokenMintOrder.into());
        }
    }
    ...
}
```

RECOMMENDATION

Use `Account<'info, Mint>` for `token_mint_a` or check that the owner of `token_mint_a` is `SPL token`.

UPDATES

- *Jun 09, 2023*: This issue has been acknowledged and fixed by the Renec Foundation team.

2.2.2. initialize_pool.rs - Can initialize 2 pools for a pair with native mint MEDIUM

When `InitializePool`, the program lets native mint (`So11111111111111111111111111111111111111112`) to be the base (the first listed one) in the pair. Otherwise, the program requires `token_mint_a` pub key is less than `token_mint_b` pub key to avoid adding 2 pools for a pair (swap token position). This check is only performed when `token_mint_a` is not native mint so if `token_mint_a` is native mint and `token_mint_b` pub key is less than `token_mint_a` pub key, we can add 2 pools for this pair by swapping the tokens.

For example, we can add both pools `SRMuApVNdxXokk5GT7XD5cUUgXMBCoAz2LHeuAoKWRt/So11111111111111111111111111111111111111112`

(a < b)

and

`So11111111111111111111111111111111111111112/SRMuApVNdxXokk5GT7XD5cUUgXMBCoAz2LHeuAoKWRt`

(a is native).

```
pub struct InitializePool<'info> {
    pub whirlpools_config: Box<Account<'info, WhirlpoolsConfig>>,

    /// CHECK: token_mint_a will be verified in handler,
    pub token_mint_a: UncheckedAccount<'info>,
    pub token_mint_b: Account<'info, Mint>,
    ...
}

pub fn handler(
    ctx: Context<InitializePool>,
    bumps: WhirlpoolBumps,
    tick_spacing: u16,
    initial_sqrt_price: u128,
) -> ProgramResult {
    ...
```

```
  // Only check Mint Info when token a is not a native mint.
  if !native_mint::check_id(&token_mint_a) {
    let mut data: &[u8] = &ctx.accounts.token_mint_a.try_borrow_data()?;
    Mint::try_deserialize(&mut data)?;

    if token_mint_a.ge(&token_mint_b) {
      return Err(ErrorCode::InvalidTokenMintOrder.into());
    }
  }
  ...
}
```

RECOMMENDATION

If the program want to let native mint to be the base for all pairs, we need to require that quote is not native mint.

Otherwise, require that `token_mint_b` pub key is always greater than `token_mint_a` pub key.

```
pub struct InitializePool<'info> {
  pub whirlpools_config: Box<Account<'info, WhirlpoolsConfig>>,

  /// CHECK: token_mint_a will be verified in handler,
  pub token_mint_a: UncheckedAccount<'info>,
  pub token_mint_b: Account<'info, Mint>,
  ...
}

pub fn handler(
  ctx: Context<InitializePool>,
  bumps: WhirlpoolBumps,
  tick_spacing: u16,
  initial_sqrt_price: u128,
) -> ProgramResult {
  ...
  // require the quote is not native mint
  if native_mint::check_id(&token_mint_b) {
    return Err(ErrorCode::InvalidQuoteToken.into());
  }

  // Only check Mint Info when token a is not a native mint.
  if !native_mint::check_id(&token_mint_a) {
    let mut data: &[u8] = &ctx.accounts.token_mint_a.try_borrow_data()?;
    Mint::try_deserialize(&mut data)?;

    if token_mint_a.ge(&token_mint_b) {
      return Err(ErrorCode::InvalidTokenMintOrder.into());
    }
  }
```

```
    ...
}
```

UPDATES

- *Jun 09, 2023*: This issue has been acknowledged and fixed by the Renec Foundation team.

2.2.3. lib.rs - Wrong dev documentations INFORMATIVE

In current code, anyone can create a new pool, `pool_creator_authority` is the only one who can disable pools but the dev documentations states that `pool_creator_authority` is the one who can create a new pool.

```
/// Sets the pool creator authority for a WhirlpoolConfig.
/// Only the current pool creator authority has permission to invoke this instruction.
///
/// ### Authority
/// - "pool_creator_authority" - Set authority that can create a new pool in the
WhirlpoolConfig
pub fn set_pool_creator_authority(ctx: Context<SetPoolCreatorAuthority>) -> ProgramResult {
    return instructions::set_pool_creator_authority::handler(ctx);
}
```

RECOMMENDATION

Fix the dev documentations.

UPDATES

- *Jun 09, 2023*: This issue has been acknowledged and fixed by the Renec Foundation team.

2.2.4. config.rs - More space is allocated for `WhirlpoolsConfig` than needed INFORMATIVE

`WhirlpoolsConfig` size is only 8 bytes for `discriminators`, 128 bytes for 4 `Pubkey` and 2 bytes for an `u16` so the optimized length must be `8 + 128 + 2`.

```
#[account]
pub struct WhirlpoolsConfig {
    pub fee_authority: Pubkey,
    pub collect_protocol_fees_authority: Pubkey,
    pub reward_emissions_super_authority: Pubkey,
    pub pool_creator_authority: Pubkey,

    pub default_protocol_fee_rate: u16,
}
```

```
impl WhirlpoolsConfig {
    pub const LEN: usize = 8 + 128 + 4;
    ...
}
```

RECOMMENDATION

Fix the length.

```
#[account]
pub struct WhirlpoolsConfig {
    pub fee_authority: Pubkey,
    pub collect_protocol_fees_authority: Pubkey,
    pub reward_emissions_super_authority: Pubkey,
    pub pool_creator_authority: Pubkey,

    pub default_protocol_fee_rate: u16,
}

impl WhirlpoolsConfig {
    pub const LEN: usize = 8 + 128 + 2;
    ...
}
```

UPDATES

- *Jun 09, 2023*: This issue has been acknowledged and fixed by the Renec Foundation team.

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Jun 09, 2023*	Public Report	Verichains Lab

Table 2. Report versions history