



Pangolin PangoChef Update Audit Report

Jan 25, 2023





Table of Contents

Summary	2
Overview	3
Issues	4
[WP-I1] <code>emergencyExitLevel12()</code> should ask for a signed message rather than just a hash	4
[WP-G2] <code>SafeExternalCalls</code> is unnecessary, consider moving it into the <code>PangoChef</code> contract	5
[WP-G3] Unnecessary storage load	8
[WP-N4] Incorrect/Misleading comment	10
Appendix	12
Disclaimer	13



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 Issues	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>

```
323 function emergencyExitLevel2(uint256 poolId, bytes32 confirmation) external
    nonReentrant {
324     if (confirmation != keccak256(
325         abi.encodePacked(
326             "I am ready to lose everything in this pool. Let me go.",
327             msg.sender
328         )
329     )
330     ) revert UnprivilegedCaller();
331     _emergencyExit(poolId, false);
332 }
```

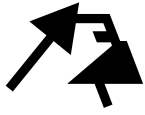
The current implementation only asks for a hash, which can be precomputed by anyone. Therefore, we cannot guarantee that the user has read and agreed to the message.

Recommendation

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

Status

① 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>

```
12  /** @dev We call this contract with try/catch to figure out if pair is an actual  
13  pair. */  
14  contract SafeExternalCalls {  
15      function getReserveTokenAddresses(address pair) external view returns  
16      (address, address) {  
17          return (IPangolinPair(pair).token0(), IPangolinPair(pair).token1());  
18      }  
19  }
```

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

```
843  // Set rewardPair if token is a Pangolin pair which has rewardsToken as one of the  
844  reserves.  
845  if (poolType == PoolType.ERC20_POOL) {  
846      if (tokenOrRecipient.code.length == 0) revert InvalidToken();  
847      // Gas griefing is not possible as only 63/64 of the gas is forwarded per  
848      EIP-150.  
849      try safeExternalCalls.getReserveTokenAddresses(  
850          tokenOrRecipient  
851      ) returns (address token0, address token1) {  
852          if (factory.getPair(token0, token1) == tokenOrRecipient) {  
853              if (token0 == address(rewardsToken)) {  
854                  pool.rewardPair = token1;  
855              } else if (token1 == address(rewardsToken)) {  
856                  pool.rewardPair = token0;  
857              }  
858          }  
859      }
```

```

856         }
857     }
858 } catch {}
859 }

```

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.

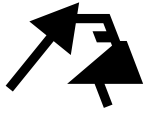
Recommendation

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

```

843 // Set rewardPair if token is a Pangolin pair which has rewardsToken as one of the
844 // reserves.
845 if (poolType == PoolType.ERC20_POOL) {
846     if (tokenOrRecipient.code.length == 0) revert InvalidToken();
847
848     // Gas griefing is not possible as only 63/64 of the gas is forwarded per
849     // EIP-150.
850     try this.getReserveTokenAddresses(
851         tokenOrRecipient
852     ) returns (address token0, address token1) {
853         if (factory.getPair(token0, token1) == tokenOrRecipient) {
854             if (token0 == address(rewardsToken)) {
855                 pool.rewardPair = token1;
856             } else if (token1 == address(rewardsToken)) {
857                 pool.rewardPair = token0;
858             }
859         }
860     } catch {}
861 }

```



Status

 Acknowledged

[WP-G3] Unnecessary storage load

Gas

Issue Description

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

```

1038 function _earned(RewardSummations memory deltaRewardSummations, User storage user)
1039     private
1040     view
1041     returns (uint256)
1042     {
1043         // Refer to the Combined Position section of the Proofs on why and how this
1044         // formula works.
1045         return
1046             user.lastUpdate == 0
1047             ? 0
1048             : user.stashedRewards +
1049               (((deltaRewardSummations.idealPosition -
1050                 (deltaRewardSummations.rewardPerValue * user.lastUpdate)) *
1051                 user.valueVariables.balance) +
1052                 (deltaRewardSummations.rewardPerValue * user.previousValues))
1053               / PRECISION);
1054     }


```

Recommendation

```

1038 function _earned(RewardSummations memory deltaRewardSummations, User storage user)
1039     private
1040     view
1041     returns (uint256)
1042     {
1043         // Refer to the Combined Position section of the Proofs on why and how this
1044         // formula works.
1045         uint256 lastUpdate = user.lastUpdate
1046         return

```



```
1046     lastUpdate == 0
1047     ? 0
1048     : user.stashedRewards +
1049       (((deltaRewardSummations.idealPosition -
1050         (deltaRewardSummations.rewardPerValue * lastUpdate)) *
1051         user.valueVariables.balance) +
1052         (deltaRewardSummations.rewardPerValue * user.previousValues))
1053   / PRECISION);
1053 }
```

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>

```

790     /**
791      * @notice Private function increment the lock count on pool zero.
792      * @param harvestPoolId The identifier of the pool the user is harvesting the
       rewards from.
793      * @param compoundPoolId The identifier of the pool that the rewards will be
       compounded to.
794      */
795     function _incrementLock(uint256 harvestPoolId, uint256 compoundPoolId) private
       {
@@ 796,800 @@
801     }
802
803     /**
804      * @notice Private function ensure pool zero is not locked and decrement the
       lock count.
805      * @param user The properties of a pool's user that is decrementing the lock.
       The user
806      *      properties of the pool must belong to the caller.
807      * @param withdrawPoolId The identifier of the aforementioned pool.
808      */
809     function _decrementLock(User storage user, uint256 withdrawPoolId) private {
@@ 810,822 @@
823     }

```



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

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Smart Contract technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim 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. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, 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, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. 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. 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.