HALBORN

Aura Finance - LP Migration Deployment

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **December 9th, 2023 - December 16th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	12/12/2022	Miguel Jalon
0.2	Final Draft	12/16/2022	Miguel Jalon
0.3	Draft Review	12/16/2022	Roberto Reigada
0.4	Draft Review	12/16/2022	Piotr Cielas
0.5	Draft Review	12/16/2022	Gabi Urrutia
1.0	Remediation Plan	12/22/2022	Miguel Jalon
1.1	Remediation Plan Review	12/22/2022	Roberto Reigada

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Miguel Jalon	Halborn	Miguel.Jalon@halborn.com
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com
Piotr Cielas	Halborn	Piotr.Cielas@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Aura Finance is a protocol that allows BAL stakers and Balancer liquidity providers to boost their yield earnings and governance power in an easy-to-use platform.

Aura Finance engaged Halborn to conduct a security audit of the new changes on their smart contracts within the following pull "Pull 2 - Booster Migration" beginning on December 9th, 2023 and ending on December 16th, 2023. The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided one week for the engagement and assigned a full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some improvements to reduce the likelihood and impact of risks, which were acknowledged by the Aura Finance team.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover

flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture and purpose
- Smart contract manual code review and walkthrough
- Graphing out functionality and contract logic/connectivity/functions (solgraph)
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes
- Manual testing by custom scripts
- Scanning of solidity files for vulnerabilities, security hotspots or bugs. (MythX)
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Testnet deployment (Brownie, Remix IDE)

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.

1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

IN-SCOPE:
The security assessment was scoped to the following Aura Smart Contracts and Convex Platform Smart Contracts:

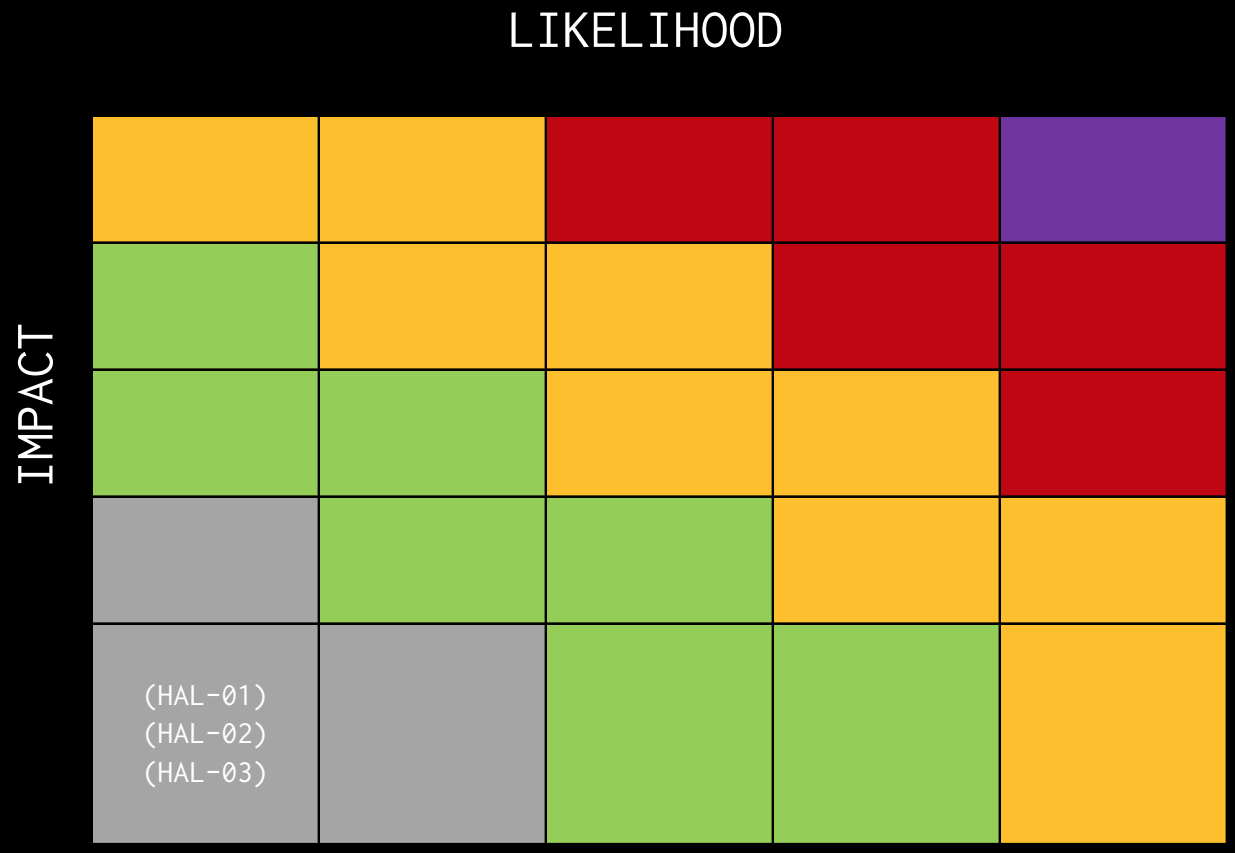
- MasterChefRewardHook.sol
- AuraClaimZap.sol
- GaugeMigrator.sol
- PoolMigrator.sol
- BaseRewardPool.sol
- ExtraRewardStashV3.sol
- Booster.sol

Aura Smart Contracts Commit ID: c18e9fd6aeaf7316c440c17f1b760130113d5570

Convex Platform Smart Contracts Commit ID: 9b93a3c976ddcc236fe23ed7cae67c8ec02f145d

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	0	3



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - POSSIBLE LOSS OF OWNERSHIP	Informational	ACKNOWLEDGED
HAL02 - ZERO ADDRESS NOT CHECKED	Informational	ACKNOWLEDGED
HAL03 - USE ++I INSTEAD OF I++ IN LOOPS FOR GAS OPTIMIZATION	Informational	ACKNOWLEDGED

FINDINGS & TECH DETAILS

3.1 (HAL-01) POSSIBLE LOSS OF OWNERSHIP - INFORMATIONAL

Description:

Ownership of the contract can be lost, as the MasterChefRewardHook contract is Ownable and its ownership can be transferred in a single-step process. If it is not verified that the address to which it is changed to is active or is willing to act as the owner.

Code Location:

Listing 1: MasterChefRewardHook.sol

```
62     function transferOwnership(address newOwner) public virtual
↳ onlyOwner {
63         require(newOwner != address(0), "Ownable: new owner is the
↳  zero address");
64         _transferOwnership(newOwner);
65     }
66
67     function _transferOwnership(address newOwner) internal virtual
↳  {
68         address oldOwner = _owner;
69         _owner = newOwner;
70         emit OwnershipTransferred(oldOwner, newOwner);
71     }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

The ownership transfer process could be split into two-step. The first one is to call requestTransferOwnership function, which proposes a new

owner for the protocol, and the second, the new owner accepts the proposal by calling the acceptsTransferOwnership function.

Remediation Plan:

ACKNOWLEDGED: The Aura Finance team acknowledged this issue.

3.2 (HAL-02) ZERO ADDRESS NOT CHECKED - INFORMATIONAL

Description:

In different sections of the code the address variables are not checked not to point to the zero address.

Code Location:

- In the constructor of the AuraLocker.sol contract
- In the constructor of the MasterChefRewardHook.sol contract
- In the constructor of the Booster.sol contract
- In the constructor of the PoolMigrator.sol contract
- In the constructor of the GaugeMigrator.sol contract
- In the constructor of the BaseRewardPool.sol contract
- In the constructor of the ExtraRewardStashV3.sol contract

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

When setting an address variable, always make sure the value is not zero.

Remediation Plan:

ACKNOWLEDGED: The Aura Finance team acknowledged this issue.

3.3 (HAL-03) USE ++I INSTEAD OF I++ IN LOOPS FOR GAS OPTIMIZATION - INFORMATIONAL

Description:

In a lot of loops throughout the code, the variable `i` is incremented using `i++`. In loops, using `++i` costs less gas per iteration than `i++`. This also affects variables incremented inside the loop code block.

Risk Level:

Likelihood - 1
Impact - 1

Code Location:

- `updateReward()` function at the `AuraLocker.sol` contract
- `getReward()` function at the `AuraLocker.sol` contract
- `getReward()` second function at the `AuraLocker.sol` contract
- `_processExpiredLocks()` function at the `AuraLocker.sol` contract
- `lockedBalances()` function at the `AuraLocker.sol` contract
- `claimableRewards()` function at the `AuraLocker.sol` contract
- `claimRewards()` function for the claim from main curve LP pools at the `AuraClaimZap.sol` contract
- `claimRewards()` function for the claim from extra rewards at the `AuraClaimZap.sol` contract
- `claimRewards()` function for the claim from multi reward token contract at the `AuraClaimZap.sol` contract
- `earmarkRewards()` function at the `BoosterHelper.sol` contract
- `processIdleRewards()` function at the `BoosterHelper.sol` contract
- `migrate()` function at the `PoolMigrator.sol` contract
- `_processStake()` function at the `BaseRewardPool.sol` contract
- `withdraw()` function at the `BaseRewardPool.sol` contract
- `_withdrawAndUnwrapTo()` function at the `BaseRewardPool.sol` contract

- getReward() function at the BaseRewardPool.sol contract
- shutdownSystem() function at the Booster.sol contract
- voteGaugeWeight() function at the Booster.sol contract
- checkForNewRewardTokens() function at the ExtraRewardStashV3.sol contract
- processStash() function at the ExtraRewardStashV3.sol contract
- setUsedAddress() function at the PoolManagerSecondaryProxy.sol contract

Recommendation:

It is recommended to use ++i instead of i++ to increment the value of a uint variable inside a loop. This also applies to variables declared inside the for loop, but does not apply outside of loops.

Remediation Plan:

ACKNOWLEDGED: The Aura Finance team acknowledged this issue.

MANUAL TESTING

MANUAL TESTING

Halborn performed manual tests on the following contracts:

Aura smart contracts:

- MasterChefRewardHook.sol
- AuraClaimZap.sol
- GaugeMigrator.sol
- PoolMigrator.sol

Aura convex-platform smart contracts:

- BaseRewardPool.sol
- ExtraRewardStashV3.sol
- Booster.sol

4.1 SUMMARY OF CHANGES INTRODUCED

Core changes:

- Booster.sol
  - Increase MaxFees from 2500 to 4000
  - Adds support for L2 Fees
  - Adds `getRewardMultipliers` for increasing or decreasing AURA rewards per pool
  - Adds reentrancy guards to improve security
  - Patch logic earmarkRewards/claimFees
- BaseRewardPool.sol
  - Add transfer event to reward pools for analytics
  - Removes `donate` function
- ExtraRewardStashV3.sol
  - Adds validation on `setToken` to only want add rewards that are not CRV

Chef and Peripheral changes:

- MasterChefRewardHook.sol
  - Use of library SafeERC20 and use of `SafeTransfer()` to transfer tokens
- AuraClaimZap.sol
  - Signature change, it adds struct as param and new `depositCvxCrvMaxAmount`, to claim and lock cvxCRV
- PoolMigrator.sol : New smart contract, allows to migrate positions from one booster to the other.

4.2 SCENARIOS TESTED

The following unit and integration tests have been performed:

- Migrate LP tokens from Booster V1 Pool to a Booster V2 Pool
- New struct in AuraClaimZap.sol and the use of it
- Checking the use of safeTransfer() in the MasterChefRewardHook contract
- Options.UseAllWalletFunds for _claimExtras() function in AuraClaimZap
- Check the stakeFor functionality in AuraClaimZap.sol

Migrate LP tokens from Booster V1 Pool to a Booster V2 Pool:

To verify that the migrator pool performs the migration correctly, a pool that was in both Booster versions was selected from the Ethereum Mainnet.

We consider the following Boosters:
Booster V1: 0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10
Booster V2: 0xA57b8d98dAE62B26Ec3bcC4a365338157060B234

We found that poolId 0 in V1 and poolId 26 in V2 share vault:
0x08b8a86B9498AC249bF4B86e14C5d4187085a239

We reviewed that the migration works properly and the tokens are transferred as expected. We confirmed users with no LP tokens to migrate are not able to abuse this functionality.

Listing 2: Brownie Manual Testing 1

```
output.greenn("--- DEPLOYMENT OF POOLMIGRATOR ---")
output.greenn("EXECUTING ---> poolMigrator = PoolMigrator.deploy
↳ ('0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10', '0
↳ xA57b8d98dAE62B26Ec3bcC4a365338157060B234', {'from': owner})")
poolMigrator = PoolMigrator.deploy("0
↳ x7818A1DA7BD1E64c199029E86Ba244a9798eEE10","0
↳ xA57b8d98dAE62B26Ec3bcC4a365338157060B234", {'from': owner})

output.greenn("--- LP MIGRATION FROM POOLID 0 IN V1 TO POOLID 26
↳ ---")
output.greenn("EXECUTING ---> poolMigrator.migrate([0], [26], [10
↳ _000000000000000000], {'from': user1})")
tx = poolMigrator.migrate([0], [26], [10_000000000000000000], {'
↳ from': user1})
```

```
################################################################################################
################################################################################################
##############################    TEST 1: MIGRATE LP TOKENS    #################################
################################################################################################
################################################################################################

output.greenn("--- DEPLOYMENT OF POOLMIGRATOR ---")
output.greenn("EXECUTING ---> poolMigrator = PoolMigrator.deploy('0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10', '0xA57b8d9
poolMigrator = PoolMigrator.deploy("0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10","0xA57b8d98dAE62B26Ec3bcC4a36533815706082

output.greenn("--- LP MIGRATION FROM POOLID 0 IN V1 TO POOLID 26 ---")
output.greenn("EXECUTING ---> poolMigrator.migrate([0], [26], [10_000000000000000000], {'from': user1})")
tx = poolMigrator.migrate([0], [26], [10_000000000000000000], {'from': user1})


Transaction sent: 0x8bcf60b07bd09a3b9930308b4a2b9508a3ca848131fcb159f17bb4e738c42151
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 104
  RewardContract.constructor confirmed   Block: 16218692   Gas used: 98565 (0.03%)
  RewardContract deployed at: 0x55F2D8d2A2cdA5515f422A1516C388c4085E11e7

Transaction sent: 0x17cd9e79158f58677eb0593197b6d1d20118f1fc16917e4b6e6a20acd9d4ba14
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 105
  ExtraRewardContract.constructor confirmed   Block: 16218693   Gas used: 92727 (0.03%)
  ExtraRewardContract deployed at: 0xa5c3C11eeed9af949f15C286229bdb21BDC29b10

Transaction sent: 0x6fbdf034b4347cd3e58cefec6937db9acf8655dfd169f78902a243f19063fa74
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 106
  TokenRewardContract.constructor confirmed   Block: 16218694   Gas used: 98985 (0.03%)
  TokenRewardContract deployed at: 0xc41E3149c14ff90063f3BF48956DC6Afef554E84

--- DEPLOYMENT OF POOLMIGRATOR ---
EXECUTING ---> poolMigrator = PoolMigrator.deploy('0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10', '0xA57b8d98dAE62B26Ec3bcC
Transaction sent: 0xa695e196dd9e6db73face137f47c7e0bf022e641dfdc6d86b295137f208c35b2
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 107
  PoolMigrator.constructor confirmed   Block: 16218695   Gas used: 685932 (0.23%)
  PoolMigrator deployed at: 0x3d6740d79B1FdFA0A80e9CB5756aA24C8cB9fD89

--- LP MIGRATION FROM POOLID 0 IN V1 TO POOLID 26 ---
EXECUTING ---> poolMigrator.migrate([0], [26], [10_000000000000000000], {'from': user1})
Transaction sent: 0xa2022d46a25540186d84702b9ebe6eab04a04fd482359c6ac0c120688a9df485
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 47
  PoolMigrator.migrate confirmed (ERC4626: withdrawal amount exceeds allowance)   Block: 16218696   Gas used: 49600 (0.02
```

Checking the use of safeTransfer in the MasterChefRewardHook contract:

To verify if SafeTransfer works as expected, we checked that the tokens in the MasterChefRewardHook contract were properly transferred to the stash after executing the onRewardClaim() function.

Listing 3: Brownie Manual Testing 2

```
output.greenn("------ DEPLOYMENT OF REWARD TOKEN MOCK ------")
output.greenn("EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000, {'from': owner})")
rewardToken = RewardTokenMock.deploy("RewardToken", "RWT", owner, 1000_000000000000000, {'from': owner})

output.greenn("------ DEPLOYMENT OF MASTER CHEF HOOK ------")
output.greenn("EXECUTING ---> poolMigrator = PoolMigrator.deploy ('0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10', '0 xA57b8d98dAE62B26Ec3bcC4a365338157060B234', {'from': owner})")
masterchefHook = MasterChefRewardHook.deploy(stash, "0 x1ab80F7Fb46B25b7e0B2cfAC23Fc88AC37aaf4e9", rewardToken, {'from': owner})

output.greenn("------ TRANSFER OF 1 TOKEN TO THE HOOK ------")
output.greenn("EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000, {'from': owner})")
transfer = rewardToken.transfer(masterchefHook, 1 _000000000000000, {'from': owner})

output.greenn("------ DEPLOYMENT OF REWARD TOKEN MOCK ------")
output.greenn("EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000, {'from': owner})")
claim = masterchefHook.onRewardClaim({'from': stash})

claim.call_trace()

output.greenn("------ CHECK STASH BALANCE ------")
output.greenn("EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000, {'from': owner})")
output.greenn("EXPECTED                    -->
1000000000000000000")
output.yelloww("rewardToken.balanceOf(stash) --> " + str( rewardToken.balanceOf(stash)))
```

```
------ DEPLOYMENT OF REWARD TOKEN MOCK ------
EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000000, {'from': owner})
Transaction sent: 0x611b790eb9d982bb5ed3cbe8ce0e81fe7e771aeba3c79cad03f2a78540291a34
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 20
  RewardTokenMock.constructor confirmed   Block: 16218563   Gas used: 810731 (0.27%)
  RewardTokenMock deployed at: 0x34050d781b5e31eF784797820b595655F72a4783

------ DEPLOYMENT OF MASTER CHEF HOOK ------
EXECUTING ---> poolMigrator = PoolMigrator.deploy('0x7818A1DA7BD1E64c199029E86Ba244a9798eEE10', '0xA57b8d98dAE62B26Ec3bcC
Transaction sent: 0x348b1dbf7139b59eee3af70433899b957ff80fe1e294ffad498cce3e31439309
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 21
  MasterChefRewardHook.constructor confirmed   Block: 16218564   Gas used: 673700 (0.22%)
  MasterChefRewardHook deployed at: 0xdcD9A78Eb544D892068C9AE5c8eE8dc123c19563

------ TRANSFER OF 1 TOKEN TO THE HOOK ------
EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000000, {'from': owner})
Transaction sent: 0x45d53a31bb5c8170e1f421d419b69807ae7a04928441d59af4e583df7efe97bc
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 22
  RewardTokenMock.transfer confirmed   Block: 16218565   Gas used: 51069 (0.02%)

------ DEPLOYMENT OF REWARD TOKEN MOCK ------
EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000000, {'from': owner})
Transaction sent: 0xd18c59826cc7da4038f29a62b38146cbbbaaa94593b0a8320279f443edfada93
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 8
  MasterChefRewardHook.onRewardClaim confirmed   Block: 16218566   Gas used: 77323 (0.03%)

Call trace for '0xd18c59826cc7da4038f29a62b38146cbbbaaa94593b0a8320279f443edfada93':
Initial call cost  [1864 gas]
MasterChefRewardHook.onRewardClaim  0:2587  [4283 / 30459 gas]
├── <UnknownContract>.0xddd5e1b2  [CALL]  110:1636  [13226 / 9032 gas]
│   ├── <UnknownContract>.0x70a08231  [STATICCALL]  343:487  [2044 gas]
│   ├── <UnknownContract>.0x70a08231  [STATICCALL]  673:804  [2002 gas]
│   └── <UnknownContract>.0xa9059cbb  [CALL]  1100:1397  [-8240 gas]
├── RewardTokenMock.balanceOf  [STATICCALL]  1682:1814  [2007 gas]
└── RewardTokenMock.transfer  [CALL]  2140:2433  [1162 / 15137 gas]
    └── ERC20.transfer  2242:2404  [21922 / 13975 gas]
        └── ERC20._transfer  2250:2357  [-7947 gas]
------ CHECK STASH BALANCE ------
EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000000, {'from': owner})
EXPECTED                  --> 1000000000000000000
rewardToken.balanceOf(stash) --> 1000000000000000000
```

AuraClaimZap changes:

We verified that the changes introduced in this contract work properly and as expected, namely:

- New struct in AuraClaimZap.sol and the use of it
- Options.UseAllWalletFunds for _claimExtras() function
- Check the stakeFor functionality

Listing 4: Brownie Manual Testing 3

```
output.greenn("------ DEPLOYMENT OF MOCKS (CURVE TOKEN, CONVEX TOKEN, CONVEX_CURVE TOKEN) ------")
output.greenn("EXECUTING ---> CurveToken.deploy('CurveToken', 'CRV', user1, 1000_000000000000000000, {'from': owner})")
crv = CurveToken.deploy('CurveToken', 'CRV', user1, 1000_000000000000000000, {'from': owner})
output.greenn("EXECUTING ---> cvx = ConvexToken.deploy('ConvexToken', 'CVX', user1, 1000_000000000000000000, {'from': owner})")
cvx = ConvexToken.deploy("ConvexToken", "CVX", user1, 1000_000000000000000000, {'from': owner})
output.greenn("EXECUTING ---> ConvexCurveToken.deploy('ConvexCurveToken', 'cvxCrv', user1, 1000_000000000000000000, {'from': owner})")
cvxCrv = ConvexCurveToken.deploy("ConvexCurveToken", "cvxCrv", user1, 1000_000000000000000000, {'from': owner})


output.greenn("------ DEPLOYMENT OF REWARD MOCKS ------")
output.greenn("EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000000, {'from': owner})")
rewardToken = RewardTokenMock.deploy("RewardToken", "RWT", owner, 1000_000000000000000000, {'from': owner})


output.greenn("------ DEPLOYMENT OF AURACLAIMZAP ------")
output.greenn("EXECUTING ---> AuraClaimZap.deploy(crv, cvx, cvxCrv, '0x8014595F2AB54cD7c604B00E9fb932176fDc86Ae', '0x3Fe65692bfCD0e6CF84cB1E7d24108E434A7587e', '0xD18140b4B819b895A3dba5442F959fA44994AF50', {'from': owner})")
auraClaimZap = AuraClaimZap.deploy(crv, cvx, cvxCrv, '0x8014595F2AB54cD7c604B00E9fb932176fDc86Ae', '0
```

```
↳ x3Fe65692bfCD0e6CF84cB1E7d24108E434A7587e', '0
↳ xD18140b4B819b895A3dba5442F959fA44994AF50', {'from': owner})
18
19
20 output.greenn("------ APPROVALS OF CRV, CVX, CVX_CRV FROM USER1 TO
↳  AURACLAIMZAP ------")
21 output.greenn("EXECUTING ---> crv.approve(auraClaimZap, 1000
↳ _000000000000000000, {'from': user1})")
22 approval1 = crv.approve(auraClaimZap, 1000_000000000000000000, {'
↳ from': user1})
23 output.greenn("EXECUTING ---> cvx.approve(auraClaimZap, 1000
↳ _000000000000000000, {'from': user1})")
24 approval2 = cvx.approve(auraClaimZap, 1000_000000000000000000, {'
↳ from': user1})
25 output.greenn("EXECUTING ---> cvxCrv.approve(auraClaimZap, 1000
↳ _000000000000000000, {'from': user1})")
26 approval3 = cvxCrv.approve(auraClaimZap, 1000_000000000000000000,
↳ {'from': user1})
27
28
29 output.greenn("------ DEPOSITING CVX_CRV TOKENS IN AURACLAIMZAP
↳ ------")
30 output.greenn("EXECUTING ---> cvxCrv.transfer(auraClaimZap, 1
↳ _000000000000000000, {'from': user1})")
31 deposit = cvxCrv.transfer(auraClaimZap, 1_000000000000000000, {'
↳ from': user1})
32
33 output.greenn("------ SETTING APPROVALS FOR CONTRACT SPENDING
↳ ------")
34 output.greenn("EXECUTING ---> auraClaimZap.setApprovals({'from':
↳ owner})")
35 approvals = auraClaimZap.setApprovals({'from': owner})
36
37 output.greenn("------ CLAIM ------")
38 output.greenn("EXECUTING ---> auraClaimZap.claimRewards([
↳ rewardContracts], [extraRewardContracts], [tokenRewardContracts],
↳ [rewardToken], (100_000000000000000000, 10000000000000000, 10
↳ _000000000000000000, 10_000000000000000000), 16, {'from': user1})"
↳ )
39 claim = auraClaimZap.claimRewards([rewardContracts], [
↳ extraRewardContracts], [tokenRewardContracts], [rewardToken], (100
↳ _000000000000000000, 10000000000000000, 10_000000000000000000, 10
↳ _000000000000000000), 16, {'from': user1})
40
```

```
------ DEPLOYMENT OF MOCKS (CURVE TOKEN, CONVEX TOKEN, CONVEX_CURVE TOKEN) ------
EXECUTING ---> CurveToken.deploy('CurveToken', 'CRV', user1, 1000_000000000000000000, {'from': owner})
Transaction sent: 0x1a4d0e84c23c2420ecf6d4b1c215c3cdd6004e9af4895480fd70d0da3a4fbe51
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 98
  CurveToken.constructor confirmed   Block: 16218681   Gas used: 810719 (0.27%)
  CurveToken deployed at: 0x9299fd0741B8eCbd108167B523C949Ba36C8F187

EXECUTING ---> cvx = ConvexToken.deploy('ConvexToken', 'CVX', user1, 1000_000000000000000000, {'from': owner})
Transaction sent: 0x9e0fba7bec92e46752c8823f1c7dbb5b7b03a3a14d6cb72ba1bf80b5a031c265
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 99
  ConvexToken.constructor confirmed   Block: 16218682   Gas used: 810731 (0.27%)
  ConvexToken deployed at: 0x854A66F6d049b3F8e86dE7AE33B001eD5115239e

EXECUTING ---> ConvexCurveToken.deploy('ConvexCurveToken', 'cvxCrv', user1, 1000_000000000000000000, {'from': owner})
Transaction sent: 0x55866d7e9be3ba74f459182191fb834feb734adda9db67d8c6afa88f13090ef7
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 100
  ConvexCurveToken.constructor confirmed   Block: 16218683   Gas used: 810815 (0.27%)
  ConvexCurveToken deployed at: 0x491C5A96D260ffE5a87b2e2748c837Ff13f504f0

------ DEPLOYMENT OF REWARD MOCKS ------
EXECUTING ---> RewardTokenMock.deploy('RewardToken', 'RWT', owner, 1000_000000000000000000, {'from': owner})
Transaction sent: 0x64615872ece895ef2daade51e604167d193827f6ba84cc1ce1ee4d4a5a94ed06
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 101
  RewardTokenMock.constructor confirmed   Block: 16218684   Gas used: 810731 (0.27%)
  RewardTokenMock deployed at: 0xbCD10fe0068A435606A4371Cae819F416708a419

------ DEPLOYMENT OF AURACLAIMZAP ------
EXECUTING ---> AuraClaimZap.deploy(crv, cvx, cvxCrv, '0x8014595F2AB54cD7c604B00E9fb932176fDc86Ae', '0x3Fe65692bfCD0e6CF84
Transaction sent: 0x9fc205b11f82e0d89031a09118839b1b4187b258f9e3fc9a77677e23458b44cf
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 102
  AuraClaimZap.constructor confirmed   Block: 16218685   Gas used: 1202016 (0.40%)
  AuraClaimZap deployed at: 0xf75E70c98BFa1deE35B80FAEc6A1B7517e8be0A5

------ APPROVALS OF CRV, CVX, CVX_CRV FROM USER1 TO AURACLAIMZAP ------
EXECUTING ---> crv.approve(auraClaimZap, 1000_000000000000000000, {'from': user1})
Transaction sent: 0xf8884d09ebde50820eb37ae914a577d80198419326d9e9760bdb00c0355b00de
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 42
  CurveToken.approve confirmed   Block: 16218686   Gas used: 44187 (0.01%)

EXECUTING ---> cvx.approve(auraClaimZap, 1000_000000000000000000, {'from': user1})
Transaction sent: 0x466a2ef0c709d94b68d99da32d12020145957a8e1eada247ede91519e6aef61f
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 43
  ConvexToken.approve confirmed   Block: 16218687   Gas used: 44187 (0.01%)

EXECUTING ---> cvxCrv.approve(auraClaimZap, 1000_000000000000000000, {'from': user1})
Transaction sent: 0x657271bed4b910881107ba05ce8a69d69cc99b9c7c7cc3f3069d6feccf5295ef
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 44
  ConvexCurveToken.approve confirmed   Block: 16218688   Gas used: 44187 (0.01%)

------ DEPOSITING CVX_CRV TOKENS IN AURACLAIMZAP ------
EXECUTING ---> cvxCrv.transfer(auraClaimZap, 1_000000000000000000, {'from': user1})
Transaction sent: 0x139cf46b74b47093985b019d6bf0151c29daf88a4a1e1aaffed81d6403c77088
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 45
  ConvexCurveToken.transfer confirmed   Block: 16218689   Gas used: 51069 (0.02%)

------ SETTING APPROVALS FOR CONTRACT SPENDING ------
EXECUTING ---> auraClaimZap.setApprovals({'from': owner})
Transaction sent: 0xb475a88043583917f9172c58f8ea56f3934111108d4add941adc7b3a7c8d0a7c
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 103
  AuraClaimZap.setApprovals confirmed   Block: 16218690   Gas used: 65889 (0.02%)

------ CLAIM ------
EXECUTING ---> auraClaimZap.claimRewards([rewardContracts], [extraRewardContracts], [tokenRewardContracts], [rewardToken]
Transaction sent: 0xb469adc7669de2f5bc4a3607ee85a1c8aefca20a47bde85f440d53747752fa2e
  Gas price: 0.0 gwei   Gas limit: 300000000   Nonce: 46
  AuraClaimZap.claimRewards confirmed   Block: 16218691   Gas used: 85660 (0.03%)
```

The new struct works as expected and sends the parameters within the `_claimExtras()` internal function. The `UseAllWalletFunds` option was correctly processed as the `msg.sender` of the transaction was able to deposit their entire available CRV balance, and the contract can stake the CVX_CRV tokens on their behalf with the `StakeFor()` function.

AUTOMATED TESTING

5.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contracts. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified all the contracts in the repository and was able to compile them correctly into their ABI and binary formats, Slither was run on the all-scoped contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

MasterChefRewardHook.sol

```
MasterChefRewardHook.deposit(address) (contracts/chef/MasterChefRewardHook.sol#28-33) ignores return value by IERC20(siphonToken).approve(chef,type()(uint256).max) (contracts/chef/MasterChefRewardHook
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

MasterChefRewardHook.constructor(address,address,address)._stash (contracts/chef/MasterChefRewardHook.sol#18) lacks a zero-check on :
        - stash = _stash (contracts/chef/MasterChefRewardHook.sol#19)
MasterChefRewardHook.constructor(address,address,address)._chef (contracts/chef/MasterChefRewardHook.sol#18) lacks a zero-check on :
        - chef = _chef (contracts/chef/MasterChefRewardHook.sol#20)
MasterChefRewardHook.constructor(address,address,address)._rewardToken (contracts/chef/MasterChefRewardHook.sol#18) lacks a zero-check on :
        - rewardToken = _rewardToken (contracts/chef/MasterChefRewardHook.sol#21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#27-37) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.11', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/access/Ownable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Context.sol#4)
        - 0.8.11 (contracts/chef/MasterChefRewardHook.sol#2)
        - 0.8.11 (contracts/interfaces/IChef.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#80-82) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#109-115) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60) is never used and should be removed
Context._msgData() (node_modules/@openzeppelin/contracts-0.8/utils/Context.sol#21-23) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#45-58) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#69-80) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#60-67) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#29-36) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/access/Ownable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Context.sol#4) allows old versions
Pragma version0.8.11 (contracts/chef/MasterChefRewardHook.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/interfaces/IChef.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.11 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter MasterChefRewardHook.setPid(uint256)._pid (contracts/chef/MasterChefRewardHook.sol#24) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (node_modules/@openzeppelin/contracts-0.8/access/Ownable.sol#54-56)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (node_modules/@openzeppelin/contracts-0.8/access/Ownable.sol#62-65)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

AUTOMATED TESTING

AuraClaimZap.sol

```
AuraClaimZap.constructor(address,address,address,address,address,address)._crv (contracts/peripheral/AuraClaimZap.sol#67) lacks a zero-check on :
        - crv = _crv (contracts/peripheral/AuraClaimZap.sol#74)
AuraClaimZap.constructor(address,address,address,address,address,address)._cvx (contracts/peripheral/AuraClaimZap.sol#68) lacks a zero-check on :
        - cvx = _cvx (contracts/peripheral/AuraClaimZap.sol#75)
AuraClaimZap.constructor(address,address,address,address,address,address)._cvxCrv (contracts/peripheral/AuraClaimZap.sol#69) lacks a zero-check on :
        - cvxCrv = _cvxCrv (contracts/peripheral/AuraClaimZap.sol#76)
AuraClaimZap.constructor(address,address,address,address,address,address)._crvDepositWrapper (contracts/peripheral/AuraClaimZap.sol#70) lacks a zero-check on :
        - crvDepositWrapper = _crvDepositWrapper (contracts/peripheral/AuraClaimZap.sol#77)
AuraClaimZap.constructor(address,address,address,address,address,address)._cvxCrvRewards (contracts/peripheral/AuraClaimZap.sol#71) lacks a zero-check on :
        - cvxCrvRewards = _cvxCrvRewards (contracts/peripheral/AuraClaimZap.sol#78)
AuraClaimZap.constructor(address,address,address,address,address,address)._locker (contracts/peripheral/AuraClaimZap.sol#72) lacks a zero-check on :
        - locker = _locker (contracts/peripheral/AuraClaimZap.sol#79)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

AuraClaimZap.claimRewards(address[],address[],address[],address[],AuraClaimZap.ClaimRewardsAmounts,uint256) (contracts/peripheral/AuraClaimZap.sol#122-160) has external calls inside a loop: IRewardSta
ral/AuraClaimZap.sol#138)
AuraClaimZap.claimRewards(address[],address[],address[],address[],AuraClaimZap.ClaimRewardsAmounts,uint256) (contracts/peripheral/AuraClaimZap.sol#122-160) has external calls inside a loop: IRewardSta
/peripheral/AuraClaimZap.sol#142)
AuraClaimZap.claimRewards(address[],address[],address[],address[],AuraClaimZap.ClaimRewardsAmounts,uint256) (contracts/peripheral/AuraClaimZap.sol#122-160) has external calls inside a loop: IRewardSta
Tokens(i_scope_1)) (contracts/peripheral/AuraClaimZap.sol#146)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#27-37) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.11', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4)
        - 0.8.11 (contracts/interfaces/IAuraLocker.sol#2)
        - 0.8.11 (contracts/interfaces/ICrvDepositorWrapper.sol#2)
        - 0.8.11 (contracts/interfaces/IRewardStaking.sol#2)
        - 0.8.11 (contracts/peripheral/AuraClaimZap.sol#2)
        - 0.8.11 (contracts/utils/AuraMath.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#80-82) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#109-115) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60) is never used and should be removed
AuraMath.add(uint256,uint256) (contracts/utils/AuraMath.sol#14-16) is never used and should be removed
AuraMath.average(uint256,uint256) (contracts/utils/AuraMath.sol#34-37) is never used and should be removed
AuraMath.div(uint256,uint256) (contracts/utils/AuraMath.sol#26-28) is never used and should be removed
AuraMath.mul(uint256,uint256) (contracts/utils/AuraMath.sol#22-24) is never used and should be removed
AuraMath.to112(uint256) (contracts/utils/AuraMath.sol#49-52) is never used and should be removed
AuraMath.to128(uint256) (contracts/utils/AuraMath.sol#44-47) is never used and should be removed
AuraMath.to224(uint256) (contracts/utils/AuraMath.sol#39-42) is never used and should be removed
AuraMath.to32(uint256) (contracts/utils/AuraMath.sol#59-62) is never used and should be removed
AuraMath.to96(uint256) (contracts/utils/AuraMath.sol#54-57) is never used and should be removed
AuraMath112.add(uint112,uint112) (contracts/utils/AuraMath.sol#74-76) is never used and should be removed
AuraMath112.sub(uint112,uint112) (contracts/utils/AuraMath.sol#78-80) is never used and should be removed
AuraMath224.add(uint224,uint224) (contracts/utils/AuraMath.sol#85-87) is never used and should be removed
AuraMath32.sub(uint32,uint32) (contracts/utils/AuraMath.sol#67-69) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#69-80) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#60-67) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#21-27) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4) allows old versions
Pragma version0.8.11 (contracts/interfaces/IAuraLocker.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/interfaces/ICrvDepositorWrapper.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/interfaces/IRewardStaking.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/peripheral/AuraClaimZap.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/utils/AuraMath.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.11 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60):
        - (success) = recipient.call(value: amount)() (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#123-134):
        - (success,returndata) = target.call(value: value)(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Variable AuraClaimZap._claimExtras(uint256,uint256,uint256,uint256,uint256,uint256,uint256).depositCrvMaxAmount (contracts/peripheral/AuraClaimZap.sol#177) is too similar to AuraClaimZap._clai
6).depositCvxMaxAmount (contracts/peripheral/AuraClaimZap.sol#179)
Variable AuraClaimZap._claimExtras(uint256,uint256,uint256,uint256,uint256,uint256,uint256).removeCrvBalance (contracts/peripheral/AuraClaimZap.sol#181) is too similar to AuraClaimZap._claimE
removeCvxBalance (contracts/peripheral/AuraClaimZap.sol#182)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
```

GaugeMigrator.sol

```
GaugeMigrator.migrate(uint256,uint256) (contracts/peripheral/GaugeMigrator.sol#26-40) ignores return value by IRewardPool4626(fromPool.crvRewards).withdraw(balance,address(this),msg.sender) (contracts
GaugeMigrator.migrate(uint256,uint256) (contracts/peripheral/GaugeMigrator.sol#26-40) ignores return value by IRewardPool4626(toPool.crvRewards).deposit(balance,msg.sender) (contracts/peripheral/Gauge
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

IBooster.poolInfo(uint256).poolInfo (contracts/interfaces/IBooster.sol#26) shadows:
        - IBooster.poolInfo(uint256) (contracts/interfaces/IBooster.sol#26) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#27-37) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.11', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4)
        - 0.8.11 (contracts/interfaces/IBooster.sol#2)
        - 0.8.11 (contracts/interfaces/IRewardPool4626.sol#2)
        - 0.8.11 (contracts/peripheral/GaugeMigrator.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#80-82) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#109-115) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#69-80) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#60-67) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#21-27) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#29-36) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4) allows old versions
Pragma version0.8.11 (contracts/interfaces/IBooster.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/interfaces/IRewardPool4626.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/peripheral/GaugeMigrator.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.11 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter GaugeMigrator.migrate(uint256,uint256)._fromPid (contracts/peripheral/GaugeMigrator.sol#26) is not in mixedCase
Parameter GaugeMigrator.migrate(uint256,uint256)._toPid (contracts/peripheral/GaugeMigrator.sol#26) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

PoolMigrator.sol

```
PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) ignores return value by IRewardPool4626(fromPool.crvRewards).withdraw(amount,address(this),msg.sender) (co
PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) ignores return value by IRewardPool4626(toPool.crvRewards).deposit(amount,msg.sender) (contracts/periphera
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

IBooster.poolInfo(uint256).poolInfo (contracts/interfaces/IBooster.sol#26) shadows:
        - IBooster.poolInfo(uint256) (contracts/interfaces/IBooster.sol#26) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) has external calls inside a loop: fromPool = boosterV1.poolInfo(_fromPid) (contracts/peripheral/PoolMigrat
PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) has external calls inside a loop: toPool = boosterV2.poolInfo(_toPid) (contracts/peripheral/PoolMigrator.s
PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) has external calls inside a loop: balance = IRewardPool4626(fromPool.crvRewards).balanceOf(msg.sender) (co
PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) has external calls inside a loop: IRewardPool4626(fromPool.crvRewards).withdraw(amount,address(this),msg.s
PoolMigrator._migrate(uint256,uint256,uint256) (contracts/peripheral/PoolMigrator.sol#56-76) has external calls inside a loop: IRewardPool4626(toPool.crvRewards).deposit(amount,msg.sender) (contracts
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#123-134) has external calls inside a loop: (success,returndata) = target.call{v
ess.sol#132)
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#60-67) has external calls inside a loop: newAllowance = token.allowan
-0.8/token/ERC20/utils/SafeERC20.sol#65)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#27-37) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#196-216) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#208-211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.11', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4)
        - 0.8.11 (contracts/interfaces/IBooster.sol#2)
        - 0.8.11 (contracts/interfaces/IRewardPool4626.sol#2)
        - 0.8.11 (contracts/peripheral/PoolMigrator.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#80-82) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#109-115) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#45-58) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#69-80) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#21-27) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#29-36) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#4) allows old versions
Pragma version0.8.11 (contracts/interfaces/IBooster.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/interfaces/IRewardPool4626.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.11 (contracts/peripheral/PoolMigrator.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.11 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.8/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter PoolMigrator.migrate(uint256[],uint256[],uint256[])._fromPids (contracts/peripheral/PoolMigrator.sol#37) is not in mixedCase
Parameter PoolMigrator.migrate(uint256[],uint256[],uint256[])._toPids (contracts/peripheral/PoolMigrator.sol#38) is not in mixedCase
Parameter PoolMigrator.migrate(uint256[],uint256[],uint256[])._amounts (contracts/peripheral/PoolMigrator.sol#39) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

AUTOMATED TESTING

BaseRewardPool.sol

```
BaseRewardPool.notifyRewardAmount(uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#365-382) performs a multiplication on the result of a division:
        -rewardRate = reward.div(duration) (convex-platform/contracts/contracts/BaseRewardPool.sol#371)
        -leftover = remaining.mul(rewardRate) (convex-platform/contracts/contracts/BaseRewardPool.sol#374)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

BaseRewardPool.rewardPerToken() (convex-platform/contracts/contracts/BaseRewardPool.sol#156-168) uses a dangerous strict equality:
        - totalSupply() == 0 (convex-platform/contracts/contracts/BaseRewardPool.sol#157)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

BaseRewardPool.queueNewRewards(uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#338-363) should emit an event for:
        - queuedRewards = 0 (convex-platform/contracts/contracts/BaseRewardPool.sol#345)
        - queuedRewards = 0 (convex-platform/contracts/contracts/BaseRewardPool.sol#358)
        - queuedRewards = _rewards (convex-platform/contracts/contracts/BaseRewardPool.sol#360)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

BaseRewardPool.constructor(uint256,address,address,address,address).operator_ (convex-platform/contracts/contracts/BaseRewardPool.sol#104) lacks a zero-check on :
                - operator = operator_ (convex-platform/contracts/contracts/BaseRewardPool.sol#110)
BaseRewardPool.constructor(uint256,address,address,address,address).rewardManager_ (convex-platform/contracts/contracts/BaseRewardPool.sol#105) lacks a zero-check on :
                - rewardManager = rewardManager_ (convex-platform/contracts/contracts/BaseRewardPool.sol#111)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

BaseRewardPool.withdraw(uint256,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#229-254) has external calls inside a loop: IRewards(extraRewards[i]).withdraw(msg.sender,amount) (convex-
BaseRewardPool.getReward(address,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#296-312) has external calls inside a loop: IRewards(extraRewards[i]).getReward(_account) (convex-platfor
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Reentrancy in BaseRewardPool._withdrawAndUnwrapTo(uint256,address,address) (convex-platform/contracts/contracts/BaseRewardPool.sol#269-285):
        External calls:
        - IDeposit(operator).withdrawTo(pid,amount,receiver) (convex-platform/contracts/contracts/BaseRewardPool.sol#279)
        Event emitted after the call(s):
        - Transfer(from,address(0),amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#282)
        - Withdrawn(from,amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#280)
Reentrancy in BaseRewardPool.getReward(address,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#296-312):
        External calls:
        - rewardToken.safeTransfer(_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#300)
        - IDeposit(operator).rewardClaimed(pid,_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#301)
        Event emitted after the call(s):
        - RewardPaid(_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#302)
Reentrancy in BaseRewardPool.stake(uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#178-188):
        External calls:
        - _processStake(_amount,msg.sender) (convex-platform/contracts/contracts/BaseRewardPool.sol#182)
                - IRewards(extraRewards[i]).stake(_receiver,_amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#220)
        - stakingToken.safeTransferFrom(msg.sender,address(this),_amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#184)
        Event emitted after the call(s):
        - Staked(msg.sender,_amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#185)
Reentrancy in BaseRewardPool.stakeFor(address,uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#196-207):
        External calls:
        - _processStake(_amount,_for) (convex-platform/contracts/contracts/BaseRewardPool.sol#200)
                - IRewards(extraRewards[i]).stake(_receiver,_amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#220)
        - stakingToken.safeTransferFrom(msg.sender,address(this),_amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#203)
        Event emitted after the call(s):
        - Staked(_for,_amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#204)
Reentrancy in BaseRewardPool.withdraw(uint256,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#229-254):
        External calls:
        - stakingToken.safeTransfer(msg.sender,amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#244)
        Event emitted after the call(s):
        - Withdrawn(msg.sender,amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#245)
Reentrancy in BaseRewardPool.withdraw(uint256,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#229-254):
        External calls:
        - stakingToken.safeTransfer(msg.sender,amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#244)
        - getReward(msg.sender,true) (convex-platform/contracts/contracts/BaseRewardPool.sol#248)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#69)
                - rewardToken.safeTransfer(_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#300)
                - IDeposit(operator).rewardClaimed(pid,_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#301)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
                - IRewards(extraRewards[i]).getReward(_account) (convex-platform/contracts/contracts/BaseRewardPool.sol#308)
        External calls sending eth:
        - getReward(msg.sender,true) (convex-platform/contracts/contracts/BaseRewardPool.sol#248)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
        Event emitted after the call(s):
        - RewardPaid(_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#302)
                - getReward(msg.sender,true) (convex-platform/contracts/contracts/BaseRewardPool.sol#248)
        - Transfer(msg.sender,address(0),amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#251)
Reentrancy in BaseRewardPool.withdrawAndUnwrap(uint256,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#260-267):
        External calls:
        - _withdrawAndUnwrapTo(amount,msg.sender,msg.sender) (convex-platform/contracts/contracts/BaseRewardPool.sol#261)
                - IRewards(extraRewards[i]).withdraw(from,amount) (convex-platform/contracts/contracts/BaseRewardPool.sol#272)
```

```
                - IDeposit(operator).withdrawTo(pid,amount,receiver) (convex-platform/contracts/contracts/BaseRewardPool.sol#279)
        - getReward(msg.sender,true) (convex-platform/contracts/contracts/BaseRewardPool.sol#264)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#69)
                - rewardToken.safeTransfer(_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#300)
                - IDeposit(operator).rewardClaimed(pid,_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#301)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
                - IRewards(extraRewards[i]).getReward(_account) (convex-platform/contracts/contracts/BaseRewardPool.sol#308)
        External calls sending eth:
        - getReward(msg.sender,true) (convex-platform/contracts/contracts/BaseRewardPool.sol#264)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
        Event emitted after the call(s):
        - RewardPaid(_account,reward) (convex-platform/contracts/contracts/BaseRewardPool.sol#302)
                - getReward(msg.sender,true) (convex-platform/contracts/contracts/BaseRewardPool.sol#264)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

BaseRewardPool.getReward(address,bool) (convex-platform/contracts/contracts/BaseRewardPool.sol#296-312) uses timestamp for comparisons
        Dangerous comparisons:
        - reward > 0 (convex-platform/contracts/contracts/BaseRewardPool.sol#298)
BaseRewardPool.processIdleRewards() (convex-platform/contracts/contracts/BaseRewardPool.sol#326-331) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp >= periodFinish && queuedRewards > 0 (convex-platform/contracts/contracts/BaseRewardPool.sol#327)
BaseRewardPool.queueNewRewards(uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#338-363) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp >= periodFinish (convex-platform/contracts/contracts/BaseRewardPool.sol#343)
        - queuedRatio < newRewardRatio (convex-platform/contracts/contracts/BaseRewardPool.sol#356)
BaseRewardPool.notifyRewardAmount(uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#365-382) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp >= periodFinish (convex-platform/contracts/contracts/BaseRewardPool.sol#370)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#33)
Address._verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#171-188) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#180-183)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.6.12', '>=0.6.0<0.8.0', '>=0.6.2<0.8.0']
        - 0.6.12 (convex-platform/contracts/contracts/BaseRewardPool.sol#2)
        - 0.6.12 (convex-platform/contracts/contracts/Interfaces.sol#2)
        - 0.6.12 (convex-platform/contracts/contracts/interfaces/MathUtil.sol#2)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/IERC20.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#3)
        - >=0.6.2<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#104-106) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#153-155) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#163-169) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#129-131) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#139-145) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#53-59) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#37-46) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#53-56) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#48-51) is never used and should be removed
SafeMath.div(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#190-193) is never used and should be removed
SafeMath.mod(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#152-155) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#210-213) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#170-173) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#24-28) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#60-63) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#70-73) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#45-53) is never used and should be removed
SafeMath.trySub(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#35-38) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/IERC20.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#3) is too complex
Pragma version>=0.6.2<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#3) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#114-121):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#139-145):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#143)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#163-169):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#167)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter BaseRewardPool.addExtraReward(address)._reward (convex-platform/contracts/contracts/BaseRewardPool.sol#126) is not in mixedCase
Parameter BaseRewardPool.stake(uint256)._amount (convex-platform/contracts/contracts/BaseRewardPool.sol#178) is not in mixedCase
Parameter BaseRewardPool.stakeFor(address,uint256)._for (convex-platform/contracts/contracts/BaseRewardPool.sol#196) is not in mixedCase
Parameter BaseRewardPool.stakeFor(address,uint256)._amount (convex-platform/contracts/contracts/BaseRewardPool.sol#196) is not in mixedCase
Parameter BaseRewardPool.getReward(address,bool)._account (convex-platform/contracts/contracts/BaseRewardPool.sol#296) is not in mixedCase
Parameter BaseRewardPool.getReward(address,bool)._claimExtras (convex-platform/contracts/contracts/BaseRewardPool.sol#296) is not in mixedCase
Parameter BaseRewardPool.queueNewRewards(uint256)._rewards (convex-platform/contracts/contracts/BaseRewardPool.sol#338) is not in mixedCase
Constant BaseRewardPool.duration (convex-platform/contracts/contracts/BaseRewardPool.sol#65) is not in UPPER_CASE_WITH_UNDERSCORES
Constant BaseRewardPool.newRewardRatio (convex-platform/contracts/contracts/BaseRewardPool.sol#78) is not in UPPER_CASE_WITH_UNDERSCORES
Function ICurveGauge.claim_rewards() (convex-platform/contracts/contracts/Interfaces.sol#10) is not in mixedCase
Function ICurveGauge.reward_tokens(uint256) (convex-platform/contracts/contracts/Interfaces.sol#11) is not in mixedCase
Function ICurveGauge.rewarded_token() (convex-platform/contracts/contracts/Interfaces.sol#12) is not in mixedCase
Function ICurveGauge.lp_token() (convex-platform/contracts/contracts/Interfaces.sol#13) is not in mixedCase
Function ICurveVoteEscrow.create_lock(uint256,uint256) (convex-platform/contracts/contracts/Interfaces.sol#17) is not in mixedCase
Function ICurveVoteEscrow.increase_amount(uint256) (convex-platform/contracts/contracts/Interfaces.sol#18) is not in mixedCase
Function ICurveVoteEscrow.increase_unlock_time(uint256) (convex-platform/contracts/contracts/Interfaces.sol#19) is not in mixedCase
Function ICurveVoteEscrow.smart_wallet_checker() (convex-platform/contracts/contracts/Interfaces.sol#21) is not in mixedCase
Function ICurveVoteEscrow.commit_smart_wallet_checker(address) (convex-platform/contracts/contracts/Interfaces.sol#22) is not in mixedCase
Function ICurveVoteEscrow.apply_smart_wallet_checker() (convex-platform/contracts/contracts/Interfaces.sol#23) is not in mixedCase
Function IVoting.vote_for_gauge_weights(address,uint256) (convex-platform/contracts/contracts/Interfaces.sol#35) is not in mixedCase
Function IRewardFactory.CreateCrvRewards(uint256,address,address) (convex-platform/contracts/contracts/Interfaces.sol#118) is not in mixedCase
Function IRewardFactory.CreateTokenRewards(address,address,address) (convex-platform/contracts/contracts/Interfaces.sol#119) is not in mixedCase
Function IStashFactory.CreateStash(uint256,address,address,uint256) (convex-platform/contracts/contracts/Interfaces.sol#126) is not in mixedCase
Function ITokenFactory.CreateDepositToken(address) (convex-platform/contracts/contracts/Interfaces.sol#130) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

stakeFor(address,uint256) should be declared external:
        - BaseRewardPool.stakeFor(address,uint256) (convex-platform/contracts/contracts/BaseRewardPool.sol#196-207)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

ExtraRewardStashV3.sol

```
Reentrancy in ExtraRewardStashV3.claimRewards() (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#96-120):
        External calls:
        - checkForNewRewardTokens() (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#100)
                - rewardContract = IRewardFactory(rewardFactory).CreateTokenRewards(_token,mainRewardContract,address(this)) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#169-172)
        - IDeposit(operator).setGaugeRedirect(pid) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#104)
        State variables written after the call(s):
        - hasRedirected = true (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#105)
Reentrancy in ExtraRewardStashV3.setToken(address) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#159-179):
        External calls:
        - rewardContract = IRewardFactory(rewardFactory).CreateTokenRewards(_token,mainRewardContract,address(this)) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#169-172)
        State variables written after the call(s):
        - t.rewardAddress = rewardContract (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#174)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

ExtraRewardStashV3.claimRewards() (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#96-120) ignores return value by IDeposit(operator).setGaugeRedirect(pid) (convex-platform/contracts/contr
ExtraRewardStashV3.claimRewards() (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#96-120) ignores return value by IDeposit(operator).claimRewards(pid,gauge) (convex-platform/contracts/con
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

ExtraRewardStashV3.constructor(address)._crv (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#58) lacks a zero-check on :
                - crv = _crv (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#59)
ExtraRewardStashV3.initialize(uint256,address,address,address,address)._operator (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) lacks a zero-check on :
                - operator = _operator (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#72)
ExtraRewardStashV3.initialize(uint256,address,address,address,address)._staker (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) lacks a zero-check on :
                - staker = _staker (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#73)
ExtraRewardStashV3.initialize(uint256,address,address,address,address)._gauge (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) lacks a zero-check on :
                - gauge = _gauge (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#74)
ExtraRewardStashV3.initialize(uint256,address,address,address,address)._rFactory (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) lacks a zero-check on :
                - rewardFactory = _rFactory (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#75)
ExtraRewardStashV3.setRewardHook(address)._hook (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#147) lacks a zero-check on :
                - rewardHook = _hook (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#150)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

ExtraRewardStashV3.processStash() (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#195-215) has external calls inside a loop: amount = IERC20(token).balanceOf(address(this)) (convex-platfo
ExtraRewardStashV3.processStash() (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#195-215) has external calls inside a loop: IRewards(rewards).queueNewRewards(amount) (convex-platform/con
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Reentrancy in ExtraRewardStashV3.setRewardHook(address) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#147-151):
        External calls:
        - require(bool,string)(IDeposit(operator).owner() == msg.sender,!owner) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#149)
        State variables written after the call(s):
        - rewardHook = _hook (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#150)
Reentrancy in ExtraRewardStashV3.setToken(address) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#159-179):
        External calls:
        - rewardContract = IRewardFactory(rewardFactory).CreateTokenRewards(_token,mainRewardContract,address(this)) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#169-172)
        State variables written after the call(s):
        - tokenList.push(_token) (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#177)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#33)
Address._verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#171-188) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#180-183)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.6.12', '>=0.6.0<0.8.0', '>=0.6.2<0.8.0']
        - 0.6.12 (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#2)
        - 0.6.12 (convex-platform/contracts/contracts/Interfaces.sol#2)
        - 0.6.12 (convex-platform/contracts/contracts/interfaces/IRewardHook.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/IERC20.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#3)
        - >=0.6.2<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#104-106) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#153-155) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#163-169) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#129-131) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#139-146) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#53-59) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#37-46) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#53-56) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#48-51) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#26-28) is never used and should be removed
SafeMath.div(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#135-138) is never used and should be removed
SafeMath.div(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#190-193) is never used and should be removed
SafeMath.mod(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#152-155) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#210-213) is never used and should be removed
SafeMath.mul(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#116-121) is never used and should be removed
SafeMath.sub(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#101-104) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#170-173) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#24-28) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#60-63) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#70-73) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#45-53) is never used and should be removed
SafeMath.trySub(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#35-38) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/IERC20.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#3) is too complex
Pragma version>=0.6.2<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#3) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#114-121):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#139-145):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#143)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#163-169):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#167)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

ExtraRewardStashV3 (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#25-217) should inherit from IStash (convex-platform/contracts/contracts/Interfaces.sol#79-84)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-inheritance

Parameter ExtraRewardStashV3.initialize(uint256,address,address,address,address)._pid (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) is not in mixedCase
Parameter ExtraRewardStashV3.initialize(uint256,address,address,address,address)._operator (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) is not in mixedCase
Parameter ExtraRewardStashV3.initialize(uint256,address,address,address,address)._staker (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) is not in mixedCase
Parameter ExtraRewardStashV3.initialize(uint256,address,address,address,address)._gauge (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) is not in mixedCase
Parameter ExtraRewardStashV3.initialize(uint256,address,address,address,address)._rFactory (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#69) is not in mixedCase
Parameter ExtraRewardStashV3.setExtraReward(address)._token (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#139) is not in mixedCase
Parameter ExtraRewardStashV3.setRewardHook(address)._hook (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#147) is not in mixedCase
Parameter ExtraRewardStashV3.setToken(address)._token (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#159) is not in mixedCase
Constant ExtraRewardStashV3.maxRewards (convex-platform/contracts/contracts/ExtraRewardStashV3.sol#31) is not in UPPER_CASE_WITH_UNDERSCORES
Function ICurveGauge.claim_rewards() (convex-platform/contracts/contracts/Interfaces.sol#10) is not in mixedCase
Function ICurveGauge.reward_tokens(uint256) (convex-platform/contracts/contracts/Interfaces.sol#11) is not in mixedCase
Function ICurveGauge.rewarded_token() (convex-platform/contracts/contracts/Interfaces.sol#12) is not in mixedCase
Function ICurveGauge.lp_token() (convex-platform/contracts/contracts/Interfaces.sol#13) is not in mixedCase
Function ICurveVoteEscrow.create_lock(uint256,uint256) (convex-platform/contracts/contracts/Interfaces.sol#17) is not in mixedCase
Function ICurveVoteEscrow.increase_amount(uint256) (convex-platform/contracts/contracts/Interfaces.sol#18) is not in mixedCase
Function ICurveVoteEscrow.increase_unlock_time(uint256) (convex-platform/contracts/contracts/Interfaces.sol#19) is not in mixedCase
Function ICurveVoteEscrow.smart_wallet_checker() (convex-platform/contracts/contracts/Interfaces.sol#21) is not in mixedCase
Function ICurveVoteEscrow.commit_smart_wallet_checker(address) (convex-platform/contracts/contracts/Interfaces.sol#22) is not in mixedCase
Function ICurveVoteEscrow.apply_smart_wallet_checker() (convex-platform/contracts/contracts/Interfaces.sol#23) is not in mixedCase
Function IVoting.vote_for_gauge_weights(address,uint256) (convex-platform/contracts/contracts/Interfaces.sol#35) is not in mixedCase
Function IRewardFactory.CreateCrvRewards(uint256,address,address) (convex-platform/contracts/contracts/Interfaces.sol#118) is not in mixedCase
Function IRewardFactory.CreateTokenRewards(address,address,address) (convex-platform/contracts/contracts/Interfaces.sol#119) is not in mixedCase
Function IStashFactory.CreateStash(uint256,address,address,uint256) (convex-platform/contracts/contracts/Interfaces.sol#126) is not in mixedCase
Function ITokenFactory.CreateDepositToken(address) (convex-platform/contracts/contracts/Interfaces.sol#130) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

AUTOMATED TESTING

Booster.sol

```
Booster._earmarkRewards(uint256) (convex-platform/contracts/contracts/Booster.sol#611-677) ignores return value by IERC20(crv).transfer(treasury,crvBalBefore) (convex-platform/contracts/contracts/Boos
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer

Reentrancy in Booster.addPool(address,address,uint256) (convex-platform/contracts/contracts/Booster.sol#339-380):
        External calls:
        - token = ITokenFactory(tokenFactory).CreateDepositToken(_lptoken) (convex-platform/contracts/contracts/Booster.sol#348)
        - newRewardPool = IRewardFactory(rewardFactory).CreateCrvRewards(pid,token,_lptoken) (convex-platform/contracts/contracts/Booster.sol#350)
        - stash = IStashFactory(stashFactory).CreateStash(pid,_gauge,staker,_stashVersion) (convex-platform/contracts/contracts/Booster.sol#352)
        State variables written after the call(s):
        - poolInfo.push(PoolInfo(_lptoken,token,_gauge,newRewardPool,stash,false)) (convex-platform/contracts/contracts/Booster.sol#355-364)
        - poolInfo[pid].stash = stash (convex-platform/contracts/contracts/Booster.sol#370)
Reentrancy in Booster.setFeeInfo(address,address) (convex-platform/contracts/contracts/Booster.sol#227-261):
        External calls:
        - rewards = IRewardFactory(rewardFactory).CreateTokenRewards(_feeToken,lockRewards,address(this)) (convex-platform/contracts/contracts/Booster.sol#249)
        State variables written after the call(s):
        - feeTokens[_feeToken] = FeeDistro(_feeDistro,rewards,true) (convex-platform/contracts/contracts/Booster.sol#250-254)
Reentrancy in Booster.shutdownPool(uint256) (convex-platform/contracts/contracts/Booster.sol#386-399):
        External calls:
        - IStaker(staker).withdrawAll(pool.lptoken,pool.gauge) (convex-platform/contracts/contracts/Booster.sol#391-392)
        State variables written after the call(s):
        - pool.shutdown = true (convex-platform/contracts/contracts/Booster.sol#394)
Reentrancy in Booster.shutdownSystem() (convex-platform/contracts/contracts/Booster.sol#405-421):
        External calls:
        - IStaker(staker).withdrawAll(token,gauge) (convex-platform/contracts/contracts/Booster.sol#417-419)
        State variables written after the call(s):
        - pool.shutdown = true (convex-platform/contracts/contracts/Booster.sol#418)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

Booster.addPool(address,address,uint256) (convex-platform/contracts/contracts/Booster.sol#339-380) ignores return value by IStaker(staker).setStashAccess(stash,true) (convex-platform/contracts/contrac
Booster.shutdownPool(uint256) (convex-platform/contracts/contracts/Booster.sol#386-399) ignores return value by IStaker(staker).withdrawAll(pool.lptoken,pool.gauge) (convex-platform/contracts/contract
Booster.shutdownSystem() (convex-platform/contracts/contracts/Booster.sol#405-421) ignores return value by IStaker(staker).withdrawAll(token,gauge) (convex-platform/contracts/contracts/Booster.sol#41
Booster.deposit(uint256,uint256,bool) (convex-platform/contracts/contracts/Booster.sol#427-463) ignores return value by IStaker(staker).deposit(lptoken,gauge) (convex-platform/contracts/contracts/Boos
Booster.deposit(uint256,uint256,bool) (convex-platform/contracts/contracts/Booster.sol#427-463) ignores return value by IStash(stash).stashRewards() (convex-platform/contracts/contracts/Booster.sol#4
Booster._withdraw(uint256,uint256,address,address) (convex-platform/contracts/contracts/Booster.sol#483-509) ignores return value by IStaker(staker).withdraw(lptoken,gauge,_amount) (convex-platform/co
Booster._withdraw(uint256,uint256,address,address) (convex-platform/contracts/contracts/Booster.sol#483-509) ignores return value by IStash(stash).stashRewards() (convex-platform/contracts/contracts/B
Booster.setDelegate(address,address,bytes32) (convex-platform/contracts/contracts/Booster.sol#554-558) ignores return value by IStaker(staker).execute(_delegateContract,uint256(0),data) (convex-platfo
Booster.vote(uint256,address,bool) (convex-platform/contracts/contracts/Booster.sol#563-569) ignores return value by IStaker(staker).vote(_voteId,_votingAddress,_support) (convex-platform/contracts/co
Booster.voteGaugeWeight(address[],uint256[]) (convex-platform/contracts/contracts/Booster.sol#574-581) ignores return value by IStaker(staker).voteGaugeWeight(_gauge[i],_weight[i]) (convex-platform/co
Booster.claimRewards(uint256,address) (convex-platform/contracts/contracts/Booster.sol#586-592) ignores return value by IStaker(staker).claimRewards(_gauge) (convex-platform/contracts/contracts/Booste
Booster.setGaugeRedirect(uint256) (convex-platform/contracts/contracts/Booster.sol#597-604) ignores return value by IStaker(staker).execute(gauge,uint256(0),data) (convex-platform/contracts/contracts/
Booster._earmarkRewards(uint256) (convex-platform/contracts/contracts/Booster.sol#611-677) ignores return value by IStaker(staker).claimCrv(gauge) (convex-platform/contracts/contracts/Booster.sol#624)
Booster._earmarkRewards(uint256) (convex-platform/contracts/contracts/Booster.sol#611-677) ignores return value by IStash(stash).claimRewards() (convex-platform/contracts/contracts/Booster.sol#638)
Booster._earmarkRewards(uint256) (convex-platform/contracts/contracts/Booster.sol#611-677) ignores return value by IStash(stash).processStash() (convex-platform/contracts/contracts/Booster.sol#640)
Booster.earmarkFees(address) (convex-platform/contracts/contracts/Booster.sol#694-714) ignores return value by IStaker(staker).claimFees(feeDistro.distro,_feeToken) (convex-platform/contracts/contract
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Booster.constructor(address,address,address,address,address)._staker (convex-platform/contracts/contracts/Booster.sol#106) lacks a zero-check on :
                - staker = _staker (convex-platform/contracts/contracts/Booster.sol#112)
Booster.constructor(address,address,address,address,address)._minter (convex-platform/contracts/contracts/Booster.sol#107) lacks a zero-check on :
                - minter = _minter (convex-platform/contracts/contracts/Booster.sol#113)
Booster.constructor(address,address,address,address,address)._crv (convex-platform/contracts/contracts/Booster.sol#108) lacks a zero-check on :
                - crv = _crv (convex-platform/contracts/contracts/Booster.sol#114)
Booster.constructor(address,address,address,address,address)._voteOwnership (convex-platform/contracts/contracts/Booster.sol#109) lacks a zero-check on :
                - voteOwnership = _voteOwnership (convex-platform/contracts/contracts/Booster.sol#115)
Booster.constructor(address,address,address,address,address)._voteParameter (convex-platform/contracts/contracts/Booster.sol#110) lacks a zero-check on :
                - voteParameter = _voteParameter (convex-platform/contracts/contracts/Booster.sol#116)
Booster.setOwner(address)._owner (convex-platform/contracts/contracts/Booster.sol#137) lacks a zero-check on :
                - owner = _owner (convex-platform/contracts/contracts/Booster.sol#139)
Booster.setFeeManager(address)._feeM (convex-platform/contracts/contracts/Booster.sol#147) lacks a zero-check on :
                - feeManager = _feeM (convex-platform/contracts/contracts/Booster.sol#149)
Booster.setPoolManager(address)._poolM (convex-platform/contracts/contracts/Booster.sol#157) lacks a zero-check on :
                - poolManager = _poolM (convex-platform/contracts/contracts/Booster.sol#159)
Booster.setFactories(address,address,address)._sfactory (convex-platform/contracts/contracts/Booster.sol#167) lacks a zero-check on :
                - stashFactory = _sfactory (convex-platform/contracts/contracts/Booster.sol#172)
Booster.setFactories(address,address,address)._rfactory (convex-platform/contracts/contracts/Booster.sol#167) lacks a zero-check on :
                - rewardFactory = _rfactory (convex-platform/contracts/contracts/Booster.sol#178)
Booster.setFactories(address,address,address)._tfactory (convex-platform/contracts/contracts/Booster.sol#167) lacks a zero-check on :
                - tokenFactory = _tfactory (convex-platform/contracts/contracts/Booster.sol#179)
Booster.setArbitrator(address)._arb (convex-platform/contracts/contracts/Booster.sol#190) lacks a zero-check on :
                - rewardArbitrator = _arb (convex-platform/contracts/contracts/Booster.sol#192)
Booster.setVoteDelegate(address)._voteDelegate (convex-platform/contracts/contracts/Booster.sol#200) lacks a zero-check on :
                - voteDelegate = _voteDelegate (convex-platform/contracts/contracts/Booster.sol#202)
Booster.setRewardContracts(address,address)._rewards (convex-platform/contracts/contracts/Booster.sol#210) lacks a zero-check on :
                - lockRewards = _rewards (convex-platform/contracts/contracts/Booster.sol#216)
Booster.setRewardContracts(address,address)._stakerRewards (convex-platform/contracts/contracts/Booster.sol#210) lacks a zero-check on :
                - stakerRewards = _stakerRewards (convex-platform/contracts/contracts/Booster.sol#217)
```

```
Booster.setTreasury(address)._treasury (convex-platform/contracts/contracts/Booster.sol#305) lacks a zero-check on :
        - treasury = _treasury (convex-platform/contracts/contracts/Booster.sol#307)
Booster.setBridgeDelegate(address)._bridgeDelegate (convex-platform/contracts/contracts/Booster.sol#317) lacks a zero-check on :
        - bridgeDelegate = _bridgeDelegate (convex-platform/contracts/contracts/Booster.sol#319)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Booster.shutdownSystem() (convex-platform/contracts/contracts/Booster.sol#405-421) has external calls inside a loop: IStaker(staker).withdrawAll(token,gauge) (convex-platform/contracts/contracts/Boost
Booster.voteGaugeWeight(address[],uint256[]) (convex-platform/contracts/contracts/Booster.sol#574-581) has external calls inside a loop: IStaker(staker).voteGaugeWeight(_gauge[i],_weight[i]) (convex-p
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#calls-inside-a-loop

Reentrancy in Booster.addPool(address,address,uint256) (convex-platform/contracts/contracts/Booster.sol#339-380):
        External calls:
        - token = ITokenFactory(tokenFactory).CreateDepositToken(_lptoken) (convex-platform/contracts/contracts/Booster.sol#348)
        - newRewardPool = IRewardFactory(rewardFactory).CreateCrvRewards(pid,token,_lptoken) (convex-platform/contracts/contracts/Booster.sol#350)
        - stash = IStashFactory(stashFactory).CreateStash(pid,_gauge,staker,_stashVersion) (convex-platform/contracts/contracts/Booster.sol#352)
        State variables written after the call(s):
        - gaugeMap[_gauge] = true (convex-platform/contracts/contracts/Booster.sol#365)
Reentrancy in Booster.addPool(address,address,uint256) (convex-platform/contracts/contracts/Booster.sol#339-380):
        External calls:
        - token = ITokenFactory(tokenFactory).CreateDepositToken(_lptoken) (convex-platform/contracts/contracts/Booster.sol#348)
        - newRewardPool = IRewardFactory(rewardFactory).CreateCrvRewards(pid,token,_lptoken) (convex-platform/contracts/contracts/Booster.sol#350)
        - stash = IStashFactory(stashFactory).CreateStash(pid,_gauge,staker,_stashVersion) (convex-platform/contracts/contracts/Booster.sol#352)
        - IStaker(staker).setStashAccess(stash,true) (convex-platform/contracts/contracts/Booster.sol#371)
        - IRewardFactory(rewardFactory).setAccess(stash,true) (convex-platform/contracts/contracts/Booster.sol#372)
        State variables written after the call(s):
        - getRewardMultipliers[newRewardPool] = REWARD_MULTIPLIER_DENOMINATOR (convex-platform/contracts/contracts/Booster.sol#376)
Reentrancy in Booster.shutdownPool(uint256) (convex-platform/contracts/contracts/Booster.sol#386-399):
        External calls:
        - IStaker(staker).withdrawAll(pool.lptoken,pool.gauge) (convex-platform/contracts/contracts/Booster.sol#391-392)
        State variables written after the call(s):
        - gaugeMap[pool.gauge] = false (convex-platform/contracts/contracts/Booster.sol#395)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in Booster._withdraw(uint256,uint256,address,address) (convex-platform/contracts/contracts/Booster.sol#483-509):
        External calls:
        - ITokenMinter(token).burn(_from,_amount) (convex-platform/contracts/contracts/Booster.sol#490)
        - IStaker(staker).withdraw(lptoken,gauge,_amount) (convex-platform/contracts/contracts/Booster.sol#495)
        - IStash(stash).stashRewards() (convex-platform/contracts/contracts/Booster.sol#502)
        - IERC20(lptoken).safeTransfer(_to,_amount) (convex-platform/contracts/contracts/Booster.sol#506)
        Event emitted after the call(s):
        - Withdrawn(_to,_pid,_amount) (convex-platform/contracts/contracts/Booster.sol#508)
Reentrancy in Booster.addPool(address,address,uint256) (convex-platform/contracts/contracts/Booster.sol#339-380):
        External calls:
        - token = ITokenFactory(tokenFactory).CreateDepositToken(_lptoken) (convex-platform/contracts/contracts/Booster.sol#348)
        - newRewardPool = IRewardFactory(rewardFactory).CreateCrvRewards(pid,token,_lptoken) (convex-platform/contracts/contracts/Booster.sol#350)
        - stash = IStashFactory(stashFactory).CreateStash(pid,_gauge,staker,_stashVersion) (convex-platform/contracts/contracts/Booster.sol#352)
        - IStaker(staker).setStashAccess(stash,true) (convex-platform/contracts/contracts/Booster.sol#371)
        - IRewardFactory(rewardFactory).setAccess(stash,true) (convex-platform/contracts/contracts/Booster.sol#372)
        Event emitted after the call(s):
        - PoolAdded(_lptoken,_gauge,token,newRewardPool,stash,pid) (convex-platform/contracts/contracts/Booster.sol#378)
Reentrancy in Booster.deposit(uint256,uint256,bool) (convex-platform/contracts/contracts/Booster.sol#427-463):
        External calls:
        - IERC20(lptoken).safeTransferFrom(msg.sender,staker,_amount) (convex-platform/contracts/contracts/Booster.sol#434)
        - IStaker(staker).deposit(lptoken,gauge) (convex-platform/contracts/contracts/Booster.sol#439)
        - IStash(stash).stashRewards() (convex-platform/contracts/contracts/Booster.sol#444)
        - ITokenMinter(token).mint(address(this),_amount) (convex-platform/contracts/contracts/Booster.sol#450)
        - IERC20(token).safeApprove(rewardContract,0) (convex-platform/contracts/contracts/Booster.sol#452)
        - IERC20(token).safeApprove(rewardContract,_amount) (convex-platform/contracts/contracts/Booster.sol#453)
        - IRewards(rewardContract).stakeFor(msg.sender,_amount) (convex-platform/contracts/contracts/Booster.sol#454)
        - ITokenMinter(token).mint(msg.sender,_amount) (convex-platform/contracts/contracts/Booster.sol#457)
        Event emitted after the call(s):
        - Deposited(msg.sender,_pid,_amount) (convex-platform/contracts/contracts/Booster.sol#461)
Reentrancy in Booster.setFeeInfo(address,address) (convex-platform/contracts/contracts/Booster.sol#227-261):
        External calls:
        - rewards = IRewardFactory(rewardFactory).CreateTokenRewards(_feeToken,lockRewards,address(this)) (convex-platform/contracts/contracts/Booster.sol#249)
        Event emitted after the call(s):
        - FeeInfoUpdated(_feeDistro,rewards,_feeToken) (convex-platform/contracts/contracts/Booster.sol#255)
Reentrancy in Booster.shutdownPool(uint256) (convex-platform/contracts/contracts/Booster.sol#386-399):
        External calls:
        - IStaker(staker).withdrawAll(pool.lptoken,pool.gauge) (convex-platform/contracts/contracts/Booster.sol#391-392)
        Event emitted after the call(s):
        - PoolShutdown(_pid) (convex-platform/contracts/contracts/Booster.sol#397)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Booster.distributeL2Fees(uint256) (convex-platform/contracts/contracts/Booster.sol#739-766) uses timestamp for comparisons

        Dangerous comparisons:
        - require(bool,string)(l2FeesHistory[epoch] <= 70000e18,Too many L2 Fees) (convex-platform/contracts/contracts/Booster.sol#751)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Address.isContract(address) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#33)
Address._verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#171-188) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#180-183)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Booster.deposit(uint256,uint256,bool) (convex-platform/contracts/contracts/Booster.sol#427-463) compares to a boolean constant:
        -require(bool,string)(pool.shutdown == false,pool is closed) (convex-platform/contracts/contracts/Booster.sol#430)
Booster._earmarkRewards(uint256) (convex-platform/contracts/contracts/Booster.sol#611-677) compares to a boolean constant:
        -require(bool,string)(pool.shutdown == false,pool is closed) (convex-platform/contracts/contracts/Booster.sol#613)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Different versions of Solidity are used:
        - Version used: ['0.6.12', '>=0.6.0<0.8.0', '>=0.6.2<0.8.0']
        - 0.6.12 (convex-platform/contracts/contracts/Booster.sol#2)
        - 0.6.12 (convex-platform/contracts/contracts/Interfaces.sol#2)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/IERC20.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#3)
        - >=0.6.2<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#3)
        - >=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/ReentrancyGuard.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#104-106) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#153-155) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#163-169) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#129-131) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#139-145) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#53-59) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#53-56) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#48-51) is never used and should be removed
SafeMath.div(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#190-193) is never used and should be removed
SafeMath.mod(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#152-155) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#210-213) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#170-173) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#24-28) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#60-63) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#70-73) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#45-53) is never used and should be removed
SafeMath.trySub(uint256,uint256) (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#35-38) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/math/SafeMath.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/IERC20.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/token/ERC20/SafeERC20.sol#3) is too complex
Pragma version>=0.6.2<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#3) is too complex
Pragma version>=0.6.0<0.8.0 (node_modules/@openzeppelin/contracts-0.6/utils/ReentrancyGuard.sol#3) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#114-121):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#119)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#139-145):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#143)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#163-169):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts-0.6/utils/Address.sol#167)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter Booster.setOwner(address)._owner (convex-platform/contracts/contracts/Booster.sol#137) is not in mixedCase
Parameter Booster.setFeeManager(address)._feeM (convex-platform/contracts/contracts/Booster.sol#147) is not in mixedCase
Parameter Booster.setPoolManager(address)._poolM (convex-platform/contracts/contracts/Booster.sol#157) is not in mixedCase
Parameter Booster.setFactories(address,address,address)._rfactory (convex-platform/contracts/contracts/Booster.sol#167) is not in mixedCase
Parameter Booster.setFactories(address,address,address)._sfactory (convex-platform/contracts/contracts/Booster.sol#167) is not in mixedCase
Parameter Booster.setFactories(address,address,address)._tfactory (convex-platform/contracts/contracts/Booster.sol#167) is not in mixedCase
Parameter Booster.setArbitrator(address)._arb (convex-platform/contracts/contracts/Booster.sol#190) is not in mixedCase
Parameter Booster.setVoteDelegate(address)._voteDelegate (convex-platform/contracts/contracts/Booster.sol#200) is not in mixedCase
Parameter Booster.setRewardContracts(address,address)._rewards (convex-platform/contracts/contracts/Booster.sol#210) is not in mixedCase
Parameter Booster.setRewardContracts(address,address)._stakerRewards (convex-platform/contracts/contracts/Booster.sol#210) is not in mixedCase
Parameter Booster.setFeeInfo(address,address)._feeToken (convex-platform/contracts/contracts/Booster.sol#227) is not in mixedCase
Parameter Booster.setFeeInfo(address,address)._feeDistro (convex-platform/contracts/contracts/Booster.sol#227) is not in mixedCase
```

```
Parameter Booster.updateFeeInfo(address,bool)._feeToken (convex-platform/contracts/contracts/Booster.sol#266) is not in mixedCase
Parameter Booster.updateFeeInfo(address,bool)._active (convex-platform/contracts/contracts/Booster.sol#266) is not in mixedCase
Parameter Booster.setFees(uint256,uint256,uint256,uint256)._lockFees (convex-platform/contracts/contracts/Booster.sol#283) is not in mixedCase
Parameter Booster.setFees(uint256,uint256,uint256,uint256)._stakerFees (convex-platform/contracts/contracts/Booster.sol#283) is not in mixedCase
Parameter Booster.setFees(uint256,uint256,uint256,uint256)._callerFees (convex-platform/contracts/contracts/Booster.sol#283) is not in mixedCase
Parameter Booster.setFees(uint256,uint256,uint256,uint256)._platform (convex-platform/contracts/contracts/Booster.sol#283) is not in mixedCase
Parameter Booster.setTreasury(address)._treasury (convex-platform/contracts/contracts/Booster.sol#305) is not in mixedCase
Parameter Booster.setBridgeDelegate(address)._bridgeDelegate (convex-platform/contracts/contracts/Booster.sol#317) is not in mixedCase
Parameter Booster.addPool(address,address,uint256)._lptoken (convex-platform/contracts/contracts/Booster.sol#339) is not in mixedCase
Parameter Booster.addPool(address,address,uint256)._gauge (convex-platform/contracts/contracts/Booster.sol#339) is not in mixedCase
Parameter Booster.addPool(address,address,uint256)._stashVersion (convex-platform/contracts/contracts/Booster.sol#339) is not in mixedCase
Parameter Booster.shutdownPool(uint256)._pid (convex-platform/contracts/contracts/Booster.sol#386) is not in mixedCase
Parameter Booster.deposit(uint256,uint256,bool)._pid (convex-platform/contracts/contracts/Booster.sol#427) is not in mixedCase
Parameter Booster.deposit(uint256,uint256,bool)._amount (convex-platform/contracts/contracts/Booster.sol#427) is not in mixedCase
Parameter Booster.deposit(uint256,uint256,bool)._stake (convex-platform/contracts/contracts/Booster.sol#427) is not in mixedCase
Parameter Booster.depositAll(uint256,bool)._pid (convex-platform/contracts/contracts/Booster.sol#469) is not in mixedCase
Parameter Booster.depositAll(uint256,bool)._stake (convex-platform/contracts/contracts/Booster.sol#469) is not in mixedCase
Parameter Booster.withdraw(uint256,uint256)._pid (convex-platform/contracts/contracts/Booster.sol#515) is not in mixedCase
Parameter Booster.withdraw(uint256,uint256)._amount (convex-platform/contracts/contracts/Booster.sol#515) is not in mixedCase
Parameter Booster.withdrawAll(uint256)._pid (convex-platform/contracts/contracts/Booster.sol#523) is not in mixedCase
Parameter Booster.withdrawTo(uint256,uint256,address)._pid (convex-platform/contracts/contracts/Booster.sol#533) is not in mixedCase
Parameter Booster.withdrawTo(uint256,uint256,address)._amount (convex-platform/contracts/contracts/Booster.sol#533) is not in mixedCase
Parameter Booster.withdrawTo(uint256,uint256,address)._to (convex-platform/contracts/contracts/Booster.sol#533) is not in mixedCase
Parameter Booster.setVote(bytes32)._hash (convex-platform/contracts/contracts/Booster.sol#544) is not in mixedCase
Parameter Booster.setDelegate(address,address,bytes32)._delegateContract (convex-platform/contracts/contracts/Booster.sol#554) is not in mixedCase
Parameter Booster.setDelegate(address,address,bytes32)._delegate (convex-platform/contracts/contracts/Booster.sol#554) is not in mixedCase
Parameter Booster.setDelegate(address,address,bytes32)._space (convex-platform/contracts/contracts/Booster.sol#554) is not in mixedCase
Parameter Booster.vote(uint256,address,bool)._voteId (convex-platform/contracts/contracts/Booster.sol#563) is not in mixedCase
Parameter Booster.vote(uint256,address,bool)._votingAddress (convex-platform/contracts/contracts/Booster.sol#563) is not in mixedCase
Parameter Booster.vote(uint256,address,bool)._support (convex-platform/contracts/contracts/Booster.sol#563) is not in mixedCase
Parameter Booster.voteGaugeWeight(address[],uint256[])._gauge (convex-platform/contracts/contracts/Booster.sol#574) is not in mixedCase
Parameter Booster.voteGaugeWeight(address[],uint256[])._weight (convex-platform/contracts/contracts/Booster.sol#574) is not in mixedCase
Parameter Booster.claimRewards(uint256,address)._pid (convex-platform/contracts/contracts/Booster.sol#586) is not in mixedCase
Parameter Booster.claimRewards(uint256,address)._gauge (convex-platform/contracts/contracts/Booster.sol#586) is not in mixedCase
Parameter Booster.setGaugeRedirect(uint256)._pid (convex-platform/contracts/contracts/Booster.sol#597) is not in mixedCase
Parameter Booster.earmarkRewards(uint256)._pid (convex-platform/contracts/contracts/Booster.sol#684) is not in mixedCase
Parameter Booster.earmarkFees(address)._feeToken (convex-platform/contracts/contracts/Booster.sol#694) is not in mixedCase
Parameter Booster.rewardClaimed(uint256,address,uint256)._pid (convex-platform/contracts/contracts/Booster.sol#720) is not in mixedCase
Parameter Booster.rewardClaimed(uint256,address,uint256)._address (convex-platform/contracts/contracts/Booster.sol#720) is not in mixedCase
Parameter Booster.rewardClaimed(uint256,address,uint256)._amount (convex-platform/contracts/contracts/Booster.sol#720) is not in mixedCase
Parameter Booster.distributeL2Fees(uint256)._amount (convex-platform/contracts/contracts/Booster.sol#739) is not in mixedCase
Constant Booster.MaxFees (convex-platform/contracts/contracts/Booster.sol#31) is not in UPPER_CASE_WITH_UNDERSCORES
Function ICurveGauge.claim_rewards() (convex-platform/contracts/contracts/Interfaces.sol#10) is not in mixedCase
Function ICurveGauge.reward_tokens(uint256) (convex-platform/contracts/contracts/Interfaces.sol#11) is not in mixedCase
Function ICurveGauge.rewarded_token() (convex-platform/contracts/contracts/Interfaces.sol#12) is not in mixedCase
Function ICurveGauge.lp_token() (convex-platform/contracts/contracts/Interfaces.sol#13) is not in mixedCase
Function ICurveVoteEscrow.create_lock(uint256,uint256) (convex-platform/contracts/contracts/Interfaces.sol#17) is not in mixedCase
Function ICurveVoteEscrow.increase_amount(uint256) (convex-platform/contracts/contracts/Interfaces.sol#18) is not in mixedCase
Function ICurveVoteEscrow.increase_unlock_time(uint256) (convex-platform/contracts/contracts/Interfaces.sol#19) is not in mixedCase
Function ICurveVoteEscrow.smart_wallet_checker() (convex-platform/contracts/contracts/Interfaces.sol#21) is not in mixedCase
Function ICurveVoteEscrow.commit_smart_wallet_checker(address) (convex-platform/contracts/contracts/Interfaces.sol#22) is not in mixedCase
Function ICurveVoteEscrow.apply_smart_wallet_checker() (convex-platform/contracts/contracts/Interfaces.sol#23) is not in mixedCase
Function IVoting.vote_for_gauge_weights(address,uint256) (convex-platform/contracts/contracts/Interfaces.sol#35) is not in mixedCase
Function IRewardFactory.CreateCrvRewards(uint256,address,address) (convex-platform/contracts/contracts/Interfaces.sol#118) is not in mixedCase
Function IRewardFactory.CreateTokenRewards(address,address,address) (convex-platform/contracts/contracts/Interfaces.sol#119) is not in mixedCase
Function IStashFactory.CreateStash(uint256,address,address,uint256) (convex-platform/contracts/contracts/Interfaces.sol#126) is not in mixedCase
Function ITokenFactory.CreateDepositToken(address) (convex-platform/contracts/contracts/Interfaces.sol#130) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable Booster._earmarkRewards(uint256).crvBalBefore (convex-platform/contracts/contracts/Booster.sol#619) is too similar to Booster._earmarkRewards(uint256).crvVBalBefore (convex-platform/contract
Variable Booster.earmarkFees(address).tokenBalanceBBefore (convex-platform/contracts/contracts/Booster.sol#703) is too similar to Booster.earmarkFees(address).tokenBalanceVBefore (convex-platform/con
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

withdrawAll(uint256) should be declared external:
        - Booster.withdrawAll(uint256) (convex-platform/contracts/contracts/Booster.sol#523-528)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

- As a result of the tests carried out with the Slither tool, some results were obtained and reviewed by Halborn. Based on the results reviewed, some vulnerabilities were determined to be false positives.

5.2 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues, and to identify low-hanging fruits on the targets for this engagement. Among the tools used was MythX, a security analysis service for Ethereum smart contracts. MythX performed a scan on all the contracts and sent the compiled results to the analyzers to locate any vulnerabilities.

MythX results:

MasterChefRewardHook.sol

```
Found 5 job(s). Submit? [y/N]: y
  [##################################]  100%
Report for contracts/chef/MasterChefRewardHook.sol
https://dashboard.mythx.io/#/console/analyses/076566e8-08c4-4a3b-8007-c990e963be41
```

Line	SWC Title	Severity	Short Description
10	(SWC-123) Requirement Violation	Low	Requirement violation.
29	(SWC-123) Requirement Violation	Low	Requirement violation.

Booster.sol

```
Found 3 job(s). Submit? [y/N]: y
  [##################################]  100%
Report for convex-platform/contracts/contracts/Booster.sol
https://dashboard.mythx.io/#/console/analyses/c785e628-c414-4daf-b532-d7efdb20ecd6
```

Line	SWC Title	Severity	Short Description
49	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.

- No major issues found by Mythx.

THANK YOU FOR CHOOSING

// HALBORN