Quantstamp Security Assessment Certificate

March 15th 2023 — Quantstamp Verified

NeoSwap

This audit report was prepared by Quantstamp, the leader in blockchain security.

Executive Summary

Type	DeFi
Auditors	Marius Guggenmos, Senior Research Engineer Hytham Farah, Auditing Engineer II Valerian Callens, Senior Research Engineer
Timeline	2022-12-05 through 2022-12-12
Languages	Rust
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	NEOSWAP functional graph
Documentation Quality	Low
Test Quality	Low

Source Code

Repository	Commit
neoswap-ai/solana-multiway-swap	5d14610 initial audit

Total Issues	**15** (15 Resolved)
High Risk Issues	0 (0 Resolved)
Medium Risk Issues	**1** (1 Resolved)
Low Risk Issues	**8** (8 Resolved)
Informational Risk Issues	**6** (6 Resolved)
Undetermined Risk Issues	0 (0 Resolved)

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
Informational	The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
Undetermined	The impact of the issue is uncertain.
Unresolved	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
Acknowledged	The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).
Fixed	Adjusted program implementation, requirements or constraints to eliminate the risk.
Mitigated	Implemented actions to minimize the impact or likelihood of the risk.

Summary of Findings

Initial audit:
The audited contract allows performing swaps between several parties using an arbitrary combination of Sol and NFTs. Apart from the issue that initiating a cancellation or claim of a Sol deposit requires the admin of the swap to sign the transaction, we identified only issues of *low* or *informational* severity.
In terms of overall quality of the project, we found that both the accompanying documentation and the test suite are severely lacking. Recommendations for improvements for both can be found in the dedicated sections and the findings of this report.

Fix review: All issues of this report have been fixed or mitigated by the team. The code and external documentation, as well as the test suite have mostly stayed the same. We recommend improving them.

ID	Description	Severity	Status
QSP-1	Sol Deposit Can only Be Cancelled/Claimed by the Initializer	^ Medium	Fixed
QSP-2	`validate_initialize()` Does Not Check Expected Item Count	⌄ Low	Fixed
QSP-3	Instructions Can Inaccurately Return Success on Failure	⌄ Low	Fixed
QSP-4	Not Documenting Unchecked Accounts	⌄ Low	Mitigated
QSP-5	Automatically Closing Associated Token Accounts Never Works	⌄ Low	Fixed
QSP-6	Anybody Can Retrieve Rent Exemption Amount From Cancelled Swaps	⌄ Low	Fixed
QSP-7	Swap Status Not Always Changed to `Cancelled` when Cancelling an NFT	⌄ Low	Fixed
QSP-8	Missing Input Validation	⌄ Low	Mitigated
QSP-9	Confusing `TradeStatus` Enum Usage	⌄ Low	Fixed
QSP-10	Mixing Anchor Constrains and Manual Checks	○ Informational	Fixed
QSP-11	Wrong Equality Check for Sol Amount	○ Informational	Fixed
QSP-12	Comparing Hardcoded Value Instead of Enum	○ Informational	Fixed
QSP-13	Redundant Sol Transfer	○ Informational	Fixed
QSP-14	Possible Overflow when Calculating the Size of `SwapData`	○ Informational	Fixed
QSP-15	Error Descriptions Not Aligned with the Emission Context	○ Informational	Mitigated

Quantstamp Audit Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

DISCLAIMER:
If the final commit hash provided by the client contains features that are not within the scope of the audit or an associated fix review, those features are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

The Quantstamp auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
   ii. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Findings

QSP-1 Sol Deposit Can only Be Cancelled/Claimed by the Initializer

Severity: *Medium Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: The `cancel_sol()` and `claim_sol()` functions check that the signer is the initializer of the swap. This means users cannot retrieve their funds themselves and require the admin to cancel/retrieve the deposit for them. This is not the case when cancelling/claiming NFT deposits where users can perform this action by themselves.
Additionally, the "NEOSWAP functional graph.png" document shows that users can claim deposits themselves, which means either the code or the documentation is wrong.

Recommendation: Remove the requirement that the signer needs to be the initializer from the `cancel_sol()` and `claim_sol()` functions.

Update: Anybody who is part of the trade can now call the `cancel_sol()` and `cancel_nft()` functions, and anybody can call the `claim_sol()` and `claim_nft()` functions on behalf of others.

QSP-2 `validate_initialize()` Does Not Check Expected Item Count

Severity: *Low Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: The `validate_initialize()` function only checks that the sum of all `swap_data_account.item.amount` entries for non-NFT items is equal to zero. Therefore, it is possible to validate a swap where some NFT entries are still missing.

Recommendation: Consider adding a member to the `SwapData` struct that holds the expected number of items. This should be set during initialization and verified in the `validate_initialize()` function.

Update: The recommendation has been implemented.

QSP-3 Instructions Can Inaccurately Return Success on Failure

Severity: *Low Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: Many functions iterate over the swap data items and only perform an action if a specific condition is met. These functions use a `transfered` boolean to make sure the action is taken only once per function call. To check whether the function failed and no item met the conditions, the for loop contains the check `item_id == ctx.accounts.swap_data_account.items.len() && !transfered`. Since the for loop iterator `item_id` only goes up to, not including, the `items.len()`, this condition will never be true. This will result in the functions returning `Ok(())` despite failing.
Affected functions:

- `deposit_nft()`
- `deposit_sol()`
- `claim_sol()`
- `claim_nft()`
- `cancel_sol()`
- `cancel_nft()` (check is correct but should still be moved to after the loop)

Recommendation: Instead of checking the `item_id` in each loop iteration, check whether `transfered` is false after the loop. In case no item has been transferred, return an error. Additionally, when setting `transfered` to true, break out of the loop using `break` or return early to avoid needlessly looping over all items.

Update: The recommendation has been implemented.

QSP-4 Not Documenting Unchecked Accounts

Severity: *Low Risk*

Status: Mitigated

File(s) affected: `lib.rs`

Description: Properly verifying the input accounts is one of the most important steps when trying to secure a Solana program. Anchor enforces this by requiring a `/// CHECK:` annotation with the reason for why an account does not use a checked type. The code base uses many unchecked accounts without providing a reason for why no checks are required or what checks are performed manually, requiring to build with the `--skip-lint` flag.

Recommendation: Add `/// CHECK:` annotations for unchecked account types.

Update: While no annotations for unchecked account types have been added, most unchecked accounts have been converted to checked ones. With only two instances of unchecked accounts remaining, we consider the left-over risk to be negligible.

QSP-5 Automatically Closing Associated Token Accounts Never Works

Severity: *Low Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: The functions `claim_nft()` and `cancel_nft()` contain code that checks whether the swap's associated token account no longer contains any tokens and can be closed. Since the transfer happens in the same instruction without reloading the account data in-between, this check will always fail.

Recommendation: Perform a call to `Account.reload()` before reading state that was modified by a CPI call.

Update: Calls to `account.reload()` have been added as recommended. While potential errors originating from the `reload()` call are ignored, this is unlikely to be an issue due to the auto-close functionality being a convenience feature that can also be performed manually.

QSP-6 Anybody Can Retrieve Rent Exemption Amount From Cancelled Swaps

Severity: *Low Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: The `validate_cancel()` function is used to close the swap data account after a swap has been cancelled and all of the items have been returned. Since there are no restrictions on who can call this function, anybody can close these accounts and receive the Sol for the rent exemption. The documentation of this function also states that the signer is supposed to be the initializer.

Recommendation: Add a check that only the initializer of the swap is authorized to call this function.

Update: A constraint has been added to the `ValidateAndClose` struct that requires the caller to be the initializer.

QSP-7 Swap Status Not Always Changed to `Cancelled` when Cancelling an NFT

Severity: *Low Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: In the function `cancel_nft()`, the swap status is updated to `Cancelled` only in the second case of the main if condition (`else if` block). As a result, if an item falls into the first case of the main if condition (first `if` block), an item status can be changed to `Cancelled` without leading to the modification of the swap status.

Recommendation: Adjust the code to perform the "General status" update in case the code path enters the first `if` statement as well.

Update: The first if block has been removed. This is possible due to the changes in the `validate_cancel()` function, which does not require items to be cancelled if they have not been deposited already.

QSP-8 Missing Input Validation

Severity: *Low Risk*

Status: Mitigated

File(s) affected: `lib.rs`

Description: It is important to validate inputs, even if they only come from trusted addresses, to avoid human error. In particular:

1. the addresses `owner` and `destinary` of a new item can be the same in the functions `init_initialize()` and `initialize_add()`.
2. items can be accepted with an amount of `0` in the functions `init_initialize()` and `initialize_add()`, because the functions `is_positive()` and `is_negative()` return `false` for a value of `0`.
3. negative `amount` values are accepted in `init_initialize()` in case `is_nft` is `true`.
4. since the `sent_data.items[0]` entry in the `init_initialize()` function is supposed to be the swap fee, Sol deposits with a positive `amount` should probably not be allowed.

Recommendation: Implement more input validation as laid out in the description of this finding.
To do so, consider extracting the logic that validates whether a `NftSwapItem` is in a valid state into a separate function that can be reused in both `init_initialize()` and `initialize_add()`. Additionally restrict the item that constitutes the fee in `init_initialize()` even more to only allow actual fees.

Update: The issues from the description have been resolved as follows:

1. Unresolved.
2. Mitigated. It is still possible to add nfts with the amount set to `0`.
3. Unresolved.
4. Resolved. Mention of the item being the fee has been removed. It is therefore likely that positive values should be accepted as well.

QSP-9 Confusing `TradeStatus` Enum Usage

Severity: *Low Risk*

Status: Fixed

File(s) affected: `lib.rs`

Description: The `TradeStatus` enum is used for both the swap data status and the status of the items contained in the swap data. Additionally, the name of the enumeration does not always match the operation that was performed by an instruction, e.g. the item status is set to `Deposited` when adding a "Sol item" with a negative `amount`.

Recommendation: We recommend taking several actions to avoid any confusion stemming from the usage of the `TradeStatus` enum:

- use a different enum for the swap status and the item status.
- add a diagram of the state machines of status changes to the documentation.
- clearly document state transitions in the code that are unexpected, e.g. using `Deposited` when adding a fee item.

Update: In addition to the `TradeStatus`, an `ItemStatus` enum has been introduced. In addition to using different enums, more values have been added to both of them. Overall, the changes lead to the state transitions being easier to follow.

QSP-10 Mixing Anchor Constrains and Manual Checks

Severity: *Informational*

Status: Fixed

File(s) affected: `lib.rs`

Description: Many functions perform checks at the start that could be performed using anchor annotations instead. Mixing manual checks and anchor annotations makes it harder to reason about the security of the implementation since the checks are done in more than one place.

Recommendation: Replace `require*!` statements with anchor constraints whenever possible. Additionally, use the checked types `SystemAccount` and `TokenAccount` for the system and token program respectively.

Update: Most require statements have been replaced by anchor constraints. The system and token accounts now use a checked account type.

QSP-11 Wrong Equality Check for Sol Amount

Severity: *Informational*

Status: Fixed

File(s) affected: `lib.rs`

Description: The functions `claim_sol()` and `cancel_sol()` both use the following if statement to check that an account has sufficient balance: `if swap_data_lamports_initial > amount_to_send`. While this is unlikely to be an issue due to accounts storing some amount of Sol for rent exemption, the correct check is greater **or equal**.

Recommendation: Replace the check with `if swap_data_lamports_initial >= amount_to_send`.

Update: The code now performs a greater than or equal check as recommended.

QSP-12 Comparing Hardcoded Value Instead of Enum

Severity: *Informational*

Status: Fixed

File(s) affected: `lib.rs`

Description: The functions `deposit_sol()`, `claim_nft()` and `cancel_nft()` perform checks against the `status` of items using hardcoded values. In case the code gets updated and enum values change, these instances might be missed and lead to incorrect results.

Recommendation: Replace the comparisons using hardcoded values with the corresponding enum type.

Update: Hardcoded values have been replaced by the appropriate enum.

QSP-13 Redundant Sol Transfer

Severity: *Informational*

Status: Fixed

File(s) affected: `lib.rs`

Description: The functions `validate_claimed()` and `validate_cancel()` contain code to transfer remaining Sol to the signer. Since the `swap_data_account` member of `ValidateAndClose` is annotated with `close = signer`, anchor already sends any remaining Sol to the signer.

Recommendation: Remove the Sol transfer code from the relevant functions.

Update: The redundant code has been removed.

QSP-14 Possible Overflow when Calculating the Size of `SwapData`

Severity: *Informational*

Status: Fixed

File(s) affected: `lib.rs`

Description: The size of the object `SwapData` is calculated using the formula `SwapData::LEN + ( nb_items as usize * NftSwapItem::LEN )`. For unusually high values of the variable `nb_items`, an overflow is possible.

Recommendation: Consider using functions that catch overflows, such as `checked_mul()` and `checked_add()`.

Update: The calculation now uses functions that catch overflows.

QSP-15 Error Descriptions Not Aligned with the Emission Context

Severity: *Informational*

Status: Mitigated

File(s) affected: `lib.rs`

Description: In the function `claim_sol()`, an error can be raised with a message that does not describe the situation accurately:

- the error `SumNotNull` is raised instead of an error mentioning that there is an insufficient amount of lamports in the escrow.
- the error `NotReady` is raised instead of the error `NoSend`.

Additionally, the message of the error `UnexpectedState` (`"The status given is not correct"`) is not always aligned with the situation when that error is raised. It is accurate in the function `init_initialize()` when it describes an object sent by the user. However, it can also describe objects currently stored in the swap data account. In that case, the status is not "given" by the user but is already stored on-chain. This is also the case in the functions `initialize_add()`, `initialize_add()`, `cancel_sol()`, `deposit_nft()`, `deposit_sol()` and `validate_deposit()`.
These discrepancies could negatively affect troubleshooting.

Recommendation: Consider adding error codes that more accurately describe the errors.

Update: The error codes used by the `claim_sol()` function have been updated to better reflect the error condition. However, the error message of `UnexpectedState` has not been changed and is still slightly misleading for most cases where it is used.

Adherence to Specification

1. In the "NEOSWAP functional graph" document, change
    - "Deposit User" to "User Deposit"
    - "Claim User" to "User Claim"
    - "Cancel User" to "User Cancel"

    The former sounds like depositing, claiming, and canceling user accounts, whereas the latter implies that the user is performing the action of depositing/claiming/canceling.
2. The "Deposit User" flowchart has a segment that has no incoming arrows but only outgoing arrows, namely, the yellow diamond with the text "findOrCreate PDA ATA with NFT mint". Consider making it more clear how this part of the flowchart is accessed.
3. Change "is all items deposited" to "are all items deposited?" in the Validate DEPOSIT ADMIN flowchart.

Code Documentation

1. The documentation on the functions `claim_nft()` and `cancel_nft()` states that the `signer` is supposed to be the initializer. Since there are no constraints on this account, anybody can be the signer. We believe users should be able to cancel a swap and therefore recommend fixing the documentation to remove the mention of the initializer.
2. **Fixed:** Incorrect description in the documentation of the function `validate_cancel()`: the status is changed to `CancelledRecovered` instead of `Closed` state.
3. The `NftSwapItem` struct described in the documentation of the function `initialize_add()`. Such a description should be moved to where that object is declared.
4. Parameters of the functions `cancel_nft()` (`token_program`) and `claim_nft()` (`signer`) are listed more than once in the preceding line.
5. Incorrect comment in the function `initialize_add()`. The for loop does not check "if already one user has a Sol item", but whether the owner of the item to add already has a Sol item.
6. Incorrect comment in the documentation of the function `cancel_nft()`. The function does not transfer the NFT from the shared user to the escrow, but the opposite.

Adherence to Best Practices

1. Frequently perform static analysis using `cargo clippy` during development and implement the suggestions. Most of the suggestions can be automatically implemented by using the `--fix` flag.
2. **Fixed:** Programs with known IDs are verified manually in each instruction implementation. Consider using the `SystemAccount` and `TokenAccount` types instead.
3. **Mitigated** (the bumps are still present as parameters): Many functions have the bump seed as a parameter for the instruction. By using only the canonical bump, this can be omitted. Consider using `#[account(seeds = ..., bump)]` and removing the `bump` parameter from all instructions.
4. The `InitInitialize` struct contains an entry for the SPL token program. Since this account is unused, remove it.
5. The implementation of `SwapData::size` can be simplified by removing the `SwapData` parameter. Simplified implementation: `rust pub fn size(nb_items: u32 ) -> usize { SwapData::LEN + ( nb_items as usize * NftSwapItem::LEN ) }`
6. Whenever possible, it is recommended to use a context struct name that matches the endpoint that uses it (when dedicated). This is not the case for:

    . `validate_initialize()` and `VerifyInitialize`;

    . `validate_deposit()` and `Validate`;

7. `if` blocks are not always properly indented. This is the case in the functions `deposit_nft()`, `claim_nft()` and `cancel_sol()`.
8. When depositing NFTs on Solana, rather than relying on the user to correctly input the ATA of the swap PDA, it is advisable to use `get_associated_token_address`, which simply needs the mint, and PDA's pubkey to deterministically derive the ATA. This will prevent any potential errors of submitting the wrong or a secondary token account.
9. Use the functions provided by anchor to perform token transfers. For the `deposit_nft()` for example, use the following implementation:

```
anchor_spl::token::transfer(
    CpiContext::new(
        token_program.clone(),
        anchor_spl::token::Transfer {
            from: item_from_deposit.to_account_info(),
            to: item_to_deposit.to_account_info(),
            authority: signer.to_account_info(),
        }
    ),
    ctx.accounts.swap_data_account.items[item_id].amount.unsigned_abs()
)?;
```

Test Results

Test Suite Results

Initial audit:
To run the tests, we followed the README instructions:

1. Run `anchor test --skip-lint`.
2. Once the "Running test suite: ..." message appears, copy the `good_idl.json` over the generated idl with `cp good_idl.json target/idl/neo_swap.json`.

We believe the current test suite is of fairly low quality and should be improved in the following ways:

1. The existing tests only send the transactions and as long as there are no failed transactions, the tests pass. This is not sufficient, as it does not make sure that the expected actions were taken. We therefore highly recommend adding verification code that ensures balances were changed, accounts deleted or created, etc. As an example, the feature of closing the swap's associated token accounts automatically in the `claim_nft()` and `cancel_nft()` functions is not working but the tests are passing.
2. Add more tests. While we were unable to collect coverage data, the test file is only about 500 lines of code and supposed to test roughly 700 lines. While this is not a great measure to use, drawing from experience shows that good test suites tend to contain significantly more code than the implementation under test.

Fix review:
The tests have been adjusted to work with changes introduced by the fixes. However, the quality and quantity of the tests are mostly unchanged from the initial audit.

```
  swapCoontractTest
programId DX1pLgDgRWgUCLHHDgVcnKkSnr5r6gokHprjYXo7eykZ
signer airdrop done ERp9tkkNjBjtJEiHut5be5WF2M1s6pZWwwjWVfDPUqL3
user airdrop done FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam
user airdrop done FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1
user airdrop done AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA
     ✓ Initializing accounts (1011ms)
XXXXXXXXXXXXXXX - user  FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam
mint  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
with ata:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe

XXXXXXXXXXXXXXX - user  FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1
mint  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J
with ata:  DhCTp3JG8GFXBafyu7GW9vyNrTMnappLmJMH4u6V4n5N

XXXXXXXXXXXXXXX - user  AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA
mint  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf
with ata:  6fDto5KJhyzRwHSvBwmhHvFHvqr7g4WFPtV8xuw1xgsD

     ✓ users instruction (3638ms)
initInitTransaction Added
```

```
XXXXXXX - added to init item n°  1  XXXXXXX
XXXXXXX - added to init item n°  2  XXXXXXX
XXXXXXX - added to init item n°  3  XXXXXXX
XXXXXXX - added to init item n°  4  XXXXXXX
XXXXXXX - added to init item n°  5  XXXXXXX
XXXXXXX - added to init item n°  6  XXXXXXX
XXXXXXX - added to init item n°  7  XXXXXXX
XXXXXXX - added to init item n°  8  XXXXXXX
XXXXXXX - added to init item n°  9  XXXXXXX
validateInitTransaction Added
XXX-XXX pda 34LzBCFdZS6T3JBrPDfTc2NAfh1GoQkYkhagQUL72M1M
initialized
     ✓ initialize (1617ms)
XXXXXXX - Deposit sol item n°  3  XXXXXXX
depositSolinstruction added
XXX - Deposit item n°  4  X X  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F  - XXX
CreatePdaAta Deposit Tx added
from:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe
to:  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
mint:  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
added mint  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
depositNftInstruction added
XXX - Deposit item n°  5  X X  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F  - XXX
from:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe
to:  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
mint:  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
depositNftInstruction added
XXXXXXX - Deposit sol item n°  2  XXXXXXX
depositSolinstruction added
XXX - Deposit item n°  6  X X  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J  - XXX
CreatePdaAta Deposit Tx added
from:  DhCTp3JG8GFXBafyu7GW9vyNrTMnappLmJMH4u6V4n5N
to:  5Wg9xSyxNQLnPFnqCeELqRfF2cn3NYPpfzBCeSrsdbKP
mint:  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J
added mint  5Wg9xSyxNQLnPFnqCeELqRfF2cn3NYPpfzBCeSrsdbKP
depositNftInstruction added
XXX - Deposit item n°  7  X X  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J  - XXX
from:  DhCTp3JG8GFXBafyu7GW9vyNrTMnappLmJMH4u6V4n5N
to:  5Wg9xSyxNQLnPFnqCeELqRfF2cn3NYPpfzBCeSrsdbKP
mint:  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J
depositNftInstruction added
XXXXXXX - Deposit sol item n°  0  XXXXXXX
depositSolinstruction added
XXX - Deposit item n°  8  X X  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf  - XXX
CreatePdaAta Deposit Tx added
from:  6fDto5KJhyzRwHSvBwmhHvFHvqr7g4WFPtV8xuw1xgsD
to:  4QEbyH5MVFoF5YGueAUb2b4LxUJSTPr5hLfhuaAp3uMB
mint:  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf
added mint  4QEbyH5MVFoF5YGueAUb2b4LxUJSTPr5hLfhuaAp3uMB
depositNftInstruction added
XXX - Deposit item n°  9  X X  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf  - XXX
from:  6fDto5KJhyzRwHSvBwmhHvFHvqr7g4WFPtV8xuw1xgsD
to:  4QEbyH5MVFoF5YGueAUb2b4LxUJSTPr5hLfhuaAp3uMB
mint:  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf
depositNftInstruction added
deposited users  [
  '3ANapPNEmzXJzzmiNdXZ3fkHB3LGjLDRehEzP2EojRiSvSgNfVWZabqebTELwhXLjJUTxjkzZxDeraj2KY4DTohE',
  '2fr4ssAR4xcdJQWYAJP8evUEgh2v6nUkJx5F3M5YsFmaMy3r4XP4se8RZ9TgDa5oXyJBythve83XAG9DCsWW6Jki',
  '3qiTbSGq3LdTUNbeSV5pKRn8f96xmVcXAMrF2JrmvCVDE3y4nCYUpBtHKV6Ag4ryZeCoKW26ANUwHXRaquXTb5k4'
]
     ✓ Deposit (2428ms)
not to claim item n°  0
XXXXXXX - item n°  1  XXXXXXX {
  isNft: false,
  mint: PublicKey { _bn: <BN: 0> },
  amount: <BN: -2cb41780>,
  owner: PublicKey {
    _bn: <BN: c7832f96208302e75828c45efb8f31be8aed2553542db3f3c708a17facdceca8>
  },
  destinary: PublicKey { _bn: <BN: 0> },
  status: 22
}
claimSolinstruction added
not to claim item n°  2
not to claim item n°  3
XXXXXXX - item n°  4  XXXXXXX
createUserAta ClaimNft Tx Added
claimNftinstruction added
XXXXXXX - item n°  5  XXXXXXX
createUserAta ClaimNft Tx Added
claimNftinstruction added
XXXXXXX - item n°  6  XXXXXXX
createUserAta ClaimNft Tx Added
claimNftinstruction added
XXXXXXX - item n°  7  XXXXXXX
createUserAta ClaimNft Tx Added
claimNftinstruction added
XXXXXXX - item n°  8  XXXXXXX
createUserAta ClaimNft Tx Added
claimNftinstruction added
XXXXXXX - item n°  9  XXXXXXX
createUserAta ClaimNft Tx Added
claimNftinstruction added
claimAndCloseHash : [
  '2VpQGXPtg3qLFb6w2gQk5q2QbpqWrftUbCmW3r19iUK3iBnp466kFLZyWoZMcBJBhvHDtcukTXwCd2rc3Uxvg4PN',
  '2xLZdnkAfc459NM35sX7AZRWcvbW62bNaLhQrbb6MQ3QMDzBrvTMRr6rum9UDWrjPebvA2pTGsvZXKCrpUzmRRZF',
  '4h4Zk8x3ziUmV4N42u7hGiqznYLQNZqsR9sQjTemyToHDXR64nctYY8QQPqkBB5dMAx4novLVvYBxXAmyoTJBrWm',
  'SmbJAumSGixqcCCCEWW4HHXBY2376h2PfcjE5x7JNeL69w8yPpJXAok6H47p9TqTF9EzgJyp2QGVz8Ty896QM7n',
  '3EADQ84VXbPycEforYT28ErLvzjqrvNQP56k3zQMEAT2qHbuBTX7Gr5RDSYCqPMG67rYR6uZvvcdaodi4B379g3r'
]
     ✓ claim and close (2020ms)
initInitTransaction Added
XXXXXXX - added to init item n°  1  XXXXXXX
XXXXXXX - added to init item n°  2  XXXXXXX
XXXXXXX - added to init item n°  3  XXXXXXX
XXXXXXX - added to init item n°  4  XXXXXXX
XXXXXXX - added to init item n°  5  XXXXXXX
XXXXXXX - added to init item n°  6  XXXXXXX
XXXXXXX - added to init item n°  7  XXXXXXX
XXXXXXX - added to init item n°  8  XXXXXXX
XXXXXXX - added to init item n°  9  XXXXXXX
validateInitTransaction Added
XXX-XXX pda 34LzBCFdZS6T3JBrPDfTc2NAfh1GoQkYkhagQUL72M1M
initialized
     ✓ initialize for cancel (1618ms)
XXXXXXX - Deposit sol item n°  3  XXXXXXX
depositSolinstruction added
XXX - Deposit item n°  4  X X  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F  - XXX
CreatePdaAta Deposit Tx added
from:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe
to:  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
mint:  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
added mint  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
depositNftInstruction added
XXX - Deposit item n°  5  X X  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F  - XXX
from:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe
to:  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
mint:  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
depositNftInstruction added
XXXXXXX - Deposit sol item n°  2  XXXXXXX
```

```
depositSolinstruction added
XXX - Deposit item n°  6  X X  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J  - XXX
CreatePdaAta Deposit Tx added
from:  DhCTp3JG8GFXBafyu7GW9vyNrTMnappLmJMH4u6V4n5N
to:  5Wg9xSyxNQLnPFnqCeELqRfF2cn3NYPpfzBCeSrsdbKP
mint:  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J
added mint  5Wg9xSyxNQLnPFnqCeELqRfF2cn3NYPpfzBCeSrsdbKP
depositNftInstruction added
XXX - Deposit item n°  7  X X  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J  - XXX
from:  DhCTp3JG8GFXBafyu7GW9vyNrTMnappLmJMH4u6V4n5N
to:  5Wg9xSyxNQLnPFnqCeELqRfF2cn3NYPpfzBCeSrsdbKP
mint:  FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J
depositNftInstruction added
XXXXXXX - Deposit sol item n°  0  XXXXXXX
depositSolinstruction added
XXX - Deposit item n°  8  X X  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf  - XXX
CreatePdaAta Deposit Tx added
from:  6fDto5KJhyzRwHSvBwmhHvFHvqr7g4WFPtV8xuw1xgsD
to:  4QEbyH5MVFoF5YGueAUb2b4LxUJSTPr5hLfhuaAp3uMB
mint:  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf
added mint  4QEbyH5MVFoF5YGueAUb2b4LxUJSTPr5hLfhuaAp3uMB
depositNftInstruction added
XXX - Deposit item n°  9  X X  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf  - XXX
from:  6fDto5KJhyzRwHSvBwmhHvFHvqr7g4WFPtV8xuw1xgsD
to:  4QEbyH5MVFoF5YGueAUb2b4LxUJSTPr5hLfhuaAp3uMB
mint:  kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf
depositNftInstruction added
deposited user  [
  '3m5PiKrEcscRm8h3G6qyAaPWPLPQYCh4ut8MJW4HzfkzAuSAuxCVoHreZf8RjngxmYS5cGenTwuhQNN7S9zHqEVx',
  '4RGJhCRkQiwVCU4ekYnUpMWs3bfcRuQitC1STnEiKMXzywaGhSVsdHtecPykU9DaE27bpgjXFYT4WZsVvSXLeC14',
  '5ni77L8JZ6QswHwaDLjx5CsrSKLMfwNG8eksYzQprr7FHeTQMqBW1i1r1k31AS71BoBFbC8ebbsLXxiZfsgdpzGg'
]
    ✓ Deposit for cancel (2426ms)
swdata 1
21 21
XXXXXXX - cancelling SOL n°  0  XXXXXXX 21 AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA
cancelSolinstruction added
22 21
XXXXXXX - not adding SOL n°  1 22
21 21
XXXXXXX - cancelling SOL n°  2  XXXXXXX 21 FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1
cancelSolinstruction added
21 21
XXXXXXX - cancelling SOL n°  3  XXXXXXX 21 FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam
cancelSolinstruction added
XXXXXXX - cancelling NFT n°  4  XXXXXXX 4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  5  XXXXXXX 4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  6  XXXXXXX FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  7  XXXXXXX FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  8  XXXXXXX kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  9  XXXXXXX kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA 20
cancelNftinstruction added
cancelAndCloseHash : [
  '21vSp3x3kov5XAiNZCnPcsZLv5k1UhU2LbGVNUg1WvFH4Dwtpdxrn787FiH5r8GJjuoanQp3cBf1KfugARyqznQN'
]
    ✓ partial cancel and close from in trade user (562ms)
swdata 100
111 21
XXXXXXX - not adding SOL n°  0 111
22 21
XXXXXXX - not adding SOL n°  1 22
111 21
XXXXXXX - not adding SOL n°  2 111
111 21
XXXXXXX - not adding SOL n°  3 111
XXXXXXX - not adding NFT n°  4
XXXXXXX - not adding NFT n°  5
XXXXXXX - cancelling NFT n°  6  XXXXXXX FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  7  XXXXXXX FhhagfrwqcrDQWoXcwbPDUV57xmr5pX5RcoPw6vk6h6J FYCzTHYFqK92DWMG8cFbgmFfgJXrsJyTtEpzq3v4Pko1 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  8  XXXXXXX kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  9  XXXXXXX kdbidFrPqsKRvrJXY6Wp1zLQBPu5RGPdM3zPXb8iYKf AfYMCWPH6HpU6nWJmALrTu5ZHV1gPyi8EPBCJo2SG4NA 20
cancelNftinstruction added
cancelAndCloseHash : [
  'fLzN3m1e3NYpHKWsU5PPgfCswZjUtSCQhHB9cbqckZNtdrTpZsF3Rdj3aWakoSo9MGidUw8fndXV3DPjCkLYfEW',
  '3aPFJDQjYKEB7GX1KCApURwLqxCjyW8Z1ojrdA2wuGMQ8yZP1AN4SN4ntKn3eYd6NJFfPFQAph36HpqzufNyQY5F'
]
    ✓ finish cancel and close from signer (811ms)
initInitTransaction Added
XXXXXXX - added to init item n°  1  XXXXXXX
XXXXXXX - added to init item n°  2  XXXXXXX
XXXXXXX - added to init item n°  3  XXXXXXX
XXXXXXX - added to init item n°  4  XXXXXXX
XXXXXXX - added to init item n°  5  XXXXXXX
XXXXXXX - added to init item n°  6  XXXXXXX
XXXXXXX - added to init item n°  7  XXXXXXX
XXXXXXX - added to init item n°  8  XXXXXXX
XXXXXXX - added to init item n°  9  XXXXXXX
validateInitTransaction Added
XXXXXXXXXXXXXXXXX-XXXXXXXXX pda 34LzBCFdZS6T3JBrPDfTc2NAfh1GoQkYkhagQUL72M1M
initialized [
  '2xsZwsBd3m8xafSCDWJKHNbBEyFeeEb9PBkB6ZHL1b2ABFNdMae3svzQehNmtgRqBZ9rHJ6EYBDxv9ejetyLvUoN',
  '5nJ2giTFh25kcU2uiQKxtWXuTWqM7niiAUxYEyrkC2d9xGwzx5orG3T8rH9gbrpyKYGQUXMGmKaEs1jbNcTYXr18',
  '59dKibX3Fu7UWohSfzFSeFS64JaYva8SnNtvRZyd27XQMCMFVx76tfAtgmEjqgqJN8CdgtztAY4hWdccKYjkmYmb',
  '2zcBt6L4xSpm2qZkYRaSPYGDZoq7BgJuzsQQ8S7ArGi3zPjxJTDYJZyAj4UQBGbeaFqtVpPLKHpBbBFDbhd5Eukj'
]
    ✓ initialize for mishandling (1617ms)
initInitTransaction Added
XXXXXXX - added to init item n°  1  XXXXXXX
XXXXXXX - added to init item n°  2  XXXXXXX
XXXXXXX - added to init item n°  3  XXXXXXX
XXXXXXX - added to init item n°  4  XXXXXXX
XXXXXXX - added to init item n°  5  XXXXXXX
XXXXXXX - added to init item n°  6  XXXXXXX
XXXXXXX - added to init item n°  7  XXXXXXX
XXXXXXX - added to init item n°  8  XXXXXXX
XXXXXXX - added to init item n°  9  XXXXXXX
validateInitTransaction Added
XXX-XXX pda 34LzBCFdZS6T3JBrPDfTc2NAfh1GoQkYkhagQUL72M1M
initialized
    ✓ reinitialize mishandling
initInitTransaction Added
XXXXXXX - added to init item n°  1  XXXXXXX
XXXXXXX - added to init item n°  2  XXXXXXX
XXXXXXX - added to init item n°  3  XXXXXXX
XXXXXXX - added to init item n°  4  XXXXXXX
XXXXXXX - added to init item n°  5  XXXXXXX
XXXXXXX - added to init item n°  6  XXXXXXX
XXXXXXX - added to init item n°  7  XXXXXXX
XXXXXXX - added to init item n°  8  XXXXXXX
XXXXXXX - added to init item n°  9  XXXXXXX
validateInitTransaction Added
XXX-XXX pda 34LzBCFdZS6T3JBrPDfTc2NAfh1GoQkYkhagQUL72M1M
```

```
    ✓ wrong reinitialize mishandling
not to claim item n°  0
XXXXXXX - item n°  1  XXXXXXX {
  isNft: false,
  mint: PublicKey { _bn: <BN: 0> },
  amount: <BN: -2cb41780>,
  owner: PublicKey {
    _bn: <BN: c7832f96208302e75828c45efb8f31be8aed2553542db3f3c708a17facdceca8>
  },
  destinary: PublicKey { _bn: <BN: 0> },
  status: 22
}
claimSolinstruction added
not to claim item n°  2
not to claim item n°  3
    ✓ wrong claim and close
XXXXXXX - Deposit sol item n°  3  XXXXXXX
depositSolinstruction added
XXX - Deposit item n°  4  X X  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F  - XXX
CreatePdaAta Deposit Tx added
from:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe
to:  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
mint:  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
added mint  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
depositNftInstruction added
XXX - Deposit item n°  5  X X  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F  - XXX
from:  Dr3gCPTxamz7Ap86ruotTimkcFwQEYvuGxvbhfHBwJDe
to:  4Xi8pFUtVuz8tZjrrFQVHcVFqxEHsKWfvhqXFcyUUyXe
mint:  4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F
depositNftInstruction added
swdata 1
11 21
XXXXXXX - not adding SOL n°  0 11
22 21
XXXXXXX - not adding SOL n°  1 22
11 21
XXXXXXX - not adding SOL n°  2 11
21 21
XXXXXXX - cancelling SOL n°  3  XXXXXXX 21 FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam
cancelSolinstruction added
XXXXXXX - cancelling NFT n°  4  XXXXXXX 4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam 20
cancelNftinstruction added
XXXXXXX - cancelling NFT n°  5  XXXXXXX 4s1wkZZbJQ43oR4R6BVe44narPmg9VTy93FsAgtvLH8F FyLsxDjvB7cUjzFyogv6VsGa24jKMBuuXXUpkBov9aam 20
cancelNftinstruction added
XXXXXXX - not adding NFT n°  6
XXXXXXX - not adding NFT n°  7
XXXXXXX - not adding NFT n°  8
XXXXXXX - not adding NFT n°  9
SendTransactionError: failed to send transaction: Transaction simulation failed: Error processing Instruction 0: custom program error: 0x1770
    at Connection.sendEncodedTransaction (/home/mg/audits/neoswap/neoswap_ai-solana-multiway-swap-7af498fed9d0f1520112a49800f834cb3193425b-github/node_modules/@solana/web3.js/src/connection.ts:5054:13)
    at processTicksAndRejections (node:internal/process/task_queues:95:5)
    at Connection.sendRawTransaction (/home/mg/audits/neoswap/neoswap_ai-solana-multiway-swap-7af498fed9d0f1520112a49800f834cb3193425b-github/node_modules/@solana/web3.js/src/connection.ts:5013:20)
    at sendAndConfirmRawTransaction (/home/mg/audits/neoswap/neoswap_ai-solana-multiway-swap-7af498fed9d0f1520112a49800f834cb3193425b-github/node_modules/@project-serum/anchor/src/provider.ts:288:21)
    at AnchorProvider.sendAll (/home/mg/audits/neoswap/neoswap_ai-solana-multiway-swap-7af498fed9d0f1520112a49800f834cb3193425b-github/node_modules/@project-serum/anchor/src/provider.ts:209:9) {
  logs: [
    'Program DX1pLgDgRWgUCLHHDgVcnKkSnr5r6gokHprjYXo7eykZ invoke [1]',
    'Program log: Instruction: CancelSol',
    'Program log: SolCancelledRecovered',
    'Program log: AnchorError thrown in programs/neoSwap/src/lib.rs:608. Error Code: UserNotPartOfTrade. Error Number: 6000. Error Message: User not part of the trade.',
    'Program DX1pLgDgRWgUCLHHDgVcnKkSnr5r6gokHprjYXo7eykZ consumed 13827 of 600000 compute units',
    'Program DX1pLgDgRWgUCLHHDgVcnKkSnr5r6gokHprjYXo7eykZ failed: custom program error: 0x1770'
  ]
}
    ✓ cancel from unAuthorized user (905ms)


  14 passing (19s)

Done in 24.56s.
```

Code Coverage

We were unable to generate a code coverage report for this project.

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Contracts

`5f156cffbadb8e7c3e242f62eda3f65e69fa0163b0b3859ca7ba1166260f0a33 ./programs/neoSwap/src/lib.rs`

Tests

`c308a2b7e6523ccc9ed598675dc88008e5bb54c711b4d3a31975f1feb24b3888 ./tests/neoSwap.ts`

Changelog

- 2022-12-13 - Initial report
- 2023-01-11 - Fix review (`7af498fe`)

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Aave, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites&aspo; owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that your access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, or any related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.