SMART CONTRACT SECURITY AUDIT OF PERPETUEX

ETHERREUM COMPASSS

Summary

Audit Firm Ethereum Compass

Prepared By Dimo Georgiev, Pol Gallardo

Client Firm PerpetuEx

Final Report Date Octuber 10, 2023

Protocol PerpetuEx engaged Ethereum Compass to review the security of its Smart Contract system. From Begining date to Ending date, a team of 2 auditors reviewed the source code in scope. All findings have been recorded in the following report.

Notice that the examined smart contracts are not resistant to external/internal exploit. For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

Table of Contents

Project Overview

Project Summary

Project Name	**PerpetuEx**
Language	Solidity
Codebase	https://github.com/owenThurm/PerpetuEx
Commit	b7f887b

Audit Summary

Delivery Date	**Octuber 10, 2023**
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

Vulnerability Level	**Total**	**Pending**	**Declined**	**Acknowledged**	**Partially Resolved**	**Resolved**
🔴Critical	0	0	0	0	0	0
🟠High	2	2	0	0	0	0
🟡Medium	0	0	0	0	0	0
🟢Low	5	5	0	0	0	0

Audit Scope & Methodology

ID	File	SHA-1 Hash
IPE	contracts/IPerpetuEx.sol	4bf3ea9b168bbd1bd61d8ae8583145b342b867ba
ORA	contracts/Oracle.sol	4bf3ea9b168bbd1bd61d8ae8583145b342b867ba
PEX	contracts/PerpetuEx.sol	4bf3ea9b168bbd1bd61d8ae8583145b342b867ba

Methodology

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 community auditors.

Vulnerability Classifications

Vulnerability Level	Classification
🔴Critical	Easily exploitable by anyone, causing causing loss of assets or undermining of the protocol’s goals.
🟠High	Arduously exploitable by a subset of addresses, causing loss of assets or undermining of the protocol’s goals.
🟡Medium	Inherent risk of future exploits that may or may not impact the smart contract execution.
🟢Low	Minor deviation from best practices.

Findings & Resolutions

ID	Title	Category	Severity	Status
H-01	The function decreaseCollateral ignores fees.	Logic Error	HIGH	Pending
H-02	The function increaseCollateral doesn't update the collateral mapping.	Validation	HIGH	Pending
L-01	Delete the if check at line 429 in _calculateUserLeverage at PerpetuEx.sol.	Logic Error	MEDIUM	Pending
L-02	The function deposit don't have check to min deposit	Validation	MEDIUM	Pending
L-03	Move _updateOpenInterests and delete operation in closePosition outside of if check.	Logic Error	LOW	Pending
L-04	PerpetuEx__NoPositionChosen will never trigger and is not needed	Optimization	LOW	Pending
L-05	Can't decrease collateral & position at the same time	Logic Error	LOW	Pending

H-01 | The function decreaseCollateral ignores fees.

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf451 6204aab/src/PerpetuEx.sol#L246C5-L261C6

Description

In the `decreaseCollateral` function, it ignores the fees. It checks the user's leverage and considers the gains and losses, but completely disregards the fee calculation.

Recommendation

Perform the fee calculation within the `decreaseCollateral` function.

Resolution

H-02 | The function increaseCollateral doesn’t update the collateral mapping

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf451 6204aab/src/PerpetuEx.sol#L238C5-L244C6

Description

They are keeping the collateral of a user in both the position struct, as well as the collateral mapping. They are using both down the code and in this example, the collateral mapping is not updated.

Recommendation

They should update the collateral mapping.

Resolution

L-01 | Delete the if check at line 429 in _calculateUserLeverage at PerpetuEx.sol.

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf4516204aab/src/PerpetuEx.sol#L429C6-L441C10

Description

Delete the if check at line 429 in _calculateUserLeverage at PerpetuEx.sol. It's basically covered by the next if check

Recommendation

Delete the if check at line 429 in _calculateUserLeverage at PerpetuEx.sol.

Resolution

L-02 | The function deposit don't have check to min deposit.

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf4516204aab/src/PerpetuEx.sol#L97C5-L101C6

Description

The function deposit don't have check to min deposit

Recommendation

It is advisable to add the requirement that the deposited value for the user must be greater than 0.

Resolution

L-03 | Move _updateOpenInterests and delete operation in closePosition outside of if check.

On line

163 ->

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf4516204aab/src/PerpetuEx.sol#L163

172 ->

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf4516204aab/src/PerpetuEx.sol#L172

Description

We are doing the same operation in both cases of the if statement. Let's extract it outside the common logic outside.

Recommendation

We recommend, for a more optimized logic, to extract the mentioned points outside of the "if" statement, so we don't have to repeat them in each case.

Resolution

L-04 | PerpetuEx__NoPositionChosen will never trigger and is not needed.

https://github.com/owenThurm/PerpetuEx/blob/b7f887b73bd0c1ca3b9410a28a2bdf4516204aab/src/PerpetuEx.sol#L348

Description

PerpetuEx__NoPositionChosen will never trigger and is not needed

Recommendation

In solidity, there are only 2 possible states for a bool variable -> true/false. There aren't any other possibilities, that could lead to PerpetuEx__NoPositionChosen to be thrown

Resolution

L-05 | Can't decrease collateral & position at the same time.

Description

Can't decrease collateral & position at the same time.

Recommendation

As per requirements in mission 2, the contract should expose functionality to decrease size and collateral at the same time.

Resolution

Disclaimer