SCV SECURITY

AUDIT REPORT

Exotic Holding

Exotic Contracts

Prepared by SCV-Security

On 26th March 2024

Table of Contents

Introduction

SCV has been engaged by Exotic Holding to conduct a comprehensive security review with the goal of identifying potential security threats and vulnerabilities within the codebase. The purpose of this audit is to evaluate the security posture of the codebase and provide actionable recommendations to mitigate any identified risks. This report presents an overview of the findings from our security audit, outlining areas of concern and proposing effective measures to enhance the codebase's security.

Scope Functionality

The Exotic Market is an option infrastructure that can propose to buy and sell custom options. The following are the in-scope contracts:

- `contracts/exotic-holding` manages the full lifecycle of the option and creates products and holdings.
- `contracts/exotic-vanilla` creates the option with all the financial parameters needed and the mathematical formula for PnL calculation.
- `contracts/exotic-rfq` is the auction program that collects the bid of market makers in a short interval and exchanges the bid or premium against the token position.

Submitted Codebase

Exotic Contracts	
Repository	https://github.com/exotic-markets/exotic-injective
Commit	`4a974dbbd415cda7080e1bfc362be33f24268561`
Branch	`master`

Revision Codebase

Exotic Contracts	
Repository	https://github.com/exotic-markets/exotic-injective
Commit	`2b6fde4c04da2dcbc8e1666baeb2501c8933b2b4`
Branch	`master`

Methodologies

SCV performs a combination of automated and manual security testing based on the scope of testing. The testing performed is based on the extensive experience and knowledge of the auditor to provide the greatest coverage and value to Exotic Holding. Testing includes, but is not limited to, the following:

- Understanding the application and its functionality purpose.
- Deploying SCV in-house tooling to automate dependency analysis and static code review.
- Analyse each line of the code base and inspect application security perimeter.
- Review underlying infrastructure technologies and supply chain security posture.

Code Criteria

This section provides an evaluation of specific criteria aspects as described below:

- **Documentation:** Evaluating the presence and comprehensiveness of publicly available or provided explanatory information, diagram flowcharts, comments, and supporting documents to enhance code understanding.
- **Coverage:** Evaluating whether the code adequately addresses all necessary cases and scenarios, ensuring that the intended functionality or requirements are sufficiently covered.
- **Readability:** Assessing how easily the code can be understood and maintained, considering factors such as code structure, naming conventions, and overall organisation.
- **Complexity:** Evaluating the complexity of the code, including factors such as, number of lines, conditional statements, and nested structures.

The status of each criteria is categorised as either **SUFFICIENT** or **NOT-SUFFICIENT** based on the audit assessment. This categorisation provides insights to identify areas that may require further attention and improvement.

Criteria	Status	Notes
Documentation	**SUFFICIENT**	The client provided detailed documentation and workflow diagrams.
Coverage	**SUFFICIENT**	N/A
Readability	**SUFFICIENT**	The codebase had good readability overall and utilised many Rust and CosmWasm best practices.
Complexity	**SUFFICIENT**	N/A

Findings Summary

Summary Title	Risk Impact	Status
Malicious actors can lock funds by setting bid and selection end timestamp to large values	CRITICAL	RESOLVED
Incorrect authentication validation logic	SEVERE	RESOLVED
Fetched Pyth price may be stale	MODERATE	RESOLVED
Max deposit limit per user can be bypassed	MODERATE	RESOLVED
Risk of ownership retention	MODERATE	RESOLVED
No entry point to call `ConvertAddressPosition`	MODERATE	ACKNOWLEDGED
Edge case that causes locked funds due to bidders not refunding	MODERATE	RESOLVED
Symbol length is not validated during contract instantiation	LOW	RESOLVED
Missing timestamp validation for `DepositorType::Many` holdings	LOW	RESOLVED
Misconfiguring minimum and maximum deposit limits prevents users from depositing funds	LOW	RESOLVED
Two-step ownership transfer is not implemented	INFO	RESOLVED
Ambiguous error reporting	INFO	RESOLVED
Incorrect owner address attribute emitted	INFO	RESOLVED

Findings Technical Details

1. Malicious actors can lock funds by setting bid and selection end timestamp to large values

RISK IMPACT: CRITICAL	STATUS: RESOLVED

Description

The `start_product` function in `contracts/exotic-holding/src/contract/execute/start_product.rs:20` allows anyone to start any holdings. The issue is that the caller controls `msg.rfq_args`, meaning they can control the bid end (`args.bid_end`) and selection end (`args.selection_end`) timestamps.

A malicious actor can set the bid end and selection end timestamps to any `Uint64` value that can be a very long time (e.g., a few hundred years), causing the bid to be unable to be ended within a logical period.

This leads to four impacts:

1. Loss of funds deposited in the exotic-holding contract

The exotic-vanilla contract requires the “position” token denom to be included when exercising the option in `contracts/exotic-vanilla/src/contract/execute/exercise_option.rs:33`, so the profit can be distributed to the exotic-holding contract. As the “position” token denom is minted to RFQ contract in `contracts/exotic-holding/src/contract/reply.rs:49`, it is impossible to exercise the option because the “position” token denom is stuck in the RFQ contract.

Additionally, it is impossible to call the `CollectCollateral` message in the exotic-vanilla contract to recover the funds. This is because only the exotic-holding contract is authorized to call it, and it requires the RFQ state to be closed in `contracts/exotic-holding/src/contract/execute/collect.rs:35`, which is not because the selection end timestamp has not elapsed.

2. Locked bidder funds in the RFQ contract

The `Bid` message in `contracts/exotic-rfq/src/contract/mod.rs:116` allows bidders to send funds for bidding. However, it will be stuck.

The owner cannot call the `Select` message to recover the funds because it requires the bid end timestamp to elapse, and the bidder cannot call the `CancelBid` message to get their refund because it requires the selection end timestamp to elapse. Both timestamps can be maliciously set to a few hundred years later by the malicious actor.

3. Incorrect `min_bid` amount computation

If the product is still in the deposit phase, the product should not be started because the `min_bid` amount in `contracts/exotic-holding/src/contract/execute/start_product.rs:61` is not yet finalized.

If a user deposits funds, the `holding.total_deposited_amount` will increase in `contracts/exotic-holding/src/contract/execute/deposit.rs:51`, which means the `mid_bid` amount should increase, but it does not because the RFQ contract is already instantiated.

4. Winner bidder selection manipulation

The malicious actor also controls `args.selection_type`, which can be `Selection::Highest` or `Selection::Manual`. The actor can manipulate this to gain an unfair advantage from other bidders.

Additionally, there is no method to recover the locked funds from migration as the contract migration admin for the exotic-vanilla and RFQ contracts are set to `None`, meaning both contracts are immutable.

Recommendation

Consider applying the following recommendations:

- Ensure the `start_product` function can only be called after the deposit period has ended (i.e., current timestamp exceeds `holding.end_collection`).
- Modify the implementation so the `create_holding` function will be responsible for setting the `StartRFQArgs` instead of `start_product`. This allows the contract owner to invalidate illogical RFQ parameters in holdings with the `validate_holding` function.
- Add a check that ensures the bid and selection end timestamp is lesser than the start option and product duration (i.e., `args.bid_end < holding.start_option + product.duration && args.selection_end < holding.start_option + product.duration`).

2. Incorrect authentication validation logic

RISK IMPACT: SEVERE	**STATUS: RESOLVED**

Description

The `handle_no_bid_rfq` and `burn_or_exercise_option` functions in `contracts/exotic-holding/src/contract/execute/handle_no_bid_rfq.rs:30` and `contracts/exotic-holding/src/contract/execute/burn_or_exercise_option.rs:26` returns an error if the caller is not the holding ID's creator or the contract owner.

This is incorrect because the validation only passes if the caller is both the holding ID's creator and contract owner, which is not the case. Most of the time, the holding ID creator is not the contract owner.

Recommendation

Consider modifying the implementation so the validation becomes `holding.creator != info.sender && state.owner != info.sender`.

3. Fetched Pyth price may be stale

RISK IMPACT: MODERATE	STATUS: RESOLVED

Description

The `execute_expire_option` function in `contracts/exotic-vanilla/src/contract/execute/expire_option.rs:32` and the `start_product` function in `contracts/exotic-holding/src/contract/execute/start_product.rs:41-44` queries the Pyth price source for the settlement price, and for computing the strike and limit yield.

However, it does not validate that the fetched Pyth price is fresh within a specific period of time with high confidence intervals. As a result, stale prices may be used, leading to incorrect calculations.

Recommendation

Consider validating the fetched Pyth price is fresh with a high confidence interval.

4. Max deposit limit per user can be bypassed

RISK IMPACT: MODERATE	**STATUS: RESOLVED**

Description

In the `deposit` function in `contracts/exotic-holding/src/contract/execute/deposit.rs:44`, the validation checks that the sum of the caller's balance and the deposit amount does not exceed the `holding.max_deposit_per_user` limit. However, this can be bypassed by transferring the tokens to other addresses and then deposit again.

Recommendation

Consider implementing a whitelist where users can deposit and record the user-bought amount in the state.

5. Risk of ownership retention

RISK IMPACT: MODERATE	STATUS: RESOLVED

Description

In the `change_owner` function in `contracts/exotic-holding/src/contract/execute/change_owner.rs:18`, the validation checks that the caller must be the contract instantiator. This is problematic because in the event of admin ownership transfer to a different address (`state.owner`), the contract instantiator, who is also the previous owner, will retain ownership of the contract. This may be problematic if the intention is for the new owner to solely govern the protocol.

Additionally, if the contract instantiator is compromised, the protocol is compromised along with it, although the current contract owner is not the contract instantiator.

Recommendation

Consider using `state.owner` to determine the contract owner.

6. No entry point to call `ConvertAddressPosition`

RISK IMPACT: MODERATE	STATUS: ACKNOWLEDGED

Revision Notes

The team mentioned that if there is no bidder, the holding creator will own the buyer position (and seller position). They won't need to convert the position to a token.

Description

In the `validate_and_get_convert_address_position_recipient` function in `contracts/exotic-vanilla/src/state.rs:357`, if the address in `PositionType::Address` is not validated (`!address_position.validated`), it will validate the caller to be the creator (`sender.eq(&self.creator)`), which is the exotic-holding contract.

However, there is no designated entry point within the exotic-holding contract to call the `ConvertAddressPosition` message in the exotic-vanilla contract.

Recommendation

Consider implementing an entry point for the exotic-holding contract to call the `ConvertAddressPosition` message in the exotic-vanilla contract.

7. Edge case that causes locked funds due to bidders not refunding

RISK IMPACT: MODERATE	STATUS: RESOLVED

Description

The `execute_handle_no_bid` function in `contracts/exotic-rfq/src/contract/execute/handle_no_bid.rs:23` is used to handle edge cases. One of them is if the selection type is `Selection::Manual`, and the `execute_select` function is not called before the `selection_end` timestamp, the `execute_handle_no_bid` function must be called to cancel or renew the auction.

However, canceling or renewing the auction requires all bidders to refund their bids through the `execute_cancel_bid` function in `contracts/exotic-rfq/src/contract/execute/cancel_bid.rs:31`. If not, the cancellation or renewal will fail in `contracts/exotic-rfq/src/state.rs:102`.

If there is a bidder that does not perform the refund (either intentionally or not, e.g., loss of private key), the “position” token denom (`state.asset`) will be stuck in the RFQ contract. This is problematic because this will cause the funds deposited in the exotic-holding contract to be stuck and irrecoverable.

Recommendation

Consider modifying the `execute_cancel_bid` function to let bidders refund their bids within a specified timeline. If it elapsed and there are remaining bids, the `execute_cancel_bid` function should allow the exotic-holding contract to call the message to manually refund the bidders.

8. Symbol length is not validated during contract instantiation

RISK IMPACT: LOW	STATUS: RESOLVED

Description

The `modify_symbols` function in `contracts/exotic-holding/src/contract/execute/modify_symbols.rs:30-32` validates that the symbol length does not exceed five. However, this is not validated when instantiating the contract in `contracts/exotic-holding/src/contract/mod.rs:35`.

Recommendation

Consider validating the symbol length when instantiating the exotic-holding contract.

9. Missing timestamp validation for `DepositorType::Many` holdings

RISK IMPACT: LOW	STATUS: RESOLVED

Description

The `create_holding` function in `contracts/exotic-holding/src/contract/execute/create_holding.rs:79` does not validate that the `end_collection` is larger than the current timestamp when creating a `DepositorType::Many` holding.

If the above happens, the holding cannot be used and will not work correctly.

Recommendation

Consider adding a validation that checks `end_collection > env.block.time`.

10. Misconfiguring minimum and maximum deposit limits prevents users from depositing funds

RISK IMPACT: LOW	**STATUS: RESOLVED**

Description

When creating a new holding in `contracts/exotic-holding/src/contract/execute/create_holding.rs:110`, the `min_deposit_per_user` is not validated to be smaller than the `max_deposit_per_user` and `max_deposit_per_user` is not validated to be larger than `min_deposit_per_user`.

If the above misconfiguration happens, it will cause users to be unable to deposit into holdings due to the validation check in `contracts/exotic-holding/src/contract/execute/deposit.rs:43-44`.

Recommendation

Consider adding validations to ensure `min_deposit_per_user` is lesser than `max_deposit_per_user` and vice versa.

11. Two-step ownership transfer is not implemented

RISK IMPACT: INFORMATIONAL	**STATUS:** RESOLVED

Description

The codebase does not implement two-step ownership transfer in `contracts/exotic-holding/src/contract/execute/change_owner.rs:24`.

Using a two-step ownership transfer mechanism helps provide a window of opportunity for the current owner to cancel the transfer if they did not intend to initiate it or if there were any unintended actions.

As a result, the ownership will be lost and cannot be recovered if transferred to an incorrect address that no one owns.

Recommendation

Consider implementing a two-step ownership transfer that proposes a new owner in the first step and requires the proposed owner to accept it as the second step.

12. Ambiguous error reporting

RISK IMPACT: INFORMATIONAL	STATUS: RESOLVED

Description

The errors returned by the contracts at various points are too generic and do not help in identifying the exact cause of their failure.

Recommendation

Consider enriching the error information to allow for a better understanding.

13. Incorrect owner address attribute emitted

RISK IMPACT: INFORMATIONAL	**STATUS: RESOLVED**

Description

The owner attribute in `contracts/exotic-holding/src/contract/mod.rs:45` emits the address as `info.sender`. This is incorrect because it should be `msg.owner`.

Recommendation

Consider emitting the `owner` attribute value as `msg.owner`.

Document Control

Version	Date	Notes
-	21st February 2024	Security audit commencement date.
0.1	6th March 2024	Initial report with identified findings delivered.
0.5	23rd - 25th March 2024	Fixes remediations implemented and reviewed.
1.0	26 March 2024	Audit completed, final report delivered.

Appendices

A. Appendix – Risk assessment methodology

SCV-Security employs a risk assessment methodology to evaluate vulnerabilities and identified issues. This approach involves the analysis of both the LIKELIHOOD of a security incident occurring and the potential IMPACT if such an incident were to happen. For each vulnerability, SCV-Security calculates a risk level on a scale of 5 to 1, where 5 denotes the highest likelihood or impact. Consequently, an overall risk level is derived from combining these two factors, resulting in a value from 10 to 1, with 10 signifying the most elevated level of security risk

Risk Level	Range
CRITICAL	10
SEVERE	From 9 to 8
MODERATE	From 7 to 6
LOW	From 5 to 4
INFORMATIONAL	From 3 to 1

LIKELIHOOD and **IMPACT** would be individually assessed based on the below:

Rate	LIKELIHOOD	IMPACT
5	**Extremely Likely**	Could result in severe and irreparable consequences.
4	**Likely**	May lead to substantial impact or loss.
3	**Possible**	Could cause partial impact or loss on a wide scale.
2	**Unlikely**	Might cause temporary disruptions or losses.
1	**Rare**	Could have minimal or negligible impact.

B. Appendix - Report Disclaimer

This report should not be regarded as an "endorsement" or "disapproval" of any specific project or team. These reports do not indicate the economics or value of any "product" or "asset" created by a team or project that engages SCV-Security for a security review. The audit report does not make any statements or warranties about the code's utility, safety, suitability of the business model, regulatory compliance of the business model, or any other claims regarding the fitness of the implementation for its purpose or its bug-free status. The audit documentation is intended for discussion purposes only. The content of this audit report is provided "as is," without representations and warranties of any kind, and SCV-Security disclaims any liability for damages arising from or in connection with this audit report. Copyright of this report remains with SCV-Security.

THANK YOU FOR CHOOSING

SCV
SECURITY

scv.services

contact@scv.services