Pangolin PangoChef Update Audit Report

Jan 25, 2023

Table of Contents

Summary

This report has been prepared for Pangolin PangoChef Update Audit Report smart contract, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Overview

Project Summary

Project Name	**Pangolin PangoChef Update Audit Report**
Codebase	**https://github.com/pangolindex/exchange-contracts**
Commit	**f1d14da512b236a2f316af775f8c9ab07171f006**
Language	**Solidity**

Audit Summary

Delivery Date	**Jan 25, 2023**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**4**

[WP-I1] `emergencyExitLevel2()` should ask for a signed message rather than just a hash

Informational

Issue Description

https://github.com/pangolindex/exchange-contracts/blob/a7fef9af6cf33adc0341716b46148cb108cf5118/contracts/staking-positions/PangoChef.sol#L323-L332

```
function emergencyExitLevel2(uint256 poolId, bytes32 confirmation) external nonReentrant {
    if (confirmation != keccak256(
            abi.encodePacked(
                "I am ready to lose everything in this pool. Let me go.",
                msg.sender
            )
        )
    ) revert UnprivilegedCaller();
    _emergencyExit(poolId, false);
}
```

The current implementation only asks for a hash, which can be precomputed by anyone. Therefore, we cannot guarantee that the user has read and agreed to the message.

Recommendation

Consider asking the user to sign the message properly, and using `ecrecover()` to verify the signer.

Status

ⓘ Acknowledged

[WP-G2] `SafeExternalCalls` is unnecessary, consider moving it into the `PangoChef` contract

Gas

Issue Description

https://github.com/pangolindex/exchange-contracts/blob/a7fef9af6cf33adc0341716b46148cb108cf5118/contracts/staking-positions/PangoChef.sol#L12-L17

```
/** @dev We call this contract with try/catch to figure out if pair is an actual pair. */
contract SafeExternalCalls {
    function getReserveTokenAddresses(address pair) external view returns (address, address) {
        return (IPangolinPair(pair).token0(), IPangolinPair(pair).token1());
    }
}
```

https://github.com/pangolindex/exchange-contracts/blob/a7fef9af6cf33adc0341716b46148cb108cf5118/contracts/staking-positions/PangoChef.sol#L843-L859

```
// Set rewardPair if token is a Pangolin pair which has rewardsToken as one of the reserves.
if (poolType == PoolType.ERC20_POOL) {
    if (tokenOrRecipient.code.length == 0) revert InvalidToken();

    // Gas griefing is not possible as only 63/64 of the gas is forwarded per EIP-150.
    try safeExternalCalls.getReserveTokenAddresses(
        tokenOrRecipient
    ) returns (address token0, address token1) {
        if (factory.getPair(token0, token1) == tokenOrRecipient) {
            if (token0 == address(rewardsToken)) {
                pool.rewardPair = token1;
            } else if (token1 == address(rewardsToken)) {
                pool.rewardPair = token0;
            }
        }
    } catch {}
}
```

The sole goal of `SafeExternalCalls` is to allow the `getReserveTokenAddresses()` function call in `_initializePool()` to fail gracefully.

This can be done by try-catch with an external call to this. Therefore, the standalone `SafeExternalCalls` contract is unnecessary.

Recommendation

Consider moving `getReserveTokenAddresses()` into the `PangoChef` and changing `_initializePool` to:

```
// Set rewardPair if token is a Pangolin pair which has rewardsToken as one of the reserves.
if (poolType == PoolType.ERC20_POOL) {
    if (tokenOrRecipient.code.length == 0) revert InvalidToken();

    // Gas griefing is not possible as only 63/64 of the gas is forwarded per EIP-150.
    try this.getReserveTokenAddresses(
        tokenOrRecipient
    ) returns (address token0, address token1) {
        if (factory.getPair(token0, token1) == tokenOrRecipient) {
            if (token0 == address(rewardsToken)) {
                pool.rewardPair = token1;
            } else if (token1 == address(rewardsToken)) {
                pool.rewardPair = token0;
            }
        }
    } catch {}
}
```

Status

ⓘ **Acknowledged**

[WP-G3] Unnecessary storage load

Gas

Issue Description

https://github.com/pangolindex/exchange-contracts/blob/a7fef9af6cf33adc0341716b46148cb108cf5118/contracts/staking-positions/PangoChef.sol#L1038-L1052

```
function _earned(RewardSummations memory deltaRewardSummations, User storage user)
    private
    view
    returns (uint256)
{
    // Refer to the Combined Position section of the Proofs on why and how this formula works.
    return
        user.lastUpdate == 0
            ? 0
            : user.stashedRewards +
                ((((deltaRewardSummations.idealPosition -
                    (deltaRewardSummations.rewardPerValue * user.lastUpdate)) *
                    user.valueVariables.balance) +
                    (deltaRewardSummations.rewardPerValue * user.previousValues)) / PRECISION);
}
```

Recommendation

```
function _earned(RewardSummations memory deltaRewardSummations, User storage user)
    private
    view
    returns (uint256)
{
    // Refer to the Combined Position section of the Proofs on why and how this formula works.
    uint256 lastUpdate = user.lastUpdate
    return
```

```
        lastUpdate == 0
            ? 0
            : user.stashedRewards +
                ((((deltaRewardSummations.idealPosition -
                    (deltaRewardSummations.rewardPerValue * lastUpdate)) *
                    user.valueVariables.balance) +
                    (deltaRewardSummations.rewardPerValue * user.previousValues))
/ PRECISION);
}
```

Status

✓ Fixed

[WP-N4] Incorrect/Misleading comment

Issue Description

The term ”compoundPool” is no longer synonymous with ”pool zero”.

However, there are still many comments that refer to ”pool zero”, which is incorrect and misleading.

https://github.com/pangolindex/exchange-contracts/blob/a7fef9af6cf33adc0341716b46148cb108cf5118/contracts/staking-positions/PangoChef.sol#L790-L823

```
    /**
     * @notice Private function increment the lock count on pool zero.
     * @param harvestPoolId The identifier of the pool the user is harvesting the rewards from.
     * @param compoundPoolId The identifier of the pool that the rewards will be compounded to.
     */
    function _incrementLock(uint256 harvestPoolId, uint256 compoundPoolId) private {
@@ 796,800 @@
    }

    /**
     * @notice Private function ensure pool zero is not locked and decrement the lock count.
     * @param user The properties of a pool’s user that is decrementing the lock. The user
     *        properties of the pool must belong to the caller.
     * @param withdrawPoolId The identifier of the aforementioned pool.
     */
    function _decrementLock(User storage user, uint256 withdrawPoolId) private {
@@ 810,822 @@
    }
```

Recommendation

Consider changing the `pool zero` in the comments to `compoundPool` .

Status

✓ Fixed

Appendix

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 WatchPug; however, WatchPug 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.

Disclaimer