Hubble Kamino
Security Assessment

Sep 06, 2022

Table of Content

Summary

From August 16, 2022 through September 06, 2022, Hubble Protocol engaged PwnedNoMore DAO to review the security of its

- Kamino (aka yvaults) smart contract on Solana (yvaults-program)
    - commit hash **358f5e06c4c7e43e3dd7c527720047a721d8a089**

We conducted this assessment working with the code above and the security researchers mentioned in the title. The Kamino project is a tokenized, self-rebalancing CLMM positions management DEX. It makes CLMM provisioning simple, optimized, auto compoundable and fungible for the user. The user can gain better yields as an LP provider and use their position on Hubble as collateral in many different places.

During the first week, the assessment focused on the smart contract itself. This includes a quick skim of the codebase to get ourselves familiar with its architecture and operating logic, as well as putting specific focus on core functions. We were able to quickly identify two issues and draft PoCs (Proof of Concept) for the attacks. At the end of the first week, we submitted those to the developers. The developers' team confirmed and fixed the bugs quickly.

During the second week, the assessment made a more in-depth and comprehensive review of the contract. To ensure the soundness of the financial model, we did some research on the orca ecosystem, which is the underlying CLMM pool of Kamino. At the end of the week, we found a critical issue and shared the PoC along with a quick writeup with the developers.

During the third week, the assessment mainly focused on evaluating the bots code and carefully reviewing the smart contract code to make sure we did not miss anything obvious.

Throughout the engagement, we identified 5 issues, ranging in severity from low to critical. We discovered:

- Issue related to price validation deficiency
- Issue related to wrong share price calculation formula
- Issue related to imperfect post check
- Issue related to residual balances
- Issue related to insufficient precision

Disclaimer

This audit report should not be used as investment advice.

We make best effort at finding security issues in smart contracts, however we do not provide any guarantees on eliminating all possible security issues. As a single audit-based assessment cannot and should not be considered comprehensive, we always recommend proceeding with several other independent audits and a public bug bounty program to ensure the security of smart contracts.

Project Background

Hubble Kamino is a Tokenized, self-rebalancing CLMM DEX to make CLMM provisioning:

- simple, optimized, auto compoundable such that (rewards + fees + price movement il or ig) > 0
- fungible to be acceptable as collateral in different places

The project has an investing bot which automatically collects fees and rewards, invests and rebalances the number of assets in the pool.

The smart contract is responsible for

- Initialization and upgrade: initialize/update the global config and the strategies
- Positions management: open positions in the orca whirlpools
- Collection: collect fees and rewards
- Rebalance: swap between uneven vaults
- Rewards swap: transform rewards to investable assets
- Deposit: deposit tokens and return shares
- Invest: invest tokens in positions
- Withdraw: burn shares, disinvest, and return tokens to users

The self rebalancing is done in a FSM way and can be seen as the procedure of the following sequence of instructions.

```
InitializeStrategy -> status := Uninitialized
OpenPosition -> status := Active
  loop:
    Deposit
    Invest
    CollectFees
    ExecutiveWithdraw -> status := Rebalancing
    CollectFees
    OpenPosition -> status := Active
```

There are several admin accounts in the protocol

- admin_authority: super admin, can initialize/update the global config, strategies and manage positions.
- actions_authority: an allowed entity (the bot) that has permissions to perform some permissioned actions, such as rebalance.

Audit Scope

The assessment scope contains mainly the smart contracts of the yvaults-program and the yvaults-bot for the Kamino project committed by August 16, 2022. We listed the files we have audited below.

Table 1 Smart Contract Audit Scope

Contract path
programs/yvaults/src/orca_state.rs
programs/yvaults/src/config/global_config_operations.rs
programs/yvaults/src/config/mod.rs
programs/yvaults/src/instructions/collect_rewards.rs
programs/yvaults/src/instructions/withdraw.rs
programs/yvaults/src/instructions/open_liquidity_position.rs
programs/yvaults/src/instructions/orca_swap.rs
programs/yvaults/src/instructions/invest.rs
programs/yvaults/src/instructions/initialize_global_config.rs
programs/yvaults/src/instructions/update_strategy_config.rs
programs/yvaults/src/instructions/swap_uneven_vaults.rs
programs/yvaults/src/instructions/collect_fees.rs
programs/yvaults/src/instructions/update_admin_authority.rs
programs/yvaults/src/instructions/mod.rs
programs/yvaults/src/instructions/deposit.rs
programs/yvaults/src/instructions/swap_rewards.rs
programs/yvaults/src/instructions/update_reward_mapping.rs
programs/yvaults/src/instructions/update_global_config.rs
programs/yvaults/src/instructions/deposit_and_invest.rs
programs/yvaults/src/instructions/initialize_strategy.rs
programs/yvaults/src/instructions/executive_withdraw.rs
programs/yvaults/src/state.rs
programs/yvaults/src/operations/mod.rs
programs/yvaults/src/operations/vault_operations.rs
programs/yvaults/src/operations/effects.rs

programs/yvaults/src/utils/price.rs
programs/yvaults/src/utils/consts.rs
programs/yvaults/src/utils/assertions.rs
programs/yvaults/src/utils/mod.rs
programs/yvaults/src/utils/scope.rs
programs/yvaults/src/utils/enums.rs
programs/yvaults/src/utils/macros.rs
programs/yvaults/src/utils/instructions.rs
programs/yvaults/src/utils/shares.rs
programs/yvaults/src/utils/orca_operations.rs
programs/yvaults/src/utils/types.rs
programs/yvaults/src/utils/math.rs
programs/yvaults/src/utils/clmm_calcs.rs
programs/yvaults/src/lib.rs
programs/yvaults/src/components/withdrawal_cap_operations.rs
programs/yvaults/src/components/mod.rs

Table 2 Bots Audit scope

File path
bots/yvaults-bot/src/yvaults.rs
bots/yvaults-bot/src/rpc.rs
bots/yvaults-bot/src/sysvar.rs
bots/yvaults-bot/src/get_balance.rs
bots/yvaults-bot/src/error.rs
bots/yvaults-bot/src/account.rs
bots/yvaults-bot/src/macros.rs
bots/yvaults-bot/src/lib.rs
bots/yvaults-bot/src/types.rs
bots/yvaults-investing-bot/src/main.rs
bots/yvaults-investing-bot/src/web/mod.rs
bots/yvaults-investing-bot/src/web/routes.rs
bots/yvaults-investing-bot/src/utils.rs

bots/yvaults-instructions/src/macros.rs
bots/yvaults-instructions/src/instruction.rs
bots/yvaults-instructions/src/lib.rs
bots/yvaults-client/src/yvaults.rs
bots/yvaults-client/src/error.rs
bots/yvaults-client/src/macros.rs
bots/yvaults-client/src/lib.rs
bots/yvaults-client/src/client.rs

Vulnerability Dashboard

Vulnerability Summary

Table 3 Vulnerability Summary

Severity	# of Findings
Undetermined	0
Critical	1
High	1
Medium	1
Low	2
Informational	0
Total	5

Category Summary

Table 4 Category Summary

Category	# of Findings
Insufficient Verification of Data Authenticity	1
Business Logic Errors	3
Insufficient Precision or Accuracy of a Real Number	1
Total	5

Key Findings and Recommendations

Table 5 Key Audit Findings

ID	Title	Severity	Type
01	Invest instruction lacks Orca price validation	**Critical**	Insufficient Verification of Data Authenticity
02	Share price calculation loses the rewards	**High**	Business Logic Errors
03	External changes in vault balances break post checks	**Medium**	Business Logic Errors
04	Not checking the rewards owed when closing the position	**Low**	Business Logic Errors
05	Excessive loss due to accuracy error in the rewards swap	**Low**	Insufficient Precision or Accuracy of a Real Number

Invest instruction lacks Orca price validation

Severity: **Critical**
Target: Smart Contract

Difficulty: **Low**
Type: CWE-345

Description

There is no validation on the Orca price in the invest instruction, that will make it possible to steal tokens from vaults by extending the divergence-loss maliciously.

A token price may rise or fall after you deposit it in a liquidity pool, thus generating impermanent loss for the LP providers (IL / or divergence-loss in whirlpools). In Whirlpools, both the yield from trading fees/rewards and divergence loss are amplified.

First, the attacker deposits a large number of token A & B to the vault and gets about fifty percent of the total shares issued back. Then the attacker swaps a large number of token A to token B in the Orca Whirlpool, which makes the Orca price approach the lower price of the strategy position. In this step, the attacker will bear some slippage losses. Then, the attacker invokes the invest instruction to add liquidity at a low price. There is only the check about if the current price is in the strategy position but not the Orca price validation in the invest instruction. And the orca price changes caused by swap occurred after deposit. So all the checks are bypassed. The attacker can then swap more token B back to token A, which will bring the token price back to normal and get more tokens. In the end, the attacker withdraws the remaining tokens from the vault.

```rust
pub fn invest(
    strategy: &mut WhirlpoolStrategy,
    whirlpool: &Whirlpool,
    position: &Position,
    slippage_tolerance_bps: u64,
) -> VaultResult<InvestEffects> {
    let (avail_a, avail_b) = common::available_to_invest(strategy)?;

    let lower_tick_price =
sqrt_price_from_tick_index(position.tick_lower_index);
    let upper_tick_price =
sqrt_price_from_tick_index(position.tick_upper_index);

    let price_lower = calc_price_from_tick_index(position.tick_lower_index);
    let price_upper = calc_price_from_tick_index(position.tick_upper_index);
    msg!("Investing in price range {} - {}", price_lower, price_upper);
    msg!("Available {} - {}", avail_a, avail_b);

    assertions::assert_strategy_in_range(position, whirlpool,
VaultError::CannotInvestOutOfRange)?;
    // no orca price validation here!

    if avail_a == 0 && avail_b == 0 {
        return Err(VaultError::CannotInvestZeroAmount);
    }
```

The core logic of the exploit is that, the investment of the attacker extends the divergence-loss but all the money in the vault shares the loss. And finally, the attacker takes away all the tokens of the divergence-loss.

POC:
https://github.com/tarafans/hubble-poc/blob/main/programs/yvaults/tests/tests_yvaults_d_falldown_w.rs

Impact

Theft of most of the funds in the vault.

Recommendation

Add Orca price validation or add checks about timestamp to make sure the price does NOT change in the same block.

Share price calculation loses the rewards

Severity: **High**
Target: Smart Contract

Difficulty: **Low**
Type: Business Logic Errors

Description

The share price calculation formula loses the rewards which already have been collected but have not been swapped to token a/b yet. So an attacker can collect rewards before deposit. The price of share mint will reduce. And then the attacker swaps all rewards, which will raise the price of his shares. The attacker then withdraws and burns the shares with the high price. the attacker will get more money back immediately.

```
    let available = amounts_available(strategy);
        let invested = amounts_invested(whirlpool, position, mode);
        let fees = pending_fees(position);
        let rewards = pending_rewards(position, whirlpool); // <-- only rewards
owed

        holdings_usd(strategy, available, invested, fees, rewards, prices)
```

POC:
https://github.com/tarafans/hubble-poc/blob/main/programs/yvaults/tests/tests_yvaults_d_s_w.rs

Impact

Theft of a part of yield.

Recommendation

Add the rewards which already have been collected to the holdings calculation.

External changes in vault balances break post checks

Severity: **Medium**
Target: Smart Contract

Difficulty: **Low**
Type: Business Logic Errors

Description

Lack of manual rebase function. An attacker can send a dust of token to the token vault, that will cause all the post checks in the deposit/invest/withdraw to fail. It will DOS the whole program.

POC: https://github.com/tarafans/hubble-poc/blob/main/tests/tests_yvaults_no_rebalance.ts

Impact

Makes most of the interactive functions of the program break down.

Recommendation

Add manual rebase function or update the balances saved in the strategy dynamically.

Not checking the rewards owed when closing the position

Severity: **Low**
Target: Smart Contract

Difficulty: **Low**
Type: Business Logic Errors

Description

When the StrategyStatus is Rebalancing and we are opening a new position. We should check if the old position still has liquidity / fee_owed / reward_owed in it. But the instruction open_liquidity_position only checks the first two items. It will lose the rest of the reward.

```
StrategyStatus::Rebalancing => {
            let old_position_acc = ctx.accounts.old_position_or_rent.clone();
            require!(
                *old_position_acc.owner == orca_whirlpools::ID,
                VaultError::InvalidPositionAccount
            );
            let mut data = &old_position_acc.data.borrow_mut()[..];
            let position: Position = Position::try_deserialize(&mut data)?;
            // not check reward_owed here
            if position.fee_owed_a > 0 || position.fee_owed_b > 0 {
                msg!("Cannot rebalance while fees unharvested");
                return Err(VaultError::UnharvestedAmounts.into());
            }
        }
```

Orca document about position closing:
https://orca-so.gitbook.io/orca-developer-portal/whirlpools/interacting-with-the-protocol/position-management/closing-a-position

Impact

Lose the rest of the reward owed.

Recommendation

The ClosePosition instruction of the program whirlpool will check if the position is empty about liquidity / fee_owed / reward_owed and will burn and close the position token.

Excessive loss due to accuracy error in the rewards swap

Severity: **Low**
Target: Smart Contract

Difficulty: **Medium**
Type: CWE-1339

Description

The token amounts to deposit will be floor scaled when calculating the proportion of the total value of rewards with the function mul_fraction, in the swap_rewards instruction. So, one unit of token A and one unit of token B will be left in the account of the swapper but their value will be swapped to the reward tokens received.

```
let discounted_mv = mul_bps(reward_mv, 10_000 - reward_discount)?;
    // Take from the user only the discounted percentage of token A and B.
    // mul_fraction will floor scale the depositing token amounts
    let token_a_to_deposit = mul_fraction(token_a_in, &discounted_mv,
&deposit_mv)?;
    let token_b_to_deposit = mul_fraction(token_b_in, &discounted_mv,
&deposit_mv)?;

    common::deposit_into_strategy(strategy, token_a_to_deposit,
token_b_to_deposit)?;
    common::remove_reward(strategy, reward_tokens, reward_collateral_id)?;
```

Function mul_fraction:

```
pub fn mul_fraction<T: Copy + CheckedHubbleOps + From<u64>>(
    amount: T,
    numerator: &T,
    denominator: &T,
) -> VaultResult<T> {
    amount.mul(*numerator)?.div(*denominator)
}
```

POC:
https://github.com/tarafans/hubble-poc/blob/main/programs/yvaults/tests/tests_yvaults_swap_rewards_approximate.rs

Impact

Theft of a part of yield.

It’s usually not a problem about this accuracy error in a network with a high tx fee such as ETH. However, Off-by-one bugs are much easier to exploit on Solana compared to other chains, enabled by the relatively low fees on Solana. Solana packing hundreds of swap instructions into a single transaction costs the same flat rate of 5000 lamports (at least prior to the 1.9 per transaction size compute limit update). This is very helpful for the attacker to expand the income.

Recommendation

Use ceiling scale functions for deposit amounts.

Fix Log

Table 6 Fix Log

ID	Title	Severity	Status
01	Invest instruction lacks orca price validation	**Critical**	**Confirmed and Fixed**
02	Share price calculation loses the rewards	**High**	
03	External changes in vault balances break post checks	**Medium**	
04	Not check the rewards owed when closing the position	**Low**	
05	Excessive loss due to accuracy error in the rewards swap	**Low**	

Appendix

Table i Severity Categories

Severity	Description
Undetermined	The extent of the risk can not be determined during this assessment
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or Defense in Depth
Low	The risk is relatively small or is not a risk that the customer indicated as important
Medium	Individual user's information is at risk, exploitation would be bad for client's reputation, moderate financial impact, possibly legal implication for client
High	Affects large number of users, very bad for clients reputation or posses possible financial or legal risk
Critical	Affects large number of users, and is capped at 10% of economic damage

Table ii Difficulty Levels

Difficulty	Description
Undetermined	The difficulty of the exploit is undetermined during this assessment
Low	Commonly exploited, existing tools can be leveraged or can be easily scripted
Medium	Attacker must write a dedicated exploit, or need in depth knowledge of a complex system
High	The attacker must have privileged insider access or need to know extremely complicated technical details or must discover other issues to exploit this

Table iii Vulnerability Classifications

Type	CWE ID
Insufficient Verification of Data Authenticity	CWE-345
Business Logic Errors	CWE-840
Insufficient Precision or Accuracy of a Real Number	CWE-1339