shieldify

Onchain Heroes

Fishing Voyages

SECURITY REVIEW

Date: 30 June 2025

CONTENTS

1. About Shieldify

Positioned as the first hybrid Web3 Security company, Shieldify shakes things up with a unique subscription-based auditing model that entitles the customer to unlimited audits within its duration, as well as top-notch service quality thanks to a disruptive 6-layered security approach. The company works with very well-established researchers in the space and has secured multiple millions in TVL across protocols, also can audit codebases written in Solidity, Vyper, Rust, Cairo, Move and Go.

Learn more about us at shieldify.org.

2. Disclaimer

This security review does not guarantee bulletproof protection against a hack or exploit. Smart contracts are a novel technological feat with many known and unknown risks. The protocol, which this report is intended for, indemnifies Shieldify Security against any responsibility for any misbehavior, bugs, or exploits affecting the audited code during any part of the project's life cycle. It is also pivotal to acknowledge that modifications made to the audited code, including fixes for the issues described in this report, may introduce new problems and necessitate additional auditing.

3. About Onchain Heroes - Fishing Voyages

Onchain Heroes is a fully on-chain, idle RPG with novel on-chain game mechanics.

Introduction to Season 2

The next chapter in the ongoing battle for the multiverse. Picking up where Season 1 left off, Season 2 intensifies the fight as our heroes face an unprecedented threat: Dragma, the World Boss. With new mechanics, deeper strategy, and fresh rewards, this season introduces players to an evolved and expanded gameplay experience.

Season 2 Overview

- Progression Reset: All heroes have been reset to Level 1, providing a fresh start for all players.
- Hero Staking: Deploy your heroes into battle to earn $HERO tokens by defeating enemies and progressing through the season.
- New Damage System: Damage is now determined by a combination of hero level and weapon rarity, directly impacting player rewards.
- Weapon Traits: Weapons now feature Sharpness and Durability, which influence their effective damage output and how often they can be used in battle.
- Two-Phase Gameplay:
  - Phase 1: Available at launch, players will face Dragma's Underlings, preparing for the greater challenge ahead.
  - Phase 2: Unlocked several weeks after launch, this phase will bring the direct confrontation with Dragma and higher-stakes gameplay.
- Season Duration: Season 2 will run for approximately 8 to 12 weeks, offering ongoing challenges and evolving gameplay.

4. Risk Classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4.1 Impact

- **High** – results in a significant risk for the protocol's overall well-being. Affects all or most users
- **Medium** – results in a non-critical risk for the protocol affects all or only a subset of users, but is still unacceptable
- **Low** – losses will be limited but bearable – and covers vectors similar to griefing attacks that can be easily repaired

4.2 Likelihood

- **High** – almost certain to happen and highly lucrative for execution by malicious actors
- **Medium** – still relatively likely, although only conditionally possible
- **Low** – requires a unique set of circumstances and poses non-lucrative cost-of-execution to rewards ratio for the actor

5. Security Review Summary

The security review lasted 3 days, with a total of 24 hours dedicated by 1 researcher from the Shieldify team.

Overall, the code is well-written. The audit report flagged two Medium severity issues, related to by-passing the fishing duration and unintended high-level heroes' reward bonus.

5.1 Protocol Summary

Project Name	**Onchain Heroes - Fishing Voyages**
Repository	och-contracts
Type of Project	GameFi
Audit Timeline	3 days
Review Commit Hash (PR 91)	b80491511b1428bd8cb66797b43f0b589b551f39
Review Commit Hash (PR 94)	734b90e2d0d7f0238afa956e1d97bb0f35f98d16
Review Commit Hash (PR 95)	8c3982e928f57969be302c4d2fe5ba9c74e69e83
Fixes Review Commit Hash	N/A

5.2 Scope

The following smart contracts were in the scope of the security review:

File	nSLOC
src/Fishing/Fishing.sol	273
src/Fishing/OCHItems.sol	115
src/Fishing/FishingProxy.sol	21
src/season2/Blacksmith.sol	115
src/season2/OCHGym.sol	265
src/season2/Blacksmith.sol	162
Total	**951**

6. Findings Summary

The following issues have been identified, sorted by their severity:

- **Mediumh** issues: **2**

ID	Title	Severity	Status
[M-01]	User Can Bypass Fishing Duration and Unstake Immediately Due to Uninitialized `stakeDuration`	Medium	Acknowledged
[M-02]	High-level Heroes Can Receive Guaranteed Bonus Rewards	Medium	Acknowledged

7. Findings

[M-01] User Can Bypass Fishing Duration and Unstake Immediately Due to Uninitialized `stakeDuration`

Severity

Medium Risk

Description

The `Fishing` contract contains an initialization flaw that allows users to bypass the intended fishing duration mechanism. The vulnerability stems from inadequate validation during the staking process, specifically in the `stakeMany()` function. When the contract is deployed, the `stakeDuration` variable within the `FisherCatStorage` struct defaults to zero and remains uninitialized until the owner explicitly calls `setStakeDuration()`. However, the `stakeMany()` function only validates that the zone fee is non-zero through the condition `if (fee == 0) revert RewardsNotSet()` but fails to verify that `stakeDuration` has been properly configured.

This oversight creates a window of opportunity where users can stake their heroes immediately after deployment but before the owner sets the stake duration. The vulnerability manifests during the unstaking process in the `unstakeMany()` function, where the time validation check:

```
if (block.timestamp < (uint256(hero.stakeAt) + uint256($.stakeDuration)))
    revert HeroIsDoingFishing()
```

becomes ineffective. When `stakeDuration` equals zero, this condition simplifies to `if (block.timestamp < hero.stakeAt)`, which will always evaluate to false since `block.timestamp` is guaranteed to be greater than or equal to `hero.stakeAt` at the time of unstaking. Consequently, users can immediately unstake their heroes and claim rewards without waiting for any fishing duration.

Location of Affected Code

File: src/Fishing/Fishing.sol#L207

```
function stakeMany(uint256[] calldata heroIds, uint8 zone) external
   whenNotPausedOrEmergency {
// Checks zone is valid
    if (zone > 2) revert InvalidZone();

    uint256 len = heroIds.length;
    FisherCatStorage storage $ = _getStorage();

    uint256 fee = $.zoneDetails[zone].fee;
    if (fee == 0) revert RewardsNotSet();

//uint256 amount = fee * len;

    if (amount != 0) {
// transfer amount of $HERO20 as entry fee from user to contract
        hero20.transferFrom(msg.sender, address(this), amount);
    }

    for (uint256 i = 0; i < len; ++i) {
        uint256 heroId = heroIds[i];

// Checks hero owner
        if (msg.sender != hero721.ownerOf(heroId)) revert
           CallerIsNotOwner();

// Transfer hero721 to this contract
        hero721.transferFrom(msg.sender, address(this), heroId);

// Update hero information
        $.heroInfo[heroId] =
            HeroInformation({owner: msg.sender, zone: zone, stakeAt:
               uint40(block.timestamp), pendingVRF: 0});

// Emit Staked.
        emit Staked(msg.sender, heroId, zone, fee);
    }
}
```

Impact

Users who discover this flaw can stake their heroes and immediately unstake them to claim rewards, effectively receiving free tokens without respecting the intended time commitment. This creates an unfair advantage over legitimate users who stake after the duration is properly set and must wait the full fishing period.

Recommendation

The vulnerability should be addressed by implementing validation checks in the `stakeMany()` function to ensure all critical parameters are properly initialized before allowing staking operations. The function should include an additional validation condition `if ($.stakeDuration == 0) revert RewardsNotSet()` immediately after the existing fee validation check.

Team Response

Acknowledged.

[M-02] High-level Heroes Can Receive Guaranteed Bonus Rewards

Severity

Medium Risk

Description

The vulnerability exists in the `_getBonusRewards()` function within the VRF callback logic. The function uses a hero's level directly in a probability calculation without implementing proper bounds checking. Specifically, the condition `if (rN < (level << 1))` creates a scenario where heroes with levels above 500 will always receive bonus rewards regardless of the intended randomness.

The problematic logic occurs when determining bonus rewards distribution. The function generates a random number rN between 0 and 999 using
\codex{rN = (randomNumber » 128) % 1000}, then compares it against `level << 1` (level multiplied by 2). Since rN has a maximum value of 999, any hero with a level of 500 or higher will have `level << 1` equal to or greater than 1000, making the condition `rN < (level << 1)` always evaluate to true. This effectively guarantees that high-level heroes will receive bonus rewards on every unstaking operation, breaking the intended probabilistic reward system.

Location of Affected Code

File: src/Fishing/Fishing.sol#L487

```
function _getBonusRewards(uint256 randomNumber, uint256 level, uint256
    zone, uint256 weaponSharedChances) internal pure returns (uint256
    weaponShard, uint256 bonusRewards) {
     uint256 rN = randomNumber % 100;

     if (rN < weaponSharedChances) {
         weaponShard = 1;
     }

     rN = (randomNumber >> 128) % 1000;
// @audit above 500 level they will receive guaranteed  tokens
     if (rN < (level << 1)) {
// if `zone==0` means user get bronze gachaToken
// if `zone==1` means user get s2 training voucher
// if `zone==2` means user get s2 repair voucher
         bonusRewards = zone + 1;
     }
// Return weapon shard and bonus rewards.
     return (weaponShard, bonusRewards);
}
```

Impact

High-level hero owners gain an unfair advantage by receiving guaranteed bonus rewards, which include valuable items such as bronze gacha tokens, training vouchers, and repair vouchers, depending on the fishing zone.

Recommendation

The vulnerability should be addressed by implementing a robust probability calculation system that handles all edge cases while maintaining fair reward distribution across all hero levels.

Team Response

Acknowledged.

shieldify

Thank you!