PROJECT

Backd Protocol

CLIENT

Backd

REVIEWERS

AkrAn

DATE

June 2022

Table of Contents

Details

- **Client** Backd
- **Date** June 2022
- **Reviewers** AkrAn
- **Repository**: Backd Protocol
- **Commit hash** `7b7f42d699ddf1bbe0b8fb9658a4a4688c46dc66`
- **Technologies**
  - Solidity
  - Python

Issues Summary

SEVERITY	OPEN	CLOSED
Informational	0	2
Minor	0	5
Medium	0	1
Major	0	1

Executive summary

This report represents the results of the engagement with **Backd** to review **Backd Protocol**.

The review was conducted over the course of **2 weeks** from **June 6th to June 17th, 2022**. A total of **10 person-days** were spent reviewing the code.

Scope

The initial review focused on the Backd Protocol repository, identified by the commit hash `7b7f42d699ddf1bbe0b8fb9658a4a4688c46dc66`.

In the last 2 days of the review, I pulled in fixes from the `main` branch of the Backd repository, at hash `30cbfc8bf52ea2148753e6db1c44c853ff9138cb`.

On 24th of June 2022, I pulled in changes for the fixes to some of the outstanding issues in this report from the `main` branch of Backd repository, at hash `77ae1cb1e215f4b82a9189ed3700b995ed629dba`.

I focused on manually reviewing the codebase, searching for security issues such as, but not limited to, re-entrancy problems, transaction ordering, block timestamp dependency, exception handling, call stack depth limitation, integer overflow/underflow, self-destructible contracts, unsecured balance, use of origin, costly gas patterns, architectural problems, code readability.

Given the project's size, I couldn't fully cover all the code in the scope. The file paths that have `(*)` prefix have been covered on a best effort basis.

Includes:

- code/contracts/StakerVault.sol
- code/contracts/pool/EthPool.sol

- code/contracts/pool/Erc20Pool.sol
- code/contracts/pool/PoolFactory.sol
- code/contracts/pool/LiquidityPool.sol
- code/contracts/vault/Erc20Vault.sol
- code/contracts/vault/EthVault.sol
- code/contracts/vault/Vault.sol
- code/contracts/vault/VaultReserve.sol
- code/contracts/vault/VaultStorage.sol
- code/contracts/oracles/ChainlinkOracleProvider.sol
- code/contracts/swappers/SwapperRouter.sol
- code/contracts/access/RoleManager.sol
- code/contracts/access/AuthorizationBase.sol
- code/contracts/access/Authorization.sol
- code/contracts/tokenomics/LpGauge.sol
- code/contracts/LpToken.sol
- code/contracts/tokenomics/BkdToken.sol
- code/contracts/strategies/ConvexStrategyBase.sol
- code/contracts/strategies/BkdEthCvx.sol
- code/contracts/strategies/BkdTriHopCvx.sol
- (*) code/contracts/Controller.sol
- (*) code/contracts/zaps/PoolMigrationZap.sol
- (*) code/contracts/utils/Pausable.sol
- (*) code/contracts/utils/Preparable.sol
- (*) code/contracts/utils/CvxMintAmount.sol
- (*) code/contracts/utils/IPausable.sol
- (*) code/contracts/BkdLocker.sol
- (*) code/contracts/AddressProvider.sol
- (*) code/contracts/CvxCrvRewardsLocker.sol
- (*) code/contracts/actions/topup/TopUpKeeperHelper.sol
- (*) code/contracts/actions/topup/TopUpActionFeeHandler.sol
- (*) code/contracts/actions/topup/TopUpAction.sol
- (*) code/contracts/actions/topup/handlers/CTokenRegistry.sol
- (*) code/contracts/actions/topup/handlers/AaveHandler.sol
- (*) code/contracts/actions/topup/handlers/CompoundHandler.sol
- (*) code/contracts/GasBank.sol
- (*) code/contracts/tokenomics/VestedEscrowRevocable.sol
- (*) code/contracts/tokenomics/AmmConvexGauge.sol
- (*) code/contracts/tokenomics/FeeBurner.sol
- (*) code/contracts/tokenomics/VestedEscrow.sol

- (*) code/contracts/tokenomics/Minter.sol
- (*) code/contracts/tokenomics/KeeperGauge.sol
- (*) code/contracts/tokenomics/InflationManager.sol
- (*) code/contracts/tokenomics/AmmGauge.sol
- (*) code/contracts/RewardHandler.sol

Recommendations

I identified a few possible general improvements that are not security issues during the review, which will bring value to the developers and the community reviewing and using the product.

Increase the number of tests

A good rule of thumb is to have 100% test coverage. This does not guarantee the lack of security problems, but it means that the desired functionality behaves as intended. The negative tests also bring a lot of value because not allowing some actions to happen is also part of the desired behavior.

Issues

Users can claim extra rewards with multiple `LP Gauges` added to `InflationManager`

Status Fixed Severity Major

Description

A user staking their LP Tokens in the `StakerVault` are eligible for rewards. This is done by the `StakerVault.stakeFor` method, which will call `userCheckpoint` on the `LPGauge` contract:

code/contracts/StakerVault.sol#L322-L328

```
function stakeFor(address account, uint256 amount) public override notPaused returns (bool) {
    require(IERC20(token).balanceOf(msg.sender) >= amount, Error.INSUFFICIENT_BALANCE);

    address lpGauge = currentAddresses[_LP_GAUGE];
    if (lpGauge != address(0)) {
        ILpGauge(lpGauge).userCheckpoint(account);
    }
```

After waiting a certain number of blocks to pass, the user can now directly call the `LpGauge.claimRewards` to receive their reward tokens:

code/contracts/tokenomics/LpGauge.sol#L52-L61

```
    function claimRewards(address beneficiary) external override returns (uint256) {
        require(
            msg.sender == beneficiary || _roleManager().hasRole(Roles.GAUGE_ZAP, msg.sender),
            Error.UNAUTHORIZED_ACCESS
        );
        userCheckpoint(beneficiary);
        uint256 amount = perUserShare[beneficiary];
        if (amount <= 0) return 0;
        perUserShare[beneficiary] = 0;
        _mintRewards(beneficiary, amount);
```

Through the `StakerVault.prepareLpGauge` and `executeLpGauge` methods, governance can add a new `LPGauge` to the `InflationManager` contract:

code/contracts/StakerVault.sol#L87-L89

```
    function executeLpGauge() external override onlyGovernance returns (bool) {
        _executeAddress(_LP_GAUGE);
        inflationManager.addGaugeForVault(token);
```

The `addGaugeForVault` method, **adds** this new vault, without disabling the previous one:

code/contracts/tokenomics/InflationManager.sol#L482-L487

```
    function addGaugeForVault(address lpToken) external override returns (bool) {
        IStakerVault _stakerVault = IStakerVault(msg.sender);
        require(addressProvider.isStakerVault(msg.sender, lpToken), Error.UNAUTHORIZED_ACCESS);
        address lpGauge = _stakerVault.getLpGauge();
        require(lpGauge != address(0), Error.GAUGE_DOES_NOT_EXIST);
        gauges[lpGauge] = true;
```

Given this possibility, users that staked funds while the first LPGaguge was active will continue claiming rewards through it even after another LPGauge has been added.

Although some parts of the code seem to suggest that multiple `LPGauges` are a design decision and there is support for such a setup, portions of the code lack awareness of this use case. `StakerVault` code only seems to support one `LPGauge`:

code/contracts/StakerVault.sol#L118-L122

```
        address lpGauge = currentAddresses[_LP_GAUGE];
        if (lpGauge != address(0)) {
            ILpGauge(lpGauge).userCheckpoint(msg.sender);
```

```
            ILpGauge(lpGauge).userCheckpoint(account);
        }
```

Recommendation

Clarify the possibility of using single or multiple `LPGauges` in the code. If only one `LPGauge` is supported, update the code not to allow any actor (Governance or otherwise) to add multiple `LPGauge` contracts to the `InflationManager`.

`SwapperRouter` is vulnerable to price manipulation attacks

Status Fixed Severity Medium

Description

`swap` method can be used to swap a token to another:

code/contracts/swappers/SwapperRouter.sol#L125-L129

```
    function swap(
        address fromToken_,
        address toToken_,
        uint256 amountIn_
    ) public payable override returns (uint256 amountOut) {
```

This method will make use of a Curve Pool or a Uniswap V2 pool for swapping the tokens:

code/contracts/swappers/SwapperRouter.sol#L216-L221

```
            curvePool_.exchange(
                wethIndex_,
                tokenIndex_,
                amount_,
                _minTokenAmountOut(amount_, token_)
            );
```

code/contracts/swappers/SwapperRouter.sol#L249-L255

```
            UniswapRouter02(dex_).swapExactTokensForTokens(
                amount_,
                _getAmountOutMin(amount_, fromToken_, toToken_),
                path_,
                address(this),
                block.timestamp
            )[1];
```

Both Curve and Uniswap allow the caller to specify a minimum expected amount of tokens back, or put it another way: a maximum slippage allowed for the swap. This is a safety mechanism to ensure that an attacker cannot execute a price manipulation attack on the swap.

In order to calculate the minimum tokens expected back from the swap, the `_minTokenAmountOut`, `_minWethAmountOut` and `_getAmountOutMin` methods are being used, which all of them converge on using `_getPriceInEth` method to call an Oracle to check the price in ETH of a given token:

code/contracts/swappers/SwapperRouter.sol#L451-L456

```
function _getPriceInEth(address token_) internal view returns (uint256 tokenPriceInEth) {
    try _addressProvider.getOracleProvider().getPriceETH(token_) returns (uint256 price_) {
        return price_;
    } catch {
        return 0;
    }
```

The issue, however, is the `try/catch` block whereby, if the oracle provider reverts for whatever reason, the `_getPriceInEth` function will return a price of 0 - thus rendering the swap-in-flight vulnerable to a price manipulation attack.

Please note that even though the `ChainlinkOracleProvider._getPrice` method has checks to ensure that the data coming back from the Chainlink feed is valid, the above-mentioned `try/catch` block would silence any revert and return a value of 0:

code/contracts/oracles/ChainlinkOracleProvider.sol#L55-L66

```
    try _feedRegistry.latestRoundData(asset_, denomination_) returns (
        uint80 roundID_,
        int256 price_,
        uint256 startedAt,
        uint256 timeStamp_,
        uint80 answeredInRound_
    ) {
        require(timeStamp_ != 0, Error.ROUND_NOT_COMPLETE);
        require(block.timestamp <= timeStamp_ + stalePriceDelay, Error.STALE_PRICE);
        require(price_ != 0, Error.NEGATIVE_PRICE);
        require(answeredInRound_ >= roundID_, Error.STALE_PRICE);
```

Recommendation

Allow any revert coming from the Oracle Provider `getPriceETH` method to bubble up and revert the whole transaction.

For tokens that do not have reliable Oracles available, you can allow the user to pass on minimum tokens expected for the swap (Uniswap does this with their slippage input in their UI).

I do not recommend ever allowing a swap transaction to be performed where the minimum amount of expected tokens is 0.

Missing pool validation in `ConvexStrategyBase` constructor

Status Fixed Severity Minor

Description

The constructor on `ConvexStrategyBase` contract calls the `poolInfo` method on the Convex Finance Booster contract:

code/contracts/strategies/ConvexStrategyBase.sol#L88

```
        (address lp_, , , address rewards_, , ) = _BOOSTER.poolInfo(convexPid_);
```

The value returned is the `PoolInfo` struct which looks like this:

```
struct PoolInfo {
        address lptoken;
        address token;
        address gauge;
        address crvRewards;
        address stash;
        bool shutdown;
}
```

The code in the constructor therefore can ensure that the `shutdown` flag is `false` during the deployment of the strategy to ensure that the pool is still active.

`LiquidityPool._updateUserFeesOnDeposit` can save on gas costs

Status Fixed Severity Minor

Description

The method makes use of a `storage` variable that is never updated:

code/contracts/pool/LiquidityPool.sol#L759-L764

```
        WithdrawalFeeMeta storage fromMeta = withdrawalFeeMetas[from];
        feeOnDeposit = getNewCurrentFees(
            fromMeta.timeToWait,
            fromMeta.lastActionTimestamp,
            fromMeta.feeRatio
        );
```

Recommendation

Change the `fromMeta` variable from `storage` to `memory`.

`StakerVault.transfer`, `transferFrom`, `approve` methods should check for nil address `account`

Status Fixed Severity Minor

Description

A user can transfer staked tokens to an account by calling the `transfer` method:

code/contracts/StakerVault.sol#L105-L111

```
     * @notice Transfer staked tokens to an account.
     * @dev This is not an ERC20 transfer, as tokens are still owned by this contract, but fees get update
     * @param account Address to transfer to.
     * @param amount Amount to transfer.
     * @return `true` if success.
     */
    function transfer(address account, uint256 amount) external override notPaused returns (bool) {
```

Recommendation

Transferring staked tokens to a nil address doesn't make sense - therefore, this case can be considered a human error and checked against by ensuring that `account` argument is never a nil address. This is also valid for the `transferFrom` and `approve` methods:

code/contracts/StakerVault.sol#L139-L143

```
    function transferFrom(
        address src,
        address dst,
        uint256 amount
    ) external override notPaused returns (bool) {
```

code/contracts/StakerVault.sol#L185-L188

```
    function approve(address spender, uint256 amount) external override notPaused returns (bool) {
        _allowances[msg.sender][spender] = amount;
        emit Approval(msg.sender, spender, amount);
        return true;
```

`StakerVault.transfer` can save on gas

Status Fixed Severity Minor

Description

The `transfer` method can save on gas by using an unchecked arithmetical operation when updating the `balances` state variable for the `msg.sender`:

code/contracts/StakerVault.sol#L124

```
        balances[msg.sender] -= amount;
```

This is safe to do since this value is checked at the beginning of the method to ensure it's greater than or equal to the amount transferred:

code/contracts/StakerVault.sol#L111-L113

```
    function transfer(address account, uint256 amount) external override notPaused returns (bool) {
        require(msg.sender != account, Error.SELF_TRANSFER_NOT_ALLOWED);
        require(balances[msg.sender] >= amount, Error.INSUFFICIENT_BALANCE);
```

Recommendation

You can use `uncheckedSub` to avoid paying for the extra gas Solidity takes to guard against over/underflows.

`EthPool._doTransferIn` unnecessary require statement

Status Fixed Severity Minor

Description

A user can call `LiquidityPool.depositFor` method to deposit funds into the Liquidity Pool in exchange for LP Tokens:

code/contracts/pool/LiquidityPool.sol#L504-L512

```
    function depositFor(
        address account,
        uint256 depositAmount,
        uint256 minTokenAmount
    ) public payable override notPaused returns (uint256) {
        if (depositAmount == 0) return 0;
        uint256 rate = exchangeRate();

        _doTransferIn(msg.sender, depositAmount);
```

In the case of `EthPool` which inherits from the `LiquidityPool` contract, the `_doTransferIn` method looks like this:

code/contracts/pool/EthPool.sol#L20-L23

```
    function _doTransferIn(address from, uint256 amount) internal override {
        require(msg.sender == from, Error.INVALID_SENDER);
        require(msg.value == amount, Error.INVALID_AMOUNT);
    }
```

The `require` statement at line 21 is not needed:

code/contracts/pool/EthPool.sol#L21

```
        require(msg.sender == from, Error.INVALID_SENDER);
```

This is the case because, in the `LiquidityPool.depositFor` method, the argument passed for `from` is the `msg.sender`:

code/contracts/pool/LiquidityPool.sol#L512

```
        _doTransferIn(msg.sender, depositAmount);
```

Which makes the require statement evaluated `msg.sender == msg.sender` and thus consume unnecessary gas for this operation.

Recommendation

Remove the unnecessary `require` statement from line 21 in `EthPool` contract.

`StakerVault.unstakeFor` should follow Checks-Effects-Interactions pattern

Status Fixed Severity Informational

Description

A user can call `unstakeFor` method in order to unstake their LP tokens from the `StakerVault` contract:

code/contracts/StakerVault.sol#L352-L363

```
     * @notice Unstake tokens on behalf of another account.
     * @dev Needs to be approved.
     * @param src Account for which tokens will be unstaked.
     * @param dst Account receiving the tokens.
     * @param amount Amount of token to unstake/receive.
     * @return `true` if success.
     */
    function unstakeFor(
        address src,
        address dst,
        uint256 amount
    ) public override returns (bool) {
```

When unstaking the user's tokens, the method will transfer them to the required `dst` destination address:

code/contracts/StakerVault.sol#L381-L395

```
        IERC20(token).safeTransfer(dst, amount);

        uint256 unstaked = oldBal.uncheckedSub(IERC20(token).balanceOf(address(this)));

        if (src != msg.sender && allowance_ != type(uint256).max && address(pool) != msg.sender) {
            // update allowance
            _allowances[src][msg.sender] -= unstaked;
        }
        balances[src] -= unstaked;

        if (strategies[src]) {
            strategiesTotalStaked -= unstaked;
        } else {
            _poolTotalStaked -= unstaked;
        }
```

The issue, however, is that the accounting updates to the user's balances happen *after* the transfer of tokens. If the `token` supports callbacks (ie. ERC777), the caller of this function will receive execution control and can re-enter the function.

Recommendation

Move the `safeTransfer` call *after* the accounting updates to the state variables. This follows the Checks-Effects-Interactions recommended pattern for dealing with calls to other contracts.

This issue is marked as `Informational` because, in this case, the `token` in question is the LP token which does not support callbacks and is controlled by the Backd governance.

References

Solidity's Documentation

Checks-Effects-Interactions

`LiquidityPool.setStaker` incorrect documentation

Status Fixed Severity Informational

Description

The `setStaker` method mentions that its return value is:

code/contracts/pool/LiquidityPool.sol#L346

```
     * @return Address of the new staker vault for the pool.
```

However, the code will either return `true` or revert:

code/contracts/pool/LiquidityPool.sol#L356-L361

```
        require(stakerVault != address(0), Error.ZERO_ADDRESS_NOT_ALLOWED);
        staker = IStakerVault(stakerVault);
        _approveStakerVaultSpendingLpTokens();
        emit StakerVaultSet(stakerVault);
        return true;
    }
```

Recommendation

Update the documentation to reflect the outcome of the code.

Artifacts

Surya

Sūrya is a utility tool for smart contract systems. It provides a number of visual outputs and information about the structure of smart contracts. It also supports querying the function call graph in multiple ways to aid in the manual inspection and control flow analysis of contracts.

Files Description Table

File Name	SHA
code/contracts/StakerVault.sol	a1bb94ca4a7587b331
code/contracts/pool/EthPool.sol	de9e1c7c2ea91a6e11
code/contracts/pool/Erc20Pool.sol	606349c691ef867406
code/contracts/pool/PoolFactory.sol	076aa99114b1c74c46
code/contracts/pool/LiquidityPool.sol	bdfe85b41c2350bca5
code/contracts/vault/Erc20Vault.sol	f7a06e97dc17c0c780
code/contracts/vault/EthVault.sol	864e84c221e5c5be20
code/contracts/vault/Vault.sol	1e56c340fa584cc2052
code/contracts/vault/VaultReserve.sol	79a6b0c2a994557e55
code/contracts/vault/VaultStorage.sol	a238cb84d6bb6ae83a
code/contracts/oracles/ChainlinkOracleProvider.sol	094d5326adb6e52793
code/contracts/swappers/SwapperRouter.sol	40eb2b070294afa701
code/contracts/access/RoleManager.sol	50e6a48aa0dced0b5c
code/contracts/access/AuthorizationBase.sol	11550510133b1ca970
code/contracts/access/Authorization.sol	2db43762a56c482a0e
code/contracts/tokenomics/LpGauge.sol	6a0218a81a8a2e0e22
code/contracts/LpToken.sol	9d195039ac040470ec
code/contracts/tokenomics/BkdToken.sol	9db2c9e78cd5c57437
code/contracts/strategies/ConvexStrategyBase.sol	a8ae8f40fd5fc0b2b57
code/contracts/strategies/BkdEthCvx.sol	51e0cba17568b9ec99
code/contracts/strategies/BkdTriHopCvx.sol	2fc02aa1803fb4e274b
code/contracts/Controller.sol	47f678568bc083f996e
code/contracts/zaps/PoolMigrationZap.sol	aadb79779154f0d1d6
code/contracts/utils/Pausable.sol	8ac891e1302091a8fc0
code/contracts/utils/Preparable.sol	7cc8674eb26bc25044
code/contracts/utils/CvxMintAmount.sol	936996ae9e484b5697
code/contracts/utils/IPausable.sol	46a22a52bcc2c7806e
code/contracts/BkdLocker.sol	a07e9d05e30485c372
code/contracts/AddressProvider.sol	9fa571adc4b6d0cbb3

File Name	SHA
code/contracts/CvxCrvRewardsLocker.sol	38ef30ec21567f7b4d8
code/contracts/actions/topup/TopUpKeeperHelper.sol	d01cb05afcb239df61e
code/contracts/actions/topup/TopUpActionFeeHandler.sol	7699296bc81b062233
code/contracts/actions/topup/TopUpAction.sol	bd16ffd975e8a9e84e6
code/contracts/actions/topup/handlers/CTokenRegistry.sol	218a69d5496f38a9863
code/contracts/actions/topup/handlers/AaveHandler.sol	482a1f197dfab7e623e
code/contracts/actions/topup/handlers/CompoundHandler.sol	1df6b526fb74eda68bc
code/contracts/GasBank.sol	19cad31182e84eb3e8
code/contracts/tokenomics/VestedEscrowRevocable.sol	5308fabe127c0822a98
code/contracts/tokenomics/AmmConvexGauge.sol	3fe4718c5d11f119157
code/contracts/tokenomics/FeeBurner.sol	a1b970b07818d9b7e3
code/contracts/tokenomics/VestedEscrow.sol	9cb854c2305d4c6e55
code/contracts/tokenomics/Minter.sol	ab93db8319cfaea8abf
code/contracts/tokenomics/KeeperGauge.sol	a1920d9e9179bc1182
code/contracts/tokenomics/InflationManager.sol	0f4300401c6d1329cdc
code/contracts/tokenomics/AmmGauge.sol	fcca56bc236e68f86afc
code/contracts/RewardHandler.sol	b97fd75e54f23c829cb

Contracts Description Table

Contract	Type	Ba
└	**Function Name**	**Visi**
StakerVault	Implementation	IStake Authorizatic Initializable
└		Pub
└	initialize	Exter
└	initializeLpGauge	Exter
└	prepareLpGauge	Exter
└	executeLpGauge	Exter
└	transfer	Exter
└	transferFrom	Exter
└	approve	Exter
└	increaseActionLockedBalance	Exter
└	decreaseActionLockedBalance	Exter
└	poolCheckpoint	Exter
└	getLpGauge	Exter
└	getStakedByActions	Exter
└	allowance	Exter
└	balanceOf	Exter
└	getPoolTotalStaked	Exter
└	stakedAndActionLockedBalanceOf	Exter
└	actionLockedBalanceOf	Exter
└	decimals	Exter
└	getToken	Exter
└	unstake	Pub
└	stake	Pub
└	stakeFor	Pub
└	unstakeFor	Pub
└	_isAuthorizedToPause	Inter

Contract	Type	Ba
EthPool	Implementation	LiquidityPo
└		Pub
└		Exter
└	initialize	Exter
└	getUnderlying	Pub
└	_doTransferIn	Inter
└	_doTransferOut	Inter
└	_getBalanceUnderlying	Inter
└	_getBalanceUnderlying	Inter
Erc20Pool	Implementation	LiquidityPoo
└		Pub
└	initialize	Pub
└	getUnderlying	Pub
└	_doTransferIn	Inter
└	_doTransferOut	Inter
└	_getBalanceUnderlying	Inter
└	_getBalanceUnderlying	Inter
PoolFactory	Implementation	IPoolF Autho
└		Pub
└	addPoolImplementation	Exter
└	addLpTokenImplementation	Exter
└	addVaultImplementation	Exter
└	addStakerVaultImplementation	Exter
└	deployPool	Exter
└	_addImplementation	Inter

Contract	Type	Ba
LiquidityPool	Implementation	ILiquid Authorizatio Pausable,
└		Pub
└	deposit	Exte
└	deposit	Exte
└	depositAndStake	Exte
└	withdrawAll	Exte
└	setLpToken	Exte
└	handleLpTokenTransfer	Exte
└	prepareNewRequiredReserves	Exte
└	executeNewRequiredReserves	Exte
└	resetRequiredReserves	Exte
└	prepareNewReserveDeviation	Exte
└	executeNewReserveDeviation	Exte
└	resetNewReserveDeviation	Exte
└	prepareNewMinWithdrawalFee	Exte
└	executeNewMinWithdrawalFee	Exte
└	resetNewMinWithdrawalFee	Exte
└	prepareNewMaxWithdrawalFee	Exte
└	executeNewMaxWithdrawalFee	Exte
└	resetNewMaxWithdrawalFee	Exte
└	prepareNewWithdrawalFeeDecreasePeriod	Exte
└	executeNewWithdrawalFeeDecreasePeriod	Exte
└	resetNewWithdrawalFeeDecreasePeriod	Exte
└	setStaker	Exte
└	prepareNewVault	Exte
└	executeNewVault	Exte
└	resetNewVault	Exte

Contract	Type	Ba
└	redeem	Exter
└	rebalanceVault	Exter
└	depositFor	Exter
└	unstakeAndRedeem	Exter
└	getLpToken	Exter
└	calcRedeem	Exter
└	getUnderlying	Exter
└	depositFor	Pub
└	redeem	Pub
└	getRequiredReserveRatio	Pub
└	getMaxReserveDeviationRatio	Pub
└	getMinWithdrawalFee	Pub
└	getMaxWithdrawalFee	Pub
└	getWithdrawalFeeDecreasePeriod	Pub
└	getVault	Pub
└	exchangeRate	Pub
└	totalUnderlying	Pub
└	getWithdrawalFee	Pub
└	getNewCurrentFees	Pub
└	_rebalanceVault	Inter
└	_initialize	Inter
└	_approveStakerVaultSpendingLpTokens	Inter
└	_doTransferIn	Inter
└	_doTransferOut	Inter
└	_rebalanceVault	Inter
└	_updateUserFeesOnDeposit	Inter
└	_getBalanceUnderlying	Inter
└	_getBalanceUnderlying	Inter
└	_isAuthorizedToPause	Inter

Contract	Type	Ba
└	_getTime	Inter
└	_checkFeeInvariants	Inter
Erc20Vault	Implementation	Va
└		Pub
└	initialize	Exter
└	getUnderlying	Pub
└	_transfer	Inter
└	_depositToReserve	Inter
└	_depositToRewardHandler	Inter
└	_payStrategist	Inter
└	_availableUnderlying	Inter
EthVault	Implementation	Va
└		Pub
└		Exter
└	initialize	Exter
└	getUnderlying	Pub
└	_transfer	Inter
└	_depositToReserve	Inter
└	_depositToRewardHandler	Inter
└	_payStrategist	Inter
└	_availableUnderlying	Inter
Vault	Implementation	IVault, Au VaultSt Preparable
└		Pub
└	_initialize	Inter
└	deposit	Exter
└	withdraw	Exter

Contract	Type	Ba
└	withdrawAll	Exter
└	withdrawFromReserve	Exter
└	activateStrategy	Exter
└	deactivateStrategy	Exter
└	initializeStrategy	Exter
└	prepareNewStrategy	Exter
└	executeNewStrategy	Exter
└	resetNewStrategy	Exter
└	preparePerformanceFee	Exter
└	executePerformanceFee	Exter
└	resetPerformanceFee	Exter
└	prepareStrategistFee	Exter
└	executeStrategistFee	Exter
└	resetStrategistFee	Exter
└	prepareDebtLimit	Exter
└	executeDebtLimit	Exter
└	resetDebtLimit	Exter
└	prepareTargetAllocation	Exter
└	executeTargetAllocation	Exter
└	resetTargetAllocation	Exter
└	prepareReserveFee	Exter
└	executeReserveFee	Exter
└	resetReserveFee	Exter
└	prepareBound	Exter
└	executeBound	Exter
└	resetBound	Exter
└	withdrawFromStrategy	Exter
└	withdrawFromStrategyWaitingForRemoval	Exter
└	getStrategiesWaitingForRemoval	Exter

Contract	Type	Ba
└	getTotalUnderlying	Exte
└	getAllocatedToStrategyWaitingForRemoval	Exte
└	withdrawAllFromStrategy	Pub
└	harvest	Pub
└	getStrategistFee	Pub
└	getStrategy	Pub
└	getReserveFee	Pub
└	getPerformanceFee	Pub
└	getBound	Pub
└	getTargetAllocation	Pub
└	getDebtLimit	Pub
└	getUnderlying	Pub
└	_activateStrategy	Inter
└	_harvest	Inter
└	_withdrawAllFromStrategy	Inter
└	_handleExcessDebt	Inter
└	_handleExcessDebt	Inter
└	_deposit	Inter
└	_shareProfit	Inter
└	_shareFees	Inter
└	_emergencyStop	Inter
└	_deactivateStrategy	Inter
└	_payStrategist	Inter
└	_payStrategist	Inter
└	_transfer	Inter
└	_depositToReserve	Inter
└	_depositToRewardHandler	Inter
└	_availableUnderlying	Inter
└	_computeNewAllocated	Inter

Contract	Type	Ba
└	_checkFeesInvariant	Inter
└	_rebalance	Priva
VaultReserve	Implementation	IVaultF Autho
└		Pub
└	deposit	Exter
└	withdraw	Exter
└	getBalance	Pub
└	canWithdraw	Pub
VaultStorage	Implementation	
VaultStorageV1	Implementation	VaultS
ChainlinkOracleProvider	Implementation	IChainlinkOr Autho
└		Pub
└	setStalePriceDelay	Exter
└	getPriceETH	Exter
└	getPriceUSD	Pub
└	_getPrice	Inter
SwapperRouter	Implementation	ISwapp Autho
└		Pub
└		Exter
└	swapAll	Exter
└	setSlippageTolerance	Exter
└	setCurvePool	Exter
└	getAmountOut	Exter
└	swap	Pub
└	_swapForWeth	Inter

Contract	Type	Ba
└	_swapWethForToken	Inter
└	_swap	Inter
└	_approve	Inter
└	_returnTokens	Inter
└	_getWethOut	Inter
└	_getTokenOut	Inter
└	_getBestDex	Inter
└	_tokenAmountOut	Inter
└	_getAmountOutMin	Inter
└	_minTokenAmountOut	Inter
└	_minWethAmountOut	Inter
└	_getPriceInEth	Inter
└	_getIndices	Inter
RoleManager	Implementation	IRoleM
└		Pub
└	grantRole	Exter
└	addGovernor	Exter
└	renounceGovernance	Exter
└	addGaugeZap	Exter
└	removeGaugeZap	Exter
└	hasAnyRole	Exter
└	hasAnyRole	Exter
└	hasAnyRole	Exter
└	getRoleMember	Exter
└	revokeRole	Pub
└	getRoleMemberCount	Pub
└	hasRole	Pub
└	_grantRole	Inter

Contract	Type	Ba
└	_revokeRole	Inter
AuthorizationBase	Implementation	
└	roleManager	Exter
└	_roleManager	Inter
Authorization	Implementation	Authoriz
└		Pub
└	_roleManager	Inter
LpGauge	Implementation	ILpG IReward Autho
└		Pub
└	poolCheckpoint	Exter
└	claimRewards	Exter
└	claimableRewards	Exter
└	userCheckpoint	Pub
└	_mintRewards	Inter
└	_poolCheckpoint	Inter
LpToken	Implementation	ILpT ERC20Up
└		Pub
└	initialize	Exter
└	mint	Exter
└	burn	Exter
└	burn	Exter
└	decimals	Pub
└	_beforeTokenTransfer	Inter
BkdToken	Implementation	IBkdToke
└		Pub

Contract	Type	Ba
└	mint	Exter
└	cap	Exter
ConvexStrategyBase	Implementation	IConvexSt Autho CvxMin
└		Pub
└	deposit	Exter
└	withdraw	Exter
└	withdrawAll	Exter
└	harvest	Exter
└	shutdown	Exter
└	setCommunityReserve	Exter
└	setCrvCommunityReserveShare	Exter
└	setCvxCommunityReserveShare	Exter
└	setImbalanceToleranceIn	Exter
└	setImbalanceToleranceOut	Exter
└	setStrategist	Exter
└	addRewardToken	Exter
└	removeRewardToken	Exter
└	harvestable	Exter
└	strategist	Exter
└	rewardTokens	Exter
└	balance	Exter
└	name	Exter
└	hasPendingFunds	Exter
└	_deposit	Inter
└	_withdraw	Inter
└	_withdrawAll	Inter
└	_harvest	Inter

Contract	Type	Ba
└	_sendCommunityReserveShare	Inter
└	_underlyingBalance	Inter
└	_lpBalance	Inter
└	_stakedBalance	Inter
└	_underlyingAmountOut	Inter
└	_validateCurvePool	Inter
BkdEthCvx	Implementation	ConvexSt
└		Pub
└		Exter
└	name	Exter
└	balance	Pub
└	_deposit	Inter
└	_withdraw	Inter
└	_withdrawAll	Inter
└	_underlyingBalance	Inter
└	_minLpAccepted	Inter
└	_maxLpBurned	Inter
└	_minUnderlyingAccepted	Inter
└	_underlyingToLp	Inter
└	_lpToUnderlying	Inter
BkdTriHopCvx	Implementation	ConvexStr IBkdTri
└		Pub
└	setHopImbalanceToleranceIn	Exter
└	setHopImbalanceToleranceOut	Exter
└	changeConvexPool	Exter
└	balance	Pub
└	name	Pub

Contract	Type	Ba
└	_deposit	Inter
└	_withdraw	Inter
└	_withdrawAll	Inter
└	_underlyingBalance	Inter
└	_hopLpBalance	Inter
└	_minLpAccepted	Inter
└	_maxLpBurned	Inter
└	_minHopLpAcceptedFromWithdraw	Inter
└	_minHopLpAcceptedFromDeposit	Inter
└	_maxHopLpBurned	Inter
└	_minUnderlyingAccepted	Inter
└	_underlyingToHopLp	Inter
└	_hopLpToUnderlying	Inter
└	_lpToHopLp	Inter
└	_hopLpToLp	Inter
└	_withdrawAllToHopLp	Priva
Controller	Implementation	IController, A Prep
└		Pub
└	setInflationManager	Exter
└	addStakerVault	Exter
└	removePool	Exter
└	prepareKeeperRequiredStakedBKD	Exter
└	resetKeeperRequiredStakedBKD	Exter
└	executeKeeperRequiredStakedBKD	Exter
└	canKeeperExecuteAction	Exter
└	getTotalEthRequiredForGas	Exter
└	getKeeperRequiredStakedBKD	Pub

Contract	Type	Ba
PoolMigrationZap	Implementation	IPoolMig
└		Pub
└		Exter
└	migrateAll	Exter
└	migrate	Pub
Pausable	Implementation	
└	pause	Exter
└	unpause	Exter
└	_isAuthorizedToPause	Inter
Preparable	Implementation	IPrep
└	_prepareDeadline	Inter
└	_prepare	Inter
└	_prepare	Inter
└	_prepare	Inter
└	_prepare	Inter
└	_resetUInt256Config	Inter
└	_resetAddressConfig	Inter
└	_executeDeadline	Inter
└	_executeUInt256	Inter
└	_executeAddress	Inter
└	_setConfig	Inter
└	_setConfig	Inter
CvxMintAmount	Implementation	
└	getCvxMintAmount	Pub
IPausable	Interface	
└	pause	Exter
└	unpause	Exter
└	isPaused	Exter

Contract	Type	Ba
└	isAuthorizedToPause	Exter
BkdLocker	Implementation	IBkdL Authorizatio
└		Pub
└	initialize	Exter
└	migrate	Exter
└	lock	Exter
└	depositFees	Exter
└	claimFees	Exter
└	userCheckpoint	Exter
└	prepareUnlock	Exter
└	executeUnlocks	Exter
└	getUserShare	Exter
└	boostedBalance	Exter
└	balanceOf	Exter
└	getShareOfTotalBoostedBalance	Exter
└	getStashedGovTokens	Exter
└	claimableFees	Exter
└	claimFees	Pub
└	lockFor	Pub
└	getUserShare	Pub
└	claimableFees	Pub
└	computeNewBoost	Pub
└	_userCheckpoint	Inter
AddressProvider	Implementation	IAddress Authoriza Initializable
└		Pub
└	initialize	Exter

Contract	Type	Ba
└	getKnownAddressKeys	Exter
└	addFeeHandler	Exter
└	removeFeeHandler	Exter
└	addAction	Exter
└	addPool	Exter
└	removePool	Exter
└	allVaults	Exter
└	getVaultAtIndex	Exter
└	vaultsCount	Exter
└	isVault	Exter
└	updateVault	Exter
└	getAddress	Pub
└	getAddress	Pub
└	getAddressMeta	Pub
└	initializeAddress	Exter
└	initializeAddress	Pub
└	initializeAndFreezeAddress	Exter
└	freezeAddress	Exter
└	prepareAddress	Exter
└	executeAddress	Exter
└	resetAddress	Exter
└	addStakerVault	Exter
└	isWhiteListedFeeHandler	Exter
└	safeGetPoolForToken	Exter
└	getPoolForToken	Exter
└	allActions	Exter
└	isAction	Exter
└	isPool	Exter
└	allPools	Exter

Contract	Type	Ba
└	getPoolAtIndex	Exter
└	poolsCount	Exter
└	allStakerVaults	Exter
└	getStakerVault	Exter
└	tryGetStakerVault	Exter
└	isStakerVaultRegistered	Exter
└	isStakerVault	Pub
└	_roleManager	Inter
└	_initializeAddress	Inter
└	_addKnownAddressKey	Inter
CvxCrvRewardsLocker	Implementation	ICvxCrvRev Autho
└		Pub
└	lockCvx	Exter
└	lockCrv	Exter
└	setSpendRatio	Exter
└	claimRewards	Exter
└	stakeCvxCrv	Exter
└	setWithdrawalFlag	Exter
└	resetWithdrawalFlag	Exter
└	processExpiredLocks	Exter
└	setTreasury	Exter
└	withdraw	Exter
└	withdrawCvxCrv	Exter
└	unstakeCvxCrv	Exter
└	unstakeCvxCrv	Exter
└	setDelegate	Exter
└	clearDelegate	Exter
└	forfeitRewards	Exter

Contract	Type	Ba
└	lockRewards	Pub
└	withdraw	Pub
└	unstakeCvxCrv	Pub
└	_lockCrv	Inter
└	_lockCvx	Inter
└	_stakeCvxCrv	Inter
└	_unstakeCvxCrv	Inter
TopUpKeeperHelper	Implementation	ITopUpKe
└		Pub
└	getExecutableTopups	Exter
└	batchCanExecute	Exter
└	listPositions	Pub
└	canExecute	Pub
└	_canExecute	Priva
└	_positionToTopup	Priva
└	_shortenTopups	Priva
TopUpActionFeeHandler	Implementation	IActionFe Authorizatio
└		Pub
└	setInitialKeeperGaugeForToken	Exter
└	payFees	Exter
└	claimKeeperFeesForPool	Exter
└	claimTreasuryFees	Exter
└	prepareKeeperFee	Exter
└	executeKeeperFee	Exter
└	resetKeeperFee	Exter
└	prepareKeeperGauge	Exter
└	executeKeeperGauge	Exter

Contract	Type	Ba
└	resetKeeperGauge	Exter
└	prepareTreasuryFee	Exter
└	executeTreasuryFee	Exter
└	resetTreasuryFee	Exter
└	getKeeperFeeFraction	Pub
└	getKeeperGauge	Pub
└	getTreasuryFeeFraction	Pub
└	_getKeeperGaugeKey	Inter
TopUpActionLibrary	Library	
└	lockFunds	Exter
└	calcExchangeAmount	Exter
└	_approve	Priva
TopUpAction	Implementation	ITopUp Authorizatio Initia
└		Pub
└		Exter
└	initialize	Exter
└	register	Exter
└	execute	Exter
└	resetPosition	Exter
└	executeTopUpHandler	Exter
└	resetTopUpHandler	Exter
└	prepareActionFee	Exter
└	executeActionFee	Exter
└	resetActionFee	Exter
└	prepareFeeHandler	Exter
└	executeFeeHandler	Exter

Contract	Type	Ba
└	resetFeeHandler	Exter
└	prepareEstimatedGasUsage	Exter
└	executeEstimatedGasUsage	Exter
└	resetGasUsage	Exter
└	addUsableToken	Exter
└	getEthRequiredForGas	Exter
└	getUserPositions	Exter
└	getSupportedProtocols	Exter
└	usersWithPositions	Exter
└	getUsableTokens	Exter
└	getTopUpHandler	Exter
└	execute	Pub
└	prepareTopUpHandler	Pub
└	getHealthFactor	Pub
└	getHandler	Pub
└	getEstimatedGasUsage	Pub
└	getActionFee	Pub
└	getFeeHandler	Pub
└	getPosition	Pub
└	isUsable	Pub
└	_updateTopUpHandler	Inter
└	_payFees	Inter
└	_lockFunds	Inter
└	_removePosition	Inter
└	_removeUserPosition	Inter
└	_approve	Inter
└	_calcExchangeAmount	Inter
└	_getHandler	Inter
└	_isSwappable	Inter

Contract	Type	Ba
└	_getProtocolKey	Inter
CTokenRegistry	Implementation	ICToker
└		Pub
└	fetchCToken	Exter
└	getCToken	Exter
└	getCToken	Pub
└	_updateCTokenMapping	Inter
└	_isCTokenUsable	Inter
AaveHandler	Implementation	ITopUp
└		Pub
└	topUp	Exter
└	getUserFactor	Exter
└	_approve	Inter
CompoundHandler	Implementation	ITopUp Exponent
└		Pub
└	topUp	Exter
└	getUserFactor	Exter
└	_repayAnyDebt	Inter
└	_approve	Inter
└	_getAccountBorrowsAndSupply	Inter
GasBank	Implementation	IGas
└		Pub
└	depositFor	Exter
└	withdrawFrom	Exter
└	withdrawUnused	Exter
└	balanceOf	Exter
└	withdrawFrom	Pub

Contract	Type	Ba
└	_withdrawFrom	Inter
VestedEscrowRevocable	Implementation	IVestedEscr Vestec
└		Pub
└	claim	Exter
└	revoke	Exter
└	vestedOf	Exter
└	balanceOf	Exter
└	lockedOf	Exter
└	claim	Pub
AmmConvexGauge	Implementation	IAmmCon Amm CvxMin
└		Pub
└	claimRewards	Exter
└	setInflationRecipient	Exter
└	deactivateInflationRecipient	Exter
└	claimableRewards	Exter
└	allClaimableRewards	Exter
└	stakeFor	Pub
└	unstakeFor	Pub
└	poolCheckpoint	Pub
└	_userCheckpoint	Inter
FeeBurner	Implementation	IFeeB
└		Pub
└		Exter
└	burnToTarget	Pub
└	_depositInPool	Inter

Contract	Type	Ba
└	_approve	Inter
└	_swapperRouter	Inter
EscrowTokenHolder	Implementation	
└		Pub
VestedEscrow	Implementation	IVestec Reentra
└		Pub
└	setAdmin	Exter
└	setFundAdmin	Exter
└	initializeUnallocatedSupply	Exter
└	fund	Exter
└	claim	Exter
└	vestedSupply	Exter
└	lockedSupply	Exter
└	vestedOf	Exter
└	balanceOf	Exter
└	lockedOf	Exter
└	claim	Pub
└	_claimUntil	Inter
└	_computeVestedAmount	Inter
└	_totalVestedOf	Inter
└	_totalVested	Inter
└	_balanceOf	Inter
Minter	Implementation	IMinter, Au Reentra
└		Pub
└	setToken	Exter
└	startInflation	Exter

Contract	Type	Ba
└	executeInflationRateUpdate	Exter
└	mint	Exter
└	mintNonInflationTokens	Exter
└	getLpInflationRate	Exter
└	getKeeperInflationRate	Exter
└	getAmmInflationRate	Exter
└	_executeInflationRateUpdate	Inter
└	_mint	Inter
KeeperGauge	Implementation	IKeepe Autho
└		Pub
└	kill	Exter
└	reportFees	Exter
└	advanceEpoch	Exter
└	claimRewards	Exter
└	claimableRewards	Exter
└	poolCheckpoint	Pub
└	claimRewards	Pub
└	_mintRewards	Inter
└	_calcTotalClaimable	Inter
InflationManager	Implementation	Autho IInflation Prep
└		Pub
└	setMinter	Exter
└	advanceKeeperGaugeEpoch	Exter
└	mintRewards	Exter
└	deactivateWeightBasedKeeperDistribution	Exter
└	checkpointAllGauges	Exter

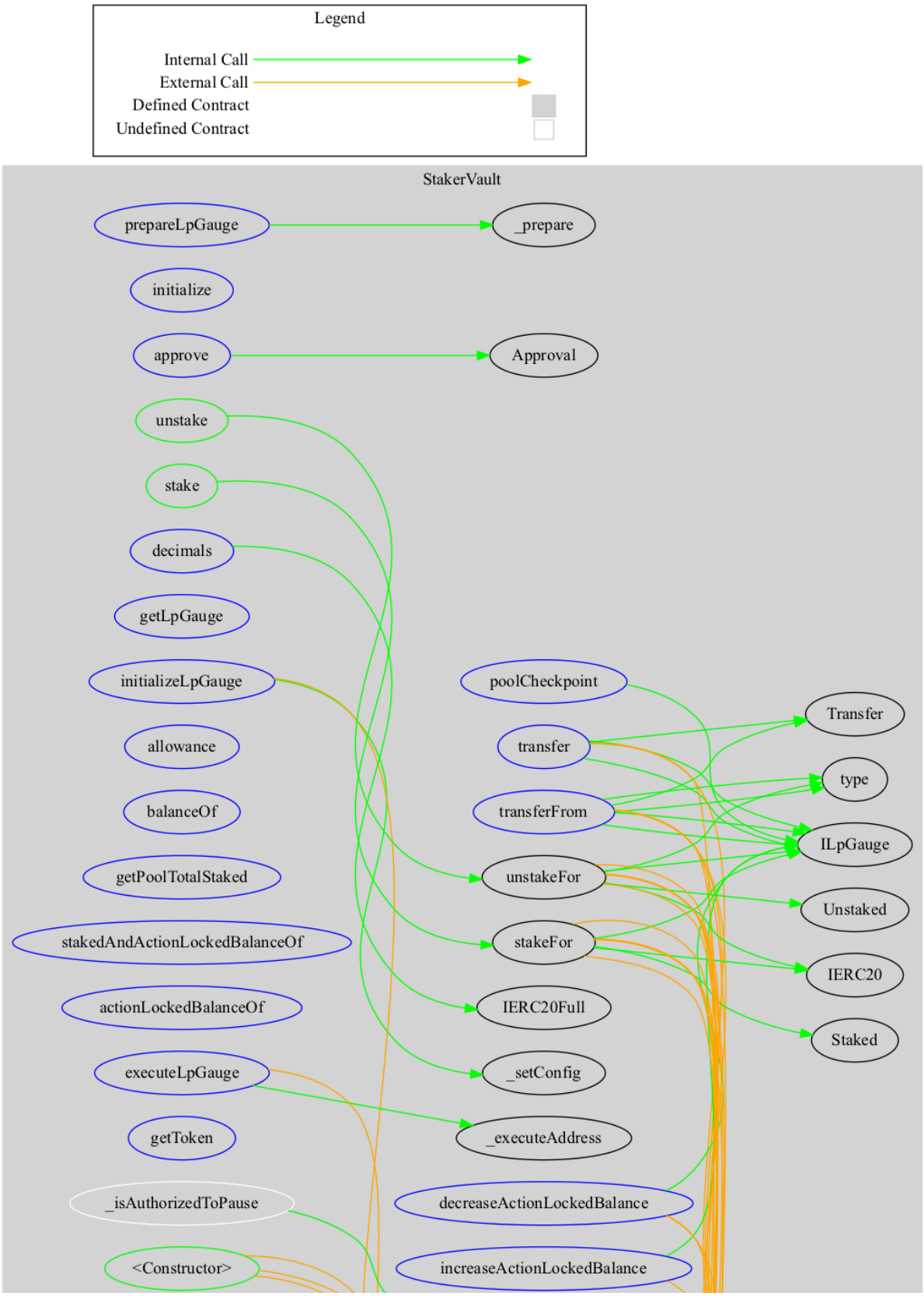
Contract	Type	Ba
└	prepareKeeperPoolWeight	Exter
└	executeKeeperPoolWeight	Exter
└	batchPrepareKeeperPoolWeights	Exter
└	whitelistGauge	Exter
└	batchExecuteKeeperPoolWeights	Exter
└	removeStakerVaultFromInflation	Exter
└	prepareLpPoolWeight	Exter
└	executeLpPoolWeight	Exter
└	batchPrepareLpPoolWeights	Exter
└	batchExecuteLpPoolWeights	Exter
└	prepareAmmTokenWeight	Exter
└	executeAmmTokenWeight	Exter
└	batchPrepareAmmTokenWeights	Exter
└	batchExecuteAmmTokenWeights	Exter
└	setKeeperGauge	Exter
└	removeKeeperGauge	Exter
└	setAmmGauge	Exter
└	removeAmmGauge	Exter
└	addGaugeForVault	Exter
└	getAllAmmGauges	Exter
└	getLpRateForStakerVault	Exter
└	getKeeperRateForPool	Exter
└	getAmmRateForToken	Exter
└	getKeeperWeightForPool	Exter
└	getAmmWeightForToken	Exter
└	getLpPoolWeight	Exter
└	getKeeperGaugeForPool	Exter
└	getAmmGaugeForToken	Exter
└	isInflationWeightManager	Pub

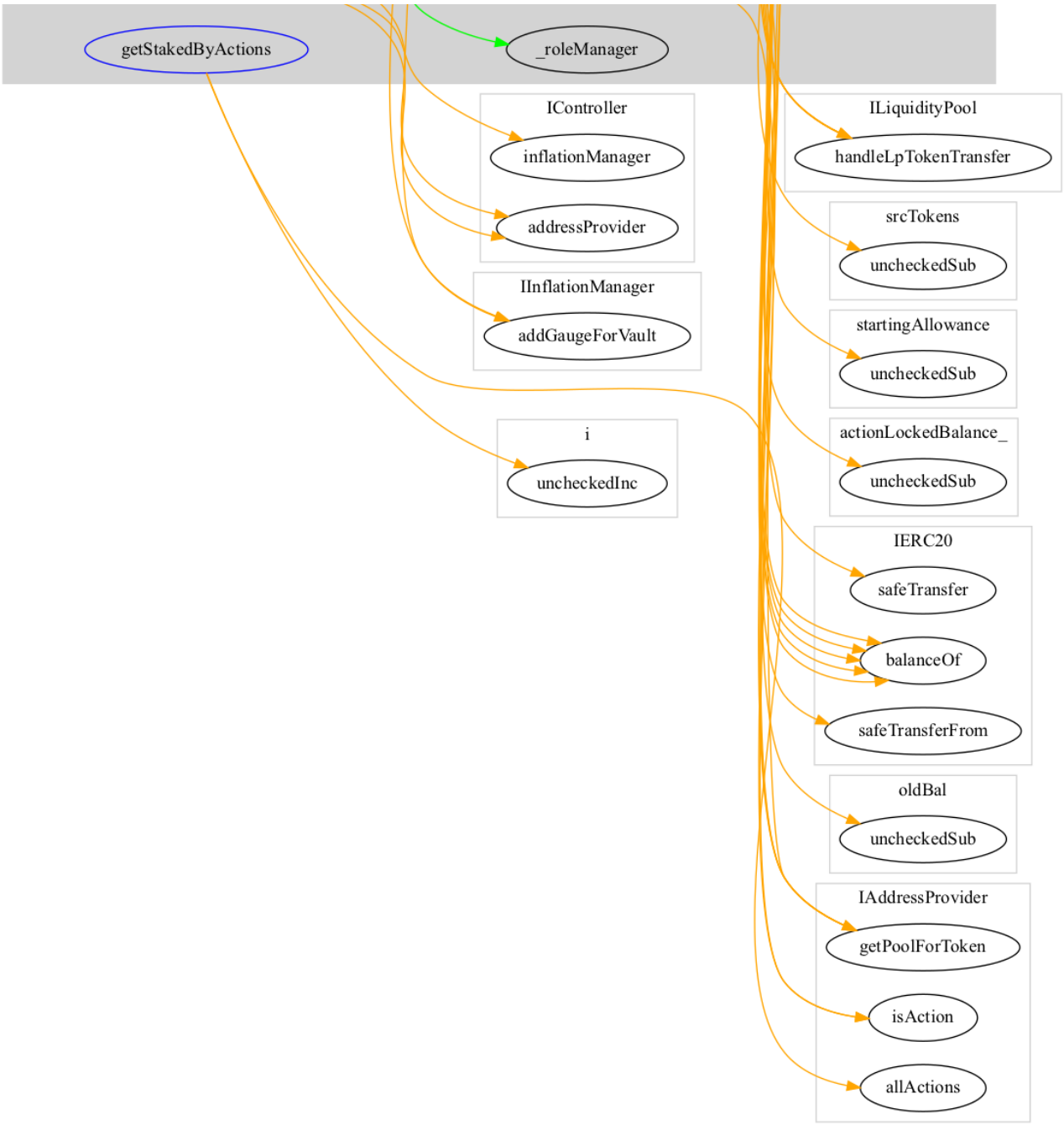
Contract	Type	Ba
└	_executeKeeperPoolWeight	Inter
└	_executeLpPoolWeight	Inter
└	_executeAmmTokenWeight	Inter
└	_removeKeeperGauge	Inter
└	_ensurePoolExists	Inter
└	_getKeeperGaugeKey	Inter
└	_getAmmGaugeKey	Inter
└	_getLpStakerVaultKey	Inter
AmmGauge	Implementation	Autho IAmm
└		Pub
└	kill	Exte
└	claimRewards	Exte
└	stake	Exte
└	unstake	Exte
└	getAmmToken	Exte
└	isAmmToken	Exte
└	claimableRewards	Exte
└	stakeFor	Pub
└	unstakeFor	Pub
└	poolCheckpoint	Pub
└	_userCheckpoint	Inter
RewardHandler	Implementation	IReward Preparable,
└		Pub
└		Exte
└	burnFees	Exte
└	_approve	Inter

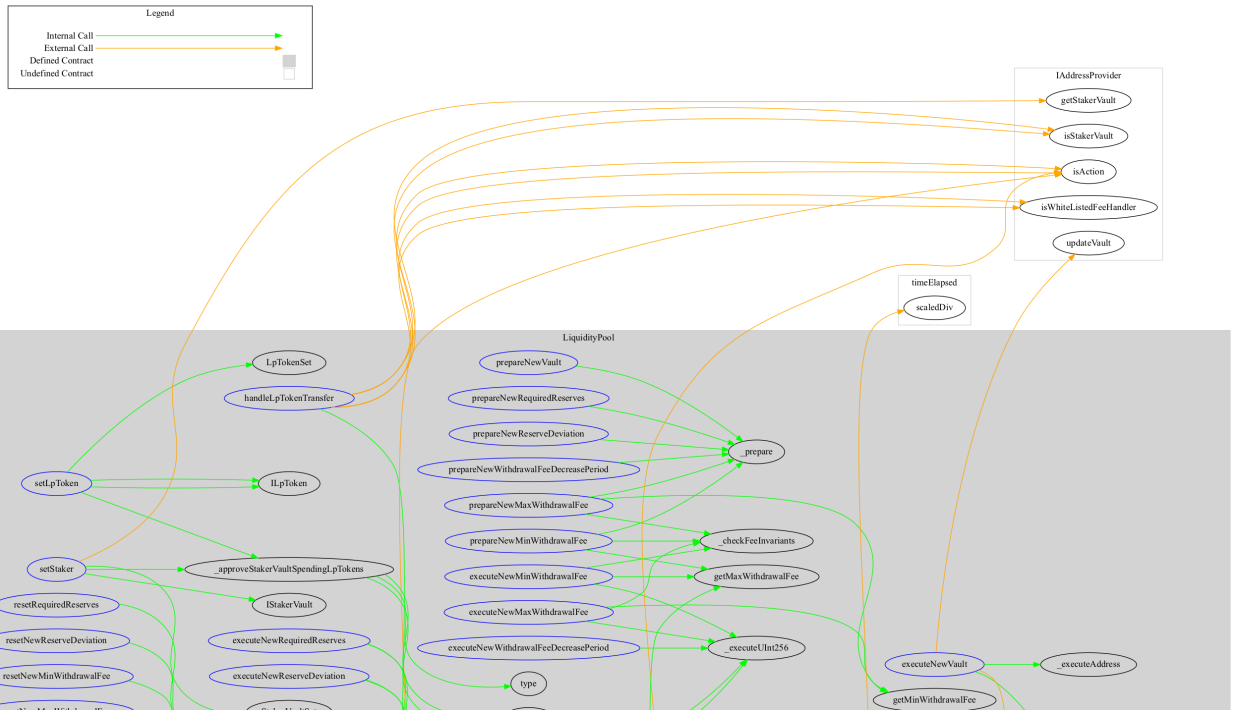
Legend

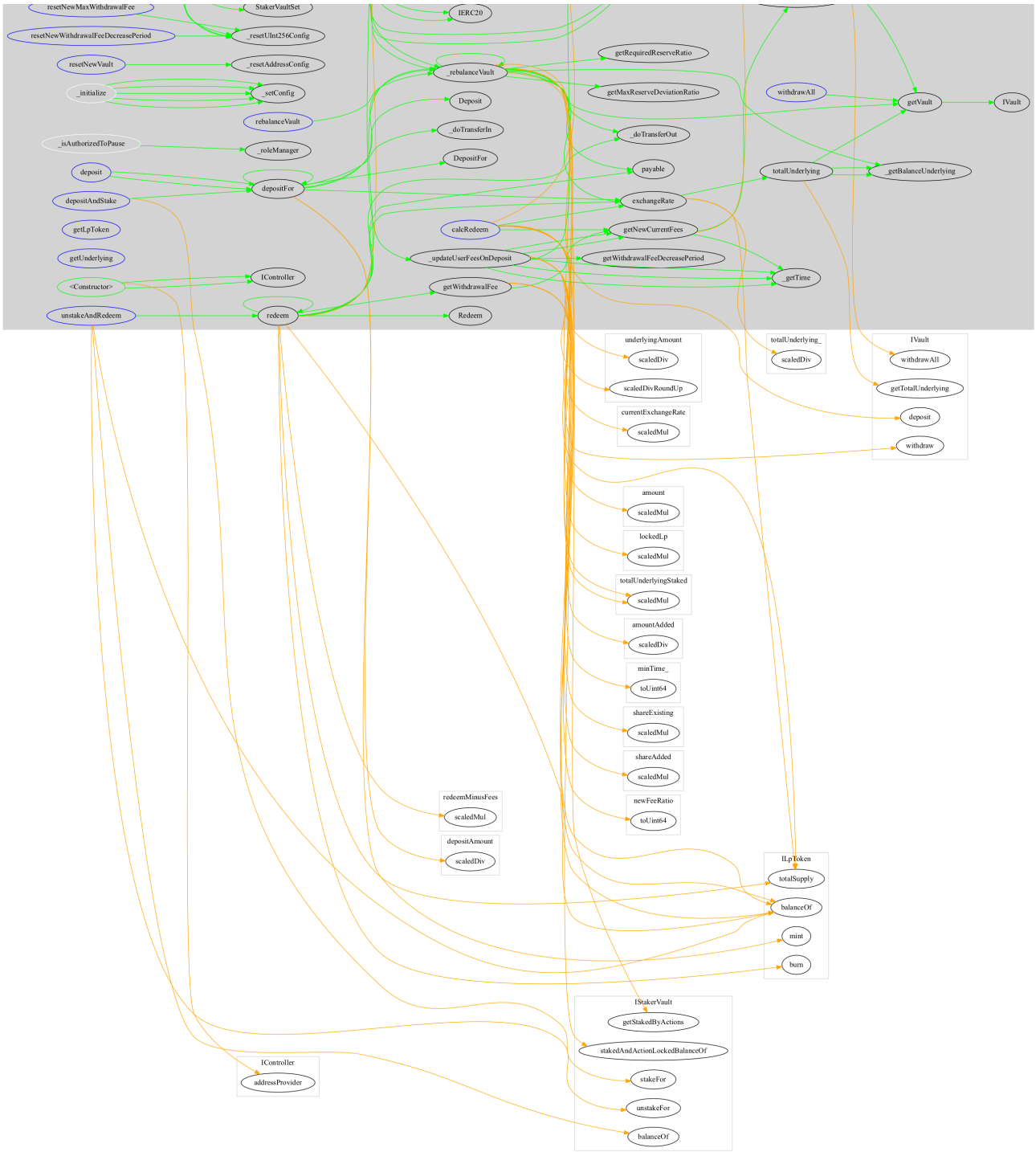
Symbol	Meaning
	Function can modify state
	Function is payable

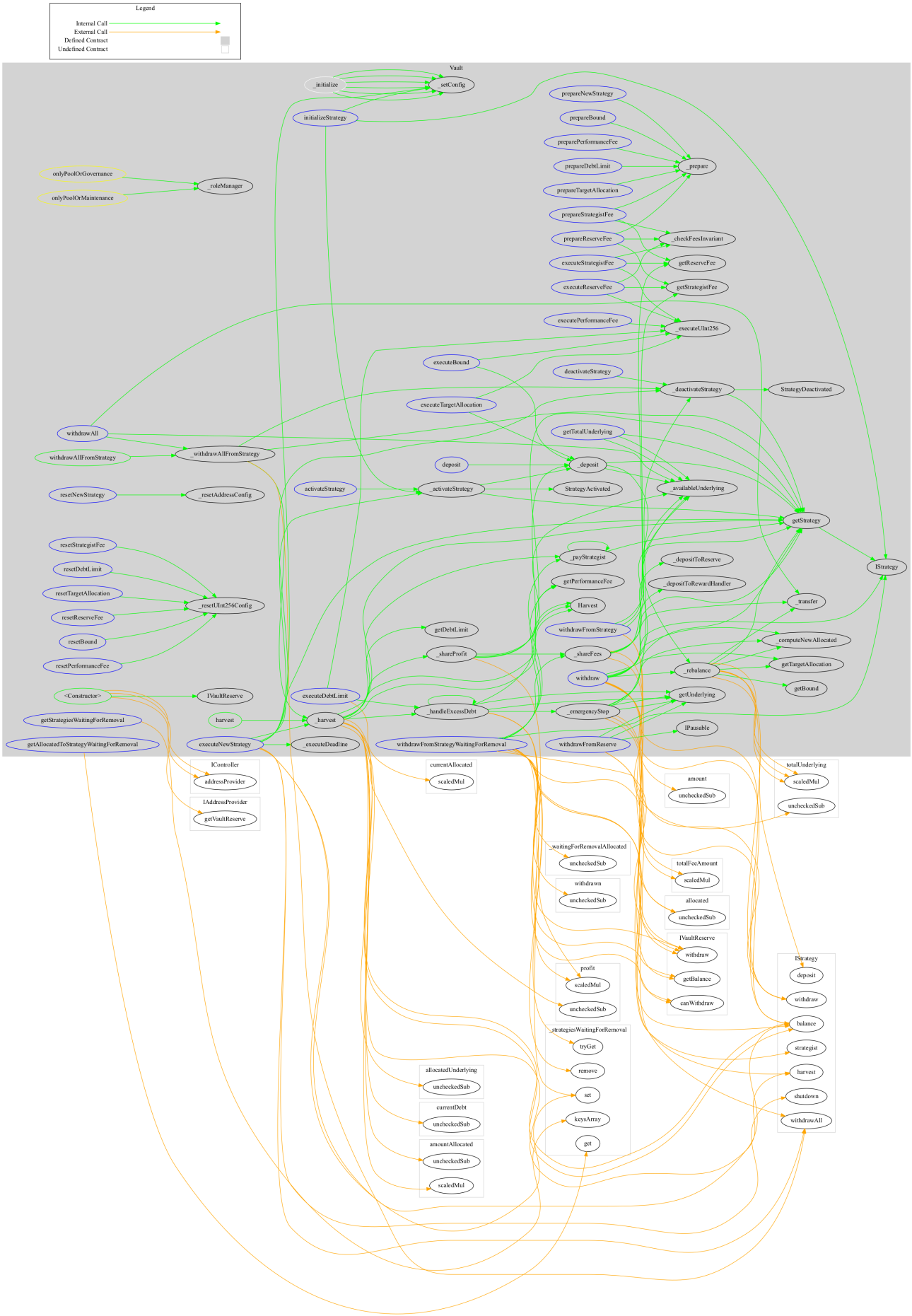
Graphs

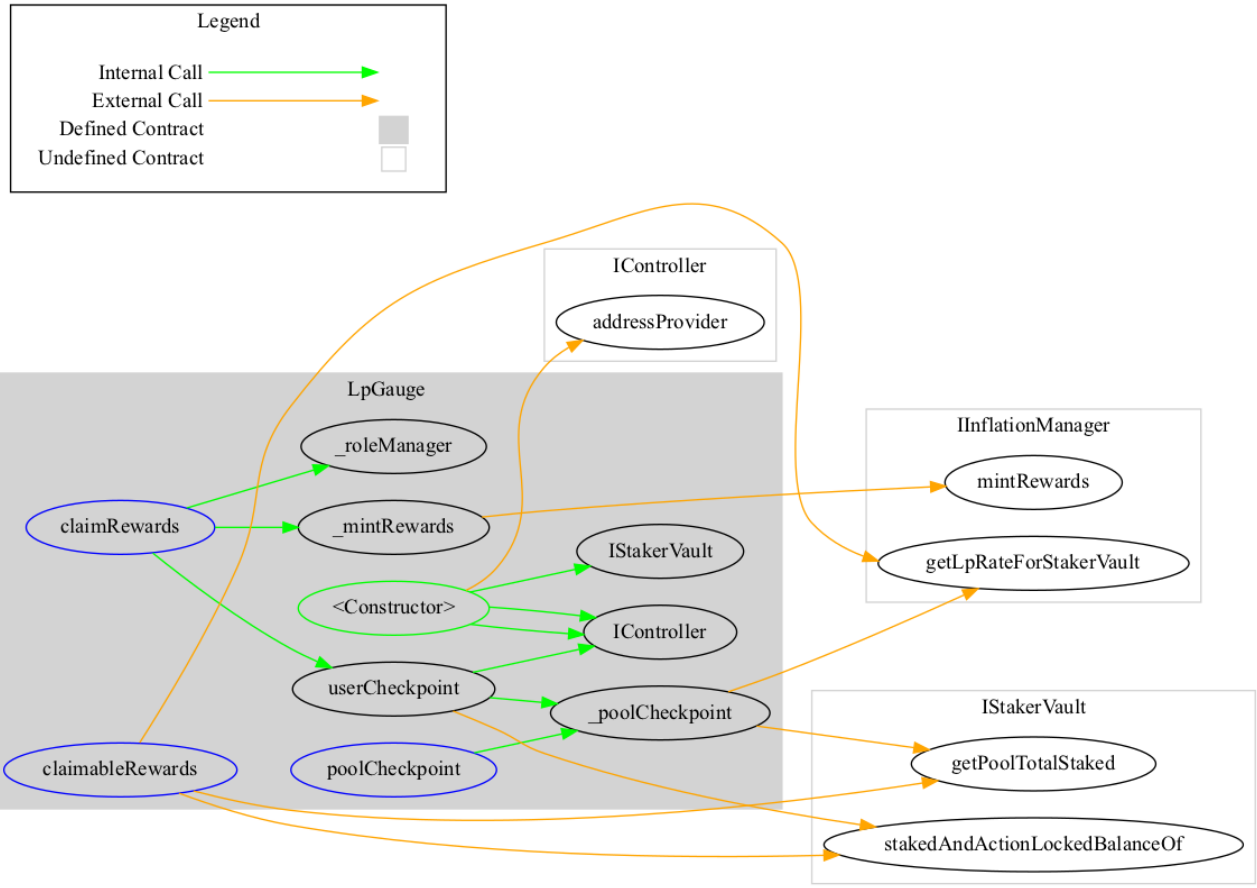


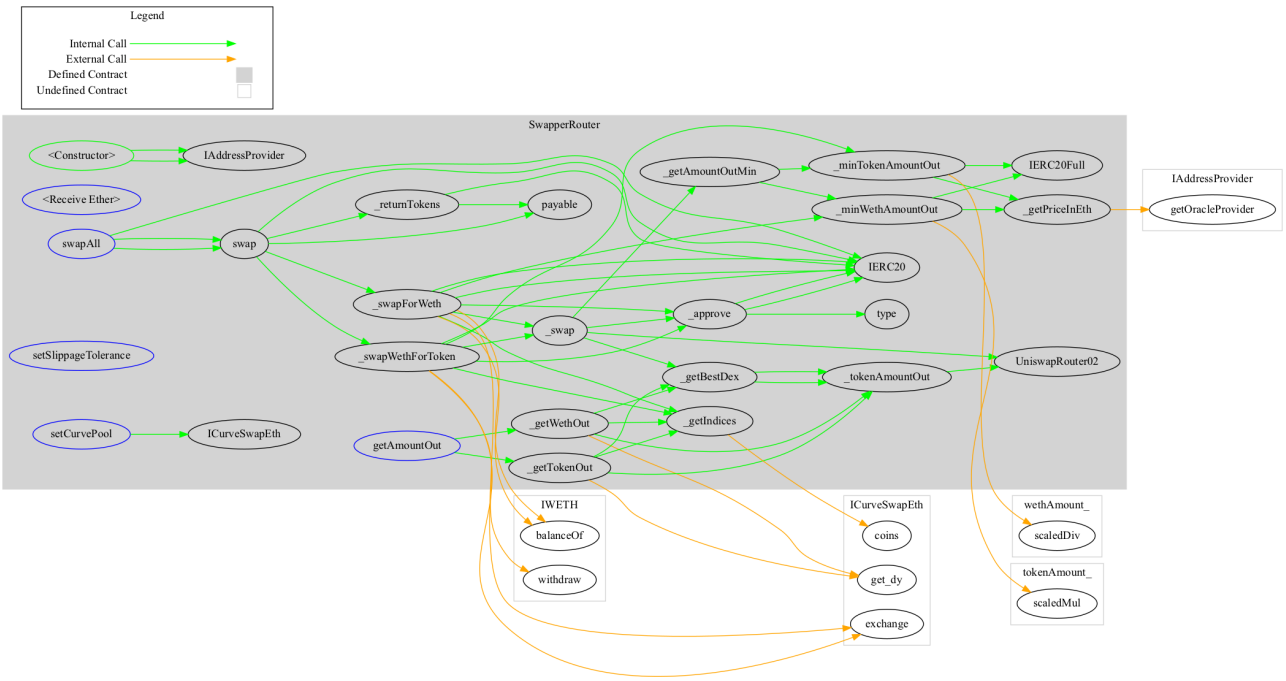

getStakedByActions
_roleManager
IController
inflationManager
addressProvider
IInflationManager
addGaugeForVault
i
uncheckedInc
ILiquidityPool
handleLpTokenTransfer
srcTokens
uncheckedSub
startingAllowance
uncheckedSub
actionLockedBalance_
uncheckedSub
IERC20
safeTransfer
balanceOf
safeTransferFrom
oldBal
uncheckedSub
IAddressProvider
getPoolForToken
isAction
allActions

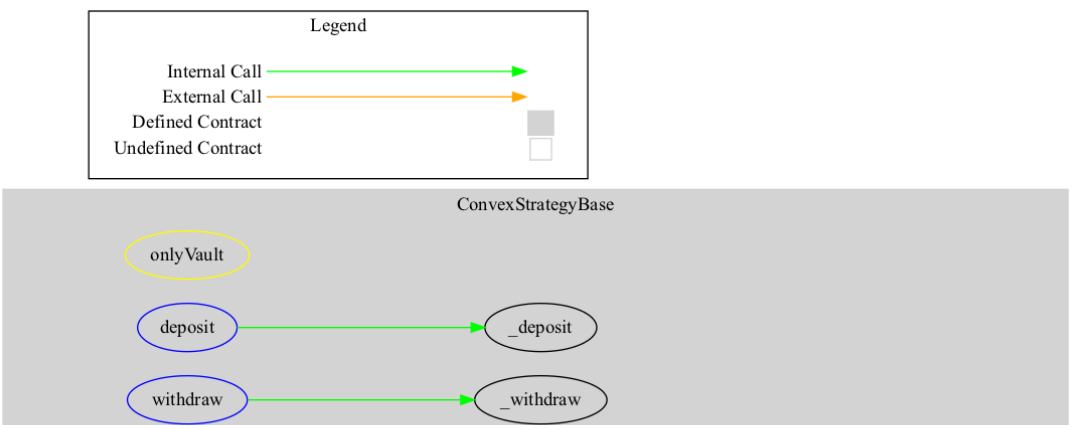

Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IAddressProvider
getStakerVault
isStakerVault
isAction
isWhiteListedFeeHandler
updateVault
timeElapsed
scaledDiv
LiquidityPool
LpTokenSet
handleLpTokenTransfer
setLpToken
ILpToken
setStaker
_approveStakerVaultSpendingLpTokens
IStakerVault
resetRequiredReserves
resetNewReserveDeviation
resetNewMinWithdrawalFee
executeNewRequiredReserves
executeNewReserveDeviation
prepareNewVault
prepareNewRequiredReserves
prepareNewReserveDeviation
prepareNewWithdrawalFeeDecreasePeriod
prepareNewMaxWithdrawalFee
prepareNewMinWithdrawalFee
executeNewMinWithdrawalFee
executeNewMaxWithdrawalFee
executeNewWithdrawalFeeDecreasePeriod
prepare
_checkFeeInvariants
getMaxWithdrawalFee
executeUInt256
type
executeNewVault
_executeAddress
getMinWithdrawalFee


resetNewMaxWithdrawalFee
StakerVaultSet
IERC20
resetNewWithdrawalFeeDecreasePeriod
_resetUint256Config
resetNewVault
_resetAddressConfig
_rebalanceVault
getRequiredReserveRatio
_initialize
_setConfig
getMaxReserveDeviationRatio
withdrawAll
getVault
IVault
Deposit
rebalanceVault
_doTransferIn
_doTransferOut
_isAuthorizedToPause
_roleManager
DepositFor
payable
totalUnderlying
_getBalanceUnderlying
deposit
depositFor
exchangeRate
depositAndStake
getLpToken
calcRedeem
getNewCurrentFees
getUnderlying
_updateUserFeesOnDeposit
getWithdrawalFeeDecreasePeriod
_getTime
<Constructor>
IController
getWithdrawalFee
unstakeAndRedeem
redeem
Redeem
underlyingAmount
scaledDiv
scaledDivRoundUp
totalUnderlying_
scaledDiv
IVault
withdrawAll
getTotalUnderlying
currentExchangeRate
scaledMul
deposit
withdraw
amount
scaledMul
lockedLp
scaledMul
totalUnderlyingStaked
scaledMul
amountAdded
scaledDiv
minTime_
toUint64
shareExisting
scaledMul
shareAdded
scaledMul
newFeeRatio
toUint64
redeemMinusFees
scaledMul
depositAmount
scaledDiv
ILpToken
totalSupply
balanceOf
mint
burn
IStakerVault
getStakedByActions
stakedAndActionLockedBalanceOf
stakeFor
unstakeFor
balanceOf
IController
addressProvider

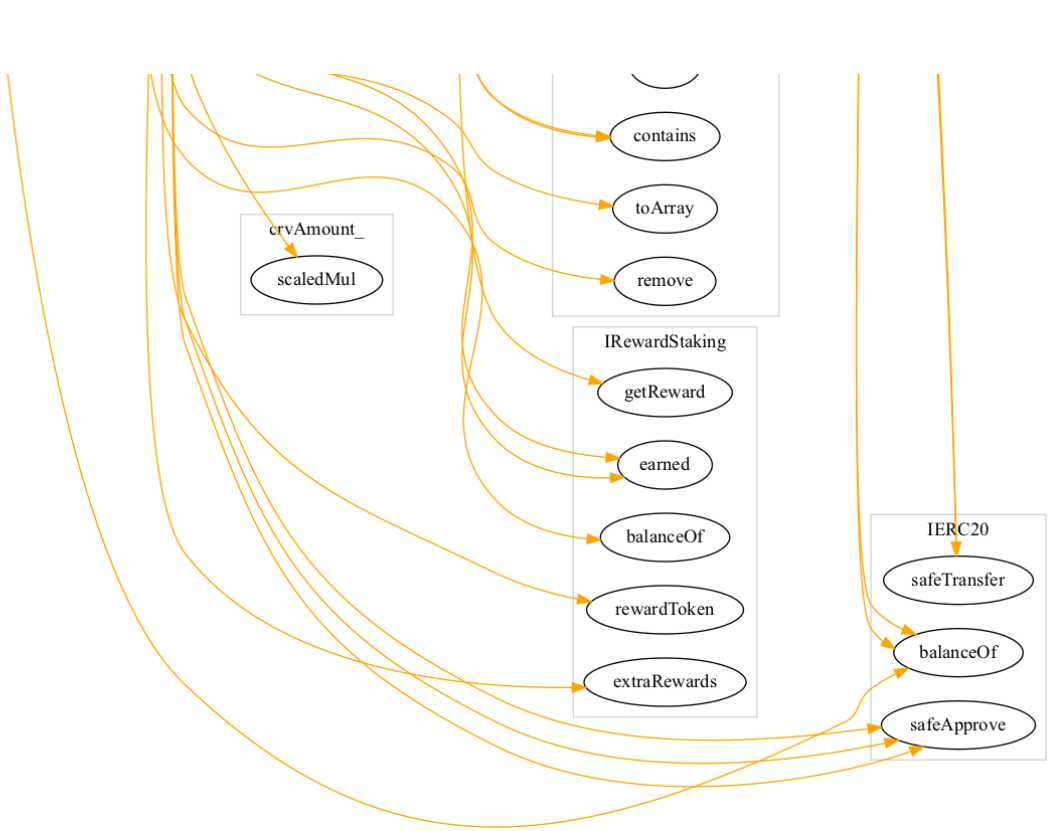

Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Vault
_initialize
setConfig
initializeStrategy
prepareNewStrategy
prepareBound
preparePerformanceFee
prepareDebtLimit
prepareTargetAllocation
prepareStrategistFee
prepareReserveFee
executeStrategistFee
executeReserveFee
executePerformanceFee
_prepare
_checkFeesInvariant
getReserveFee
getStrategistFee
_executeUint256
onlyPoolOrGovernance
onlyPoolOrMaintenance
_roleManager
executeBound
deactivateStrategy
executeTargetAllocation
_deactivateStrategy
StrategyDeactivated
withdrawAll
withdrawAllFromStrategy
_withdrawAllFromStrategy
getTotalUnderlying
deposit
_deposit
activateStrategy
_activateStrategy
StrategyActivated
_availableUnderlying
resetNewStrategy
_resetAddressConfig
getStrategy
IStrategy
resetStrategistFee
resetDebtLimit
resetTargetAllocation
resetReserveFee
resetBound
resetPerformanceFee
_resetUint256Config
_payStrategist
getPerformanceFee
_depositToReserve
_depositToRewardHandler
Harvest
withdrawFromStrategy
_transfer
getDebtLimit
shareProfit
_shareFees
_computeNewAllocated
withdraw
_rebalance
getTargetAllocation
getBound
<Constructor>
IVaultReserve
executeDebtLimit
_handleExcessDebt
getUnderlying
getStrategiesWaitingForRemoval
harvest
_harvest
_emergencyStop
IPausable
getAllocatedToStrategyWaitingForRemoval
executeNewStrategy
_executeDeadline
withdrawFromStrategyWaitingForRemoval
withdrawFromReserve
IController
addressProvider
IAddressProvider
getVaultReserve
currentAllocated
scaledMul
amount
uncheckedSub
totalUnderlying
scaledMul
uncheckedSub
_waitingForRemovalAllocated
uncheckedSub
totalFeeAmount
scaledMul
withdrawn
uncheckedSub
allocated
uncheckedSub
IVaultReserve
withdraw
getBalance
canWithdraw
IStrategy
deposit
withdraw
balance
strategist
harvest
shutdown
withdrawAll
profit
scaledMul
uncheckedSub
_strategiesWaitingForRemoval
tryGet
remove
set
keysArray
get
allocatedUnderlying
uncheckedSub
currentDebt
uncheckedSub
amountAllocated
uncheckedSub
scaledMul

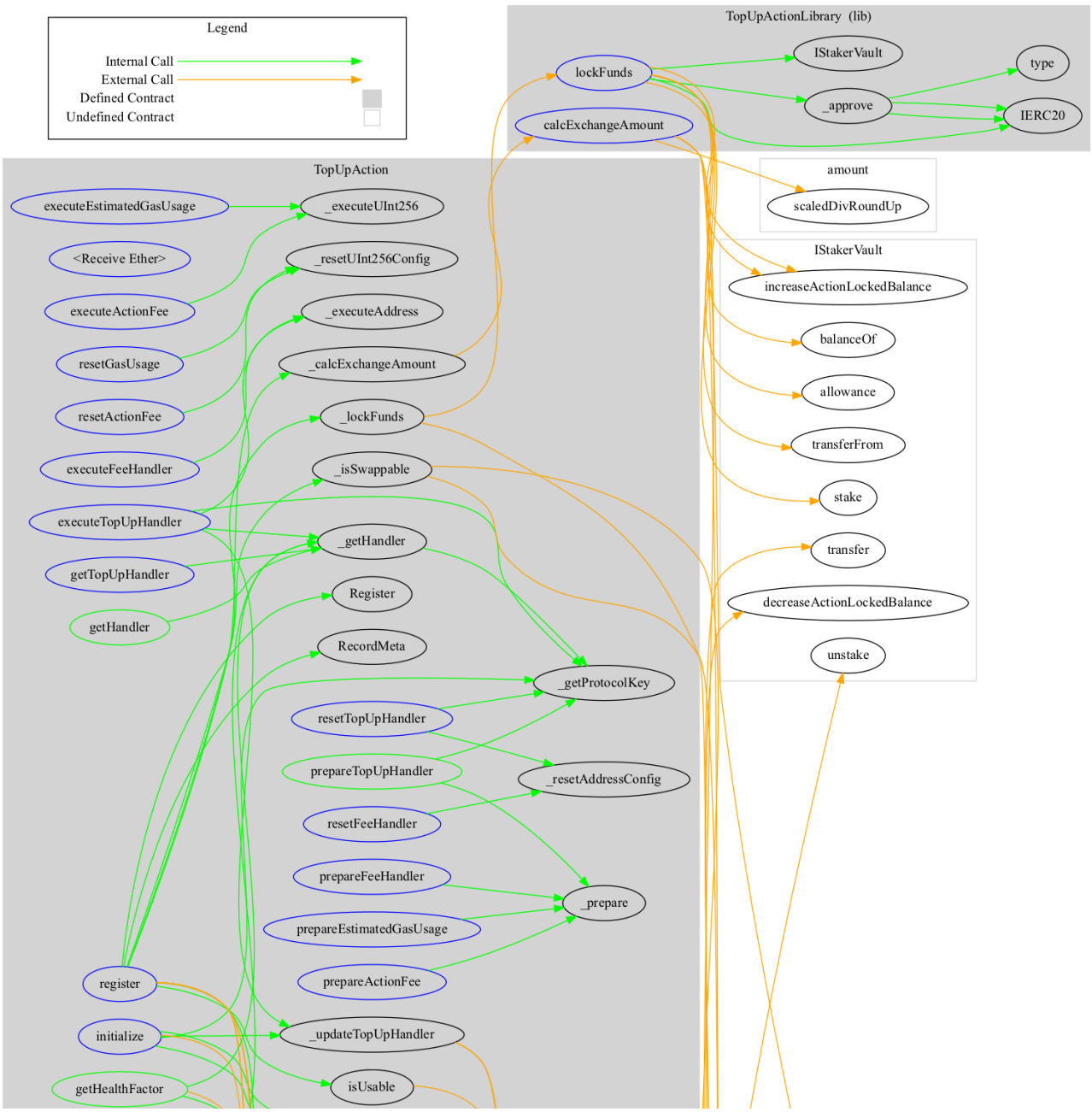

Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IController
addressProvider
LpGauge
_roleManager
claimRewards
_mintRewards
IStakerVault
<Constructor>
IController
userCheckpoint
_poolCheckpoint
claimableRewards
poolCheckpoint
IInflationManager
mintRewards
getLpRateForStakerVault
IStakerVault
getPoolTotalStaked
stakedAndActionLockedBalanceOf

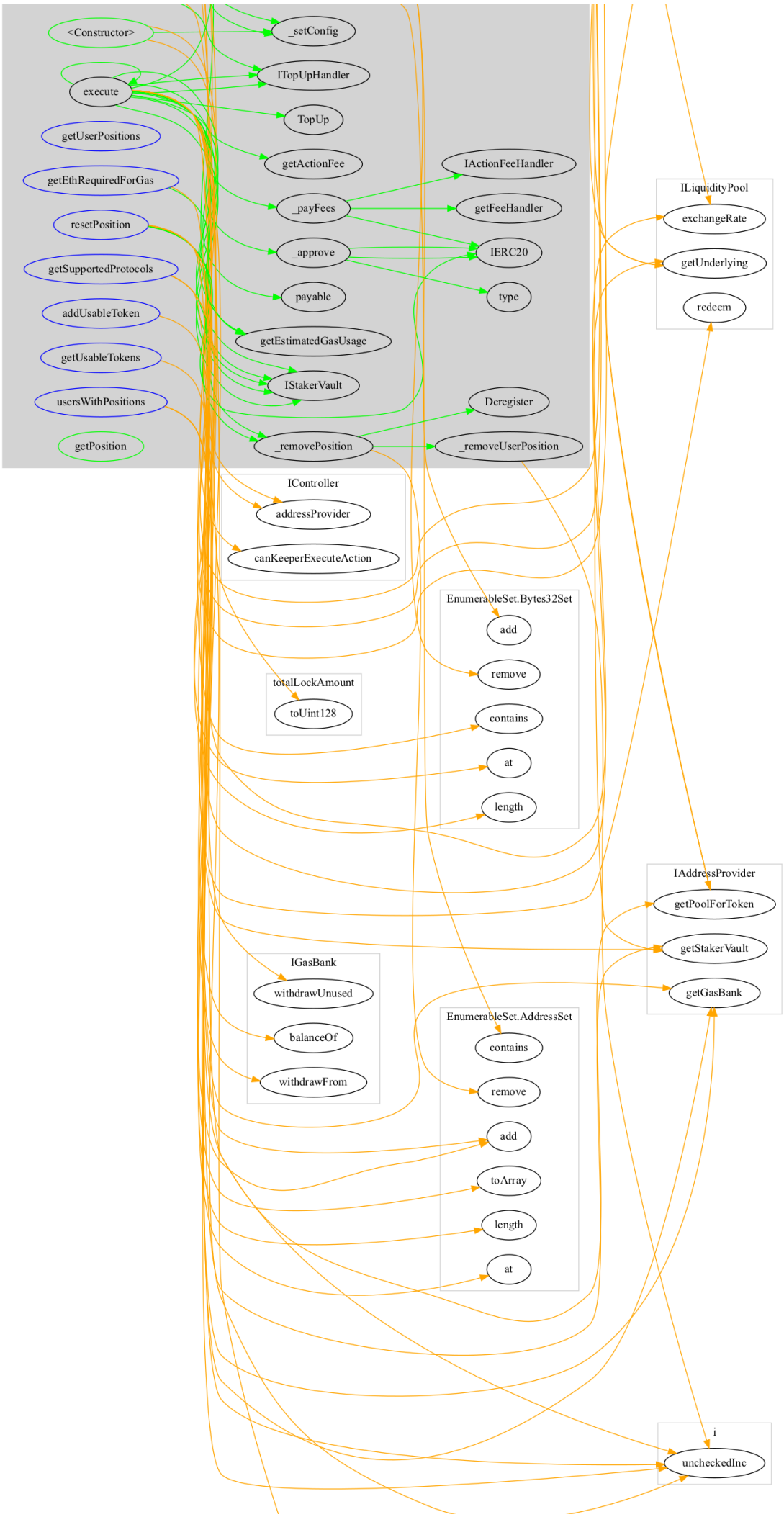

Legend
Internal Call
External Call
Defined Contract
Undefined Contract
SwapperRouter
<Constructor>
IAddressProvider
<Receive Ether>
swapAll
swap
_returnTokens
payable
_getAmountOutMin
_minTokenAmountOut
_minWethAmountOut
IERC20Full
_getPriceInEth
IAddressProvider
getOracleProvider
IERC20
_swapForWeth
_approve
type
_swap
setSlippageTolerance
_swapWethForToken
_getBestDex
_tokenAmountOut
UniswapRouter02
setCurvePool
ICurveSwapEth
getAmountOut
_getWethOut
_getIndices
_getTokenOut
IWETH
balanceOf
withdraw
ICurveSwapEth
coins
get_dy
exchange
wethAmount_
scaledDiv
tokenAmount_
scaledMul


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
ConvexStrategyBase
onlyVault
deposit
_deposit
withdraw
_withdraw

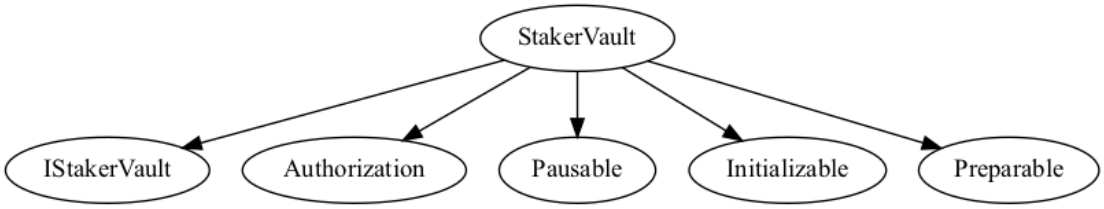

withdrawAll
_roleManager
harvest
_withdrawAll
<Constructor>
shutdown
setCommunityReserve
setCrvCommunityReserveShare
setCvxCommunityReserveShare
_harvest
setImbalanceToleranceIn
ICurveSwapEth
setImbalanceToleranceOut
_validateCurvePool
setStrategist
type
addRewardToken
IERC20
harvestable
IRewardStaking
rewardTokens
_underlyingAmountOut
strategist
getCvxMintAmount
removeRewardToken
balance
name
hasPendingFunds
_stakedBalance
_sendCommunityReserveShare
_lpBalance
_underlyingBalance
crvBalance_
scaledMul
cvxBalance_
scaledMul
IBooster
poolInfo
ISwapperRouter
swapAll
getAmountOut
IAddressProvider
getSwapperRouter
getRoleManager
ICurveRegistry
get_A
i
uncheckedInc
EnumerableSet.AddressSet
at
length
add

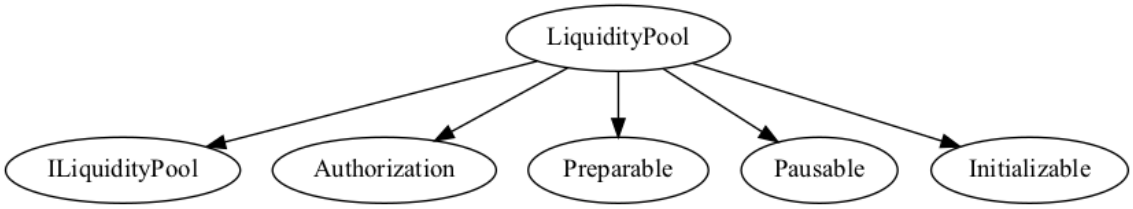

contains
toArray
remove
crvAmount_
scaledMul
IRewardStaking
getReward
earned
balanceOf
rewardToken
extraRewards
IERC20
safeTransfer
balanceOf
safeApprove

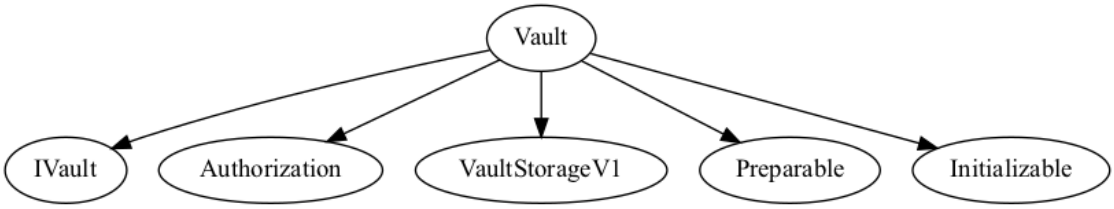

Legend
Internal Call
External Call
Defined Contract
Undefined Contract
TopUpActionLibrary (lib)
lockFunds
calcExchangeAmount
IStakerVault
_approve
type
IERC20
amount
scaledDivRoundUp
IStakerVault
increaseActionLockedBalance
balanceOf
allowance
transferFrom
stake
transfer
decreaseActionLockedBalance
unstake
TopUpAction
executeEstimatedGasUsage
<Receive Ether>
executeActionFee
resetGasUsage
resetActionFee
executeFeeHandler
executeTopUpHandler
getTopUpHandler
getHandler
_executeUInt256
_resetUInt256Config
_executeAddress
_calcExchangeAmount
_lockFunds
_isSwappable
_getHandler
Register
RecordMeta
_getProtocolKey
resetTopUpHandler
prepareTopUpHandler
resetFeeHandler
prepareFeeHandler
prepareEstimatedGasUsage
prepareActionFee
_resetAddressConfig
_prepare
register
initialize
getHealthFactor
_updateTopUpHandler
isUsable

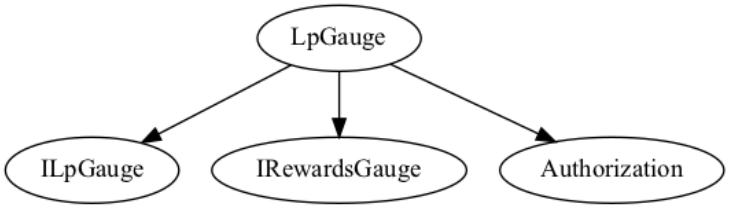

<Constructor>
_setConfig
ITopUpHandler
execute
TopUp
getUserPositions
getActionFee
IActionFeeHandler
getEthRequiredForGas
_payFees
getFeeHandler
ILiquidityPool
exchangeRate
resetPosition
_approve
IERC20
getUnderlying
getSupportedProtocols
type
payable
redeem
addUsableToken
getEstimatedGasUsage
getUsableTokens
IStakerVault
Deregister
usersWithPositions
_removePosition
_removeUserPosition
getPosition
IController
addressProvider
canKeeperExecuteAction
EnumerableSet.Bytes32Set
add
remove
totalLockAmount
toUint128
contains
at
length
IAddressProvider
getPoolForToken
getStakerVault
IGasBank
getGasBank
withdrawUnused
EnumerableSet.AddressSet
balanceOf
contains
withdrawFrom
remove
add
toArray
length
at
i
uncheckedInc

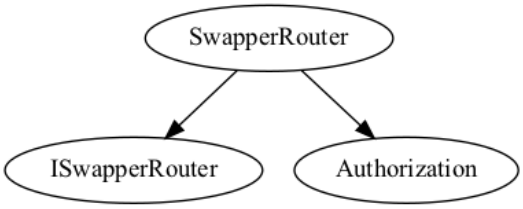
Inheritance

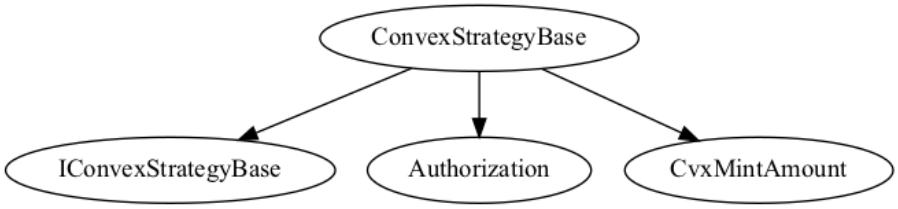


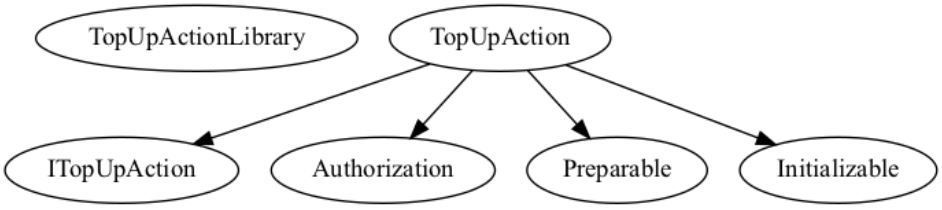












Describe

```
+  StakerVault (IStakerVault, Authorization, Pausable, Initializable, Preparable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] initialize #
      - modifiers: initializer
   - [Ext] initializeLpGauge #
      - modifiers: onlyGovernance
   - [Ext] prepareLpGauge #
      - modifiers: onlyGovernance
   - [Ext] executeLpGauge #
      - modifiers: onlyGovernance
   - [Ext] transfer #
      - modifiers: notPaused
   - [Ext] transferFrom #
      - modifiers: notPaused
   - [Ext] approve #
      - modifiers: notPaused
   - [Ext] increaseActionLockedBalance #
   - [Ext] decreaseActionLockedBalance #
   - [Ext] poolCheckpoint #
   - [Ext] getLpGauge
   - [Ext] getStakedByActions
   - [Ext] allowance
   - [Ext] balanceOf
   - [Ext] getPoolTotalStaked
   - [Ext] stakedAndActionLockedBalanceOf
   - [Ext] actionLockedBalanceOf
   - [Ext] decimals
   - [Ext] getToken
   - [Pub] unstake #
   - [Pub] stake #
   - [Pub] stakeFor #
      - modifiers: notPaused
   - [Pub] unstakeFor #
   - [Int] _isAuthorizedToPause

+  EthPool (LiquidityPool, IEthPool)
   - [Pub] <Constructor> #
      - modifiers: LiquidityPool
   - [Ext] <Fallback> ($)
   - [Ext] initialize #
   - [Pub] getUnderlying
   - [Int] _doTransferIn #
   - [Int] _doTransferOut #
   - [Int] _getBalanceUnderlying
   - [Int] _getBalanceUnderlying


+  Erc20Pool (LiquidityPool, IErc20Pool)
```

```
   - [Pub] <Constructor> #
      - modifiers: LiquidityPool
   - [Pub] initialize #
   - [Pub] getUnderlying
   - [Int] _doTransferIn #
   - [Int] _doTransferOut #
   - [Int] _getBalanceUnderlying
   - [Int] _getBalanceUnderlying

+  PoolFactory (IPoolFactory, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] addPoolImplementation #
      - modifiers: onlyGovernance
   - [Ext] addLpTokenImplementation #
      - modifiers: onlyGovernance
   - [Ext] addVaultImplementation #
      - modifiers: onlyGovernance
   - [Ext] addStakerVaultImplementation #
      - modifiers: onlyGovernance
   - [Ext] deployPool #
      - modifiers: onlyGovernance
   - [Int] _addImplementation #

+  LiquidityPool (ILiquidityPool, Authorization, Preparable, Pausable, Initializable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] deposit ($)
   - [Ext] deposit ($)
   - [Ext] depositAndStake ($)
   - [Ext] withdrawAll #
      - modifiers: onlyGovernance
   - [Ext] setLpToken #
      - modifiers: onlyRoles2
   - [Ext] handleLpTokenTransfer #
   - [Ext] prepareNewRequiredReserves #
      - modifiers: onlyGovernance
   - [Ext] executeNewRequiredReserves #
   - [Ext] resetRequiredReserves #
      - modifiers: onlyGovernance
   - [Ext] prepareNewReserveDeviation #
      - modifiers: onlyGovernance
   - [Ext] executeNewReserveDeviation #
   - [Ext] resetNewReserveDeviation #
      - modifiers: onlyGovernance
   - [Ext] prepareNewMinWithdrawalFee #
      - modifiers: onlyGovernance
   - [Ext] executeNewMinWithdrawalFee #
   - [Ext] resetNewMinWithdrawalFee #
      - modifiers: onlyGovernance

   - [Ext] prepareNewMaxWithdrawalFee #
```

```
    - modifiers: onlyGovernance
- [Ext] executeNewMaxWithdrawalFee #
- [Ext] resetNewMaxWithdrawalFee #
    - modifiers: onlyGovernance
- [Ext] prepareNewWithdrawalFeeDecreasePeriod #
    - modifiers: onlyGovernance
- [Ext] executeNewWithdrawalFeeDecreasePeriod #
- [Ext] resetNewWithdrawalFeeDecreasePeriod #
    - modifiers: onlyGovernance
- [Ext] setStaker #
    - modifiers: onlyRoles2
- [Ext] prepareNewVault #
    - modifiers: onlyGovernance
- [Ext] executeNewVault #
- [Ext] resetNewVault #
    - modifiers: onlyGovernance
- [Ext] redeem #
- [Ext] rebalanceVault #
    - modifiers: onlyGovernance
- [Ext] depositFor ($)
- [Ext] unstakeAndRedeem #
- [Ext] getLpToken
- [Ext] calcRedeem
- [Ext] getUnderlying
- [Pub] depositFor ($)
    - modifiers: notPaused
- [Pub] redeem #
- [Pub] getRequiredReserveRatio
- [Pub] getMaxReserveDeviationRatio
- [Pub] getMinWithdrawalFee
- [Pub] getMaxWithdrawalFee
- [Pub] getWithdrawalFeeDecreasePeriod
- [Pub] getVault
- [Pub] exchangeRate
- [Pub] totalUnderlying
- [Pub] getWithdrawalFee
- [Pub] getNewCurrentFees
- [Int] _rebalanceVault #
- [Int] _initialize #
    - modifiers: initializer
- [Int] _approveStakerVaultSpendingLpTokens #
- [Int] _doTransferIn #
- [Int] _doTransferOut #
- [Int] _rebalanceVault #
- [Int] _updateUserFeesOnDeposit #
- [Int] _getBalanceUnderlying
- [Int] _getBalanceUnderlying
- [Int] _isAuthorizedToPause
- [Int] _getTime
- [Int] _checkFeeInvariants
```

```
+  Erc20Vault (Vault)
   - [Pub] <Constructor> #
      - modifiers: Vault
   - [Ext] initialize #
      - modifiers: initializer
   - [Pub] getUnderlying
   - [Int] _transfer #
   - [Int] _depositToReserve #
   - [Int] _depositToRewardHandler #
   - [Int] _payStrategist #
   - [Int] _availableUnderlying

+  EthVault (Vault)
   - [Pub] <Constructor> #
      - modifiers: Vault
   - [Ext] <Fallback> ($)
   - [Ext] initialize #
      - modifiers: initializer
   - [Pub] getUnderlying
   - [Int] _transfer #
   - [Int] _depositToReserve #
   - [Int] _depositToRewardHandler #
   - [Int] _payStrategist #
   - [Int] _availableUnderlying

+  Vault (IVault, Authorization, VaultStorageV1, Preparable, Initializable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Int] _initialize #
   - [Ext] deposit ($)
      - modifiers: onlyPoolOrMaintenance
   - [Ext] withdraw #
      - modifiers: onlyPoolOrGovernance
   - [Ext] withdrawAll #
      - modifiers: onlyPoolOrGovernance
   - [Ext] withdrawFromReserve #
      - modifiers: onlyGovernance
   - [Ext] activateStrategy #
      - modifiers: onlyGovernance
   - [Ext] deactivateStrategy #
      - modifiers: onlyGovernance
   - [Ext] initializeStrategy #
      - modifiers: onlyGovernance
   - [Ext] prepareNewStrategy #
      - modifiers: onlyGovernance
   - [Ext] executeNewStrategy #
   - [Ext] resetNewStrategy #
      - modifiers: onlyGovernance
   - [Ext] preparePerformanceFee #
      - modifiers: onlyGovernance

   - [Ext] executePerformanceFee #
```

- [Ext] resetPerformanceFee #
  - modifiers: onlyGovernance
- [Ext] prepareStrategistFee #
  - modifiers: onlyGovernance
- [Ext] executeStrategistFee #
- [Ext] resetStrategistFee #
  - modifiers: onlyGovernance
- [Ext] prepareDebtLimit #
  - modifiers: onlyGovernance
- [Ext] executeDebtLimit #
- [Ext] resetDebtLimit #
  - modifiers: onlyGovernance
- [Ext] prepareTargetAllocation #
  - modifiers: onlyGovernance
- [Ext] executeTargetAllocation #
- [Ext] resetTargetAllocation #
  - modifiers: onlyGovernance
- [Ext] prepareReserveFee #
  - modifiers: onlyGovernance
- [Ext] executeReserveFee #
- [Ext] resetReserveFee #
  - modifiers: onlyGovernance
- [Ext] prepareBound #
  - modifiers: onlyGovernance
- [Ext] executeBound #
- [Ext] resetBound #
  - modifiers: onlyGovernance
- [Ext] withdrawFromStrategy #
  - modifiers: onlyGovernance
- [Ext] withdrawFromStrategyWaitingForRemoval #
- [Ext] getStrategiesWaitingForRemoval
- [Ext] getTotalUnderlying
- [Ext] getAllocatedToStrategyWaitingForRemoval
- [Pub] withdrawAllFromStrategy #
  - modifiers: onlyPoolOrGovernance
- [Pub] harvest #
  - modifiers: onlyPoolOrMaintenance
- [Pub] getStrategistFee
- [Pub] getStrategy
- [Pub] getReserveFee
- [Pub] getPerformanceFee
- [Pub] getBound
- [Pub] getTargetAllocation
- [Pub] getDebtLimit
- [Pub] getUnderlying
- [Int] _activateStrategy #
- [Int] _harvest #
- [Int] _withdrawAllFromStrategy #
- [Int] _handleExcessDebt #
- [Int] _handleExcessDebt #

- [Int] _deposit #

```
   - [Int] _shareProfit #
   - [Int] _shareFees #
   - [Int] _emergencyStop #
   - [Int] _deactivateStrategy #
   - [Int] _payStrategist #
   - [Int] _payStrategist #
   - [Int] _transfer #
   - [Int] _depositToReserve #
   - [Int] _depositToRewardHandler #
   - [Int] _availableUnderlying
   - [Int] _computeNewAllocated
   - [Int] _checkFeesInvariant
   - [Prv] _rebalance #

+  VaultReserve (IVaultReserve, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] deposit ($)
      - modifiers: onlyVault
   - [Ext] withdraw #
      - modifiers: onlyVault
   - [Pub] getBalance
   - [Pub] canWithdraw

+  VaultStorage

+  VaultStorageV1 (VaultStorage)

+  ChainlinkOracleProvider (IChainlinkOracleProvider, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] setStalePriceDelay #
      - modifiers: onlyGovernance
   - [Ext] getPriceETH
   - [Pub] getPriceUSD
   - [Int] _getPrice

+  SwapperRouter (ISwapperRouter, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] <Fallback> ($)
   - [Ext] swapAll ($)
   - [Ext] setSlippageTolerance #
      - modifiers: onlyGovernance
   - [Ext] setCurvePool #
      - modifiers: onlyGovernance
   - [Ext] getAmountOut
   - [Pub] swap ($)
   - [Int] _swapForWeth #
   - [Int] _swapWethForToken #

   - [Int] _swap #
```

```
   - [Int] _approve #
   - [Int] _returnTokens #
   - [Int] _getWethOut
   - [Int] _getTokenOut
   - [Int] _getBestDex
   - [Int] _tokenAmountOut
   - [Int] _getAmountOutMin
   - [Int] _minTokenAmountOut
   - [Int] _minWethAmountOut
   - [Int] _getPriceInEth
   - [Int] _getIndices

+  RoleManager (IRoleManager)
   - [Pub] <Constructor> #
   - [Ext] grantRole #
      - modifiers: onlyGovernance
   - [Ext] addGovernor #
      - modifiers: onlyGovernance
   - [Ext] renounceGovernance #
      - modifiers: onlyGovernance
   - [Ext] addGaugeZap #
      - modifiers: onlyGovernance
   - [Ext] removeGaugeZap #
      - modifiers: onlyGovernance
   - [Ext] hasAnyRole
   - [Ext] hasAnyRole
   - [Ext] hasAnyRole
   - [Ext] getRoleMember
   - [Pub] revokeRole #
      - modifiers: onlyGovernance
   - [Pub] getRoleMemberCount
   - [Pub] hasRole
   - [Int] _grantRole #
   - [Int] _revokeRole #

+  AuthorizationBase
   - [Ext] roleManager
   - [Int] _roleManager

+  Authorization (AuthorizationBase)
   - [Pub] <Constructor> #
   - [Int] _roleManager

+  LpGauge (ILpGauge, IRewardsGauge, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] poolCheckpoint #
   - [Ext] claimRewards #
   - [Ext] claimableRewards
   - [Pub] userCheckpoint #

   - [Int] _mintRewards #
```

```
   - [Int] _poolCheckpoint #

+  LpToken (ILpToken, ERC20Upgradeable)
   - [Pub] <Constructor> #
      - modifiers: ERC20Upgradeable
   - [Ext] initialize #
      - modifiers: initializer
   - [Ext] mint #
      - modifiers: onlyMinter
   - [Ext] burn #
   - [Ext] burn #
      - modifiers: onlyMinter
   - [Pub] decimals
   - [Int] _beforeTokenTransfer #

+  BkdToken (IBkdToken, ERC20)
   - [Pub] <Constructor> #
      - modifiers: ERC20
   - [Ext] mint #
   - [Ext] cap

+  ConvexStrategyBase (IConvexStrategyBase, Authorization, CvxMintAmount)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] deposit ($)
      - modifiers: onlyVault
   - [Ext] withdraw #
      - modifiers: onlyVault
   - [Ext] withdrawAll #
   - [Ext] harvest #
      - modifiers: onlyVault
   - [Ext] shutdown #
      - modifiers: onlyVault
   - [Ext] setCommunityReserve #
      - modifiers: onlyGovernance
   - [Ext] setCrvCommunityReserveShare #
      - modifiers: onlyGovernance
   - [Ext] setCvxCommunityReserveShare #
      - modifiers: onlyGovernance
   - [Ext] setImbalanceToleranceIn #
      - modifiers: onlyGovernance
   - [Ext] setImbalanceToleranceOut #
      - modifiers: onlyGovernance
   - [Ext] setStrategist #
   - [Ext] addRewardToken #
      - modifiers: onlyGovernance
   - [Ext] removeRewardToken #
      - modifiers: onlyGovernance
   - [Ext] harvestable
   - [Ext] strategist

   - [Ext] rewardTokens
```

```
    - [Ext] balance
    - [Ext] name
    - [Ext] hasPendingFunds
    - [Int] _deposit #
    - [Int] _withdraw #
    - [Int] _withdrawAll #
    - [Int] _harvest #
    - [Int] _sendCommunityReserveShare #
    - [Int] _underlyingBalance
    - [Int] _lpBalance
    - [Int] _stakedBalance
    - [Int] _underlyingAmountOut
    - [Int] _validateCurvePool

+   BkdEthCvx (ConvexStrategyBase)
    - [Pub] <Constructor> #
       - modifiers: ConvexStrategyBase
    - [Ext] <Fallback> ($)
    - [Ext] name
    - [Pub] balance
    - [Int] _deposit #
    - [Int] _withdraw #
    - [Int] _withdrawAll #
    - [Int] _underlyingBalance
    - [Int] _minLpAccepted
    - [Int] _maxLpBurned
    - [Int] _minUnderlyingAccepted
    - [Int] _underlyingToLp
    - [Int] _lpToUnderlying

+   BkdTriHopCvx (ConvexStrategyBase, IBkdTriHopCvx)
    - [Pub] <Constructor> #
       - modifiers: ConvexStrategyBase
    - [Ext] setHopImbalanceToleranceIn #
       - modifiers: onlyGovernance
    - [Ext] setHopImbalanceToleranceOut #
       - modifiers: onlyGovernance
    - [Ext] changeConvexPool #
       - modifiers: onlyGovernance
    - [Pub] balance
    - [Pub] name
    - [Int] _deposit #
    - [Int] _withdraw #
    - [Int] _withdrawAll #
    - [Int] _underlyingBalance
    - [Int] _hopLpBalance
    - [Int] _minLpAccepted
    - [Int] _maxLpBurned
    - [Int] _minHopLpAcceptedFromWithdraw
    - [Int] _minHopLpAcceptedFromDeposit

    - [Int] _maxHopLpBurned
```

```
   - [Int] _minUnderlyingAccepted
   - [Int] _underlyingToHopLp
   - [Int] _hopLpToUnderlying
   - [Int] _lpToHopLp
   - [Int] _hopLpToLp
   - [Prv] _withdrawAllToHopLp #

+  Controller (IController, Authorization, Preparable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] setInflationManager #
      - modifiers: onlyGovernance
   - [Ext] addStakerVault #
      - modifiers: onlyRoles2
   - [Ext] removePool #
      - modifiers: onlyGovernance
   - [Ext] prepareKeeperRequiredStakedBKD #
      - modifiers: onlyGovernance
   - [Ext] resetKeeperRequiredStakedBKD #
      - modifiers: onlyGovernance
   - [Ext] executeKeeperRequiredStakedBKD #
   - [Ext] canKeeperExecuteAction
   - [Ext] getTotalEthRequiredForGas
   - [Pub] getKeeperRequiredStakedBKD

+  PoolMigrationZap (IPoolMigrationZap)
   - [Pub] <Constructor> #
   - [Ext] <Fallback> ($)
   - [Ext] migrateAll #
   - [Pub] migrate #

+  Pausable
   - [Ext] pause #
      - modifiers: onlyAuthorizedToPause
   - [Ext] unpause #
      - modifiers: onlyAuthorizedToPause
   - [Int] _isAuthorizedToPause

+  Preparable (IPreparable)
   - [Int] _prepareDeadline #
   - [Int] _prepare #
   - [Int] _prepare #
   - [Int] _prepare #
   - [Int] _prepare #
   - [Int] _resetUInt256Config #
   - [Int] _resetAddressConfig #
   - [Int] _executeDeadline #
   - [Int] _executeUInt256 #
   - [Int] _executeAddress #
   - [Int] _setConfig #

   - [Int] _setConfig #
```

```
+  CvxMintAmount
   - [Pub] getCvxMintAmount

+ [Int] IPausable
   - [Ext] pause #
   - [Ext] unpause #
   - [Ext] isPaused
   - [Ext] isAuthorizedToPause

+  BkdLocker (IBkdLocker, Authorization, Preparable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] initialize #
      - modifiers: onlyGovernance
   - [Ext] migrate #
      - modifiers: onlyGovernance
   - [Ext] lock #
   - [Ext] depositFees #
   - [Ext] claimFees #
   - [Ext] userCheckpoint #
   - [Ext] prepareUnlock #
   - [Ext] executeUnlocks #
   - [Ext] getUserShare
   - [Ext] boostedBalance
   - [Ext] balanceOf
   - [Ext] getShareOfTotalBoostedBalance
   - [Ext] getStashedGovTokens
   - [Ext] claimableFees
   - [Pub] claimFees #
   - [Pub] lockFor #
   - [Pub] getUserShare
   - [Pub] claimableFees
   - [Pub] computeNewBoost
   - [Int] _userCheckpoint #

+  AddressProvider (IAddressProvider, AuthorizationBase, Initializable, Preparable)
   - [Pub] <Constructor> #
   - [Ext] initialize #
      - modifiers: initializer
   - [Ext] getKnownAddressKeys
   - [Ext] addFeeHandler #
      - modifiers: onlyGovernance
   - [Ext] removeFeeHandler #
      - modifiers: onlyGovernance
   - [Ext] addAction #
      - modifiers: onlyGovernance
   - [Ext] addPool #
      - modifiers: onlyRoles2
   - [Ext] removePool #

      - modifiers: onlyRole
```

```
    - [Ext] allVaults
    - [Ext] getVaultAtIndex
    - [Ext] vaultsCount
    - [Ext] isVault
    - [Ext] updateVault #
       - modifiers: onlyRole
    - [Pub] getAddress
    - [Pub] getAddress
    - [Pub] getAddressMeta
    - [Ext] initializeAddress #
    - [Pub] initializeAddress #
       - modifiers: onlyGovernance
    - [Ext] initializeAndFreezeAddress #
       - modifiers: onlyGovernance
    - [Ext] freezeAddress #
       - modifiers: onlyGovernance
    - [Ext] prepareAddress #
       - modifiers: onlyGovernance
    - [Ext] executeAddress #
    - [Ext] resetAddress #
       - modifiers: onlyGovernance
    - [Ext] addStakerVault #
       - modifiers: onlyRole
    - [Ext] isWhiteListedFeeHandler
    - [Ext] safeGetPoolForToken
    - [Ext] getPoolForToken
    - [Ext] allActions
    - [Ext] isAction
    - [Ext] isPool
    - [Ext] allPools
    - [Ext] getPoolAtIndex
    - [Ext] poolsCount
    - [Ext] allStakerVaults
    - [Ext] getStakerVault
    - [Ext] tryGetStakerVault
    - [Ext] isStakerVaultRegistered
    - [Pub] isStakerVault
    - [Int] _roleManager
    - [Int] _initializeAddress #
    - [Int] _addKnownAddressKey #

+   CvxCrvRewardsLocker (ICvxCrvRewardsLocker, Authorization)
    - [Pub] <Constructor> #
       - modifiers: Authorization
    - [Ext] lockCvx #
    - [Ext] lockCrv #
    - [Ext] setSpendRatio #
       - modifiers: onlyGovernance
    - [Ext] claimRewards #
    - [Ext] stakeCvxCrv #

    - [Ext] setWithdrawalFlag #
```

```
      - modifiers: onlyGovernance
   - [Ext] resetWithdrawalFlag #
      - modifiers: onlyGovernance
   - [Ext] processExpiredLocks #
   - [Ext] setTreasury #
      - modifiers: onlyGovernance
   - [Ext] withdraw #
      - modifiers: onlyGovernance
   - [Ext] withdrawCvxCrv #
      - modifiers: onlyGovernance
   - [Ext] unstakeCvxCrv #
      - modifiers: onlyGovernance
   - [Ext] unstakeCvxCrv #
      - modifiers: onlyGovernance
   - [Ext] setDelegate #
      - modifiers: onlyGovernance
   - [Ext] clearDelegate #
      - modifiers: onlyGovernance
   - [Ext] forfeitRewards #
      - modifiers: onlyGovernance
   - [Pub] lockRewards #
   - [Pub] withdraw #
      - modifiers: onlyGovernance
   - [Pub] unstakeCvxCrv #
      - modifiers: onlyGovernance
   - [Int] _lockCrv #
   - [Int] _lockCvx #
   - [Int] _stakeCvxCrv #
   - [Int] _unstakeCvxCrv #

+  TopUpKeeperHelper (ITopUpKeeperHelper)
   - [Pub] <Constructor> #
   - [Ext] getExecutableTopups
   - [Ext] batchCanExecute
   - [Pub] listPositions
   - [Pub] canExecute
   - [Prv] _canExecute
   - [Prv] _positionToTopup
   - [Prv] _shortenTopups

+  TopUpActionFeeHandler (IActionFeeHandler, Authorization, Preparable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] setInitialKeeperGaugeForToken #
      - modifiers: onlyGovernance
   - [Ext] payFees #
   - [Ext] claimKeeperFeesForPool #
   - [Ext] claimTreasuryFees #
   - [Ext] prepareKeeperFee #
      - modifiers: onlyGovernance

   - [Ext] executeKeeperFee #
```

```
   - [Ext] resetKeeperFee #
      - modifiers: onlyGovernance
   - [Ext] prepareKeeperGauge #
      - modifiers: onlyGovernance
   - [Ext] executeKeeperGauge #
   - [Ext] resetKeeperGauge #
      - modifiers: onlyGovernance
   - [Ext] prepareTreasuryFee #
      - modifiers: onlyGovernance
   - [Ext] executeTreasuryFee #
   - [Ext] resetTreasuryFee #
      - modifiers: onlyGovernance
   - [Pub] getKeeperFeeFraction
   - [Pub] getKeeperGauge
   - [Pub] getTreasuryFeeFraction
   - [Int] _getKeeperGaugeKey

+ [Lib] TopUpActionLibrary
   - [Ext] lockFunds #
   - [Ext] calcExchangeAmount
   - [Prv] _approve #

+  TopUpAction (ITopUpAction, Authorization, Preparable, Initializable)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] <Fallback> ($)
   - [Ext] initialize #
      - modifiers: initializer,onlyGovernance
   - [Ext] register ($)
   - [Ext] execute #
   - [Ext] resetPosition #
   - [Ext] executeTopUpHandler #
   - [Ext] resetTopUpHandler #
      - modifiers: onlyGovernance
   - [Ext] prepareActionFee #
      - modifiers: onlyGovernance
   - [Ext] executeActionFee #
   - [Ext] resetActionFee #
      - modifiers: onlyGovernance
   - [Ext] prepareFeeHandler #
      - modifiers: onlyGovernance
   - [Ext] executeFeeHandler #
   - [Ext] resetFeeHandler #
      - modifiers: onlyGovernance
   - [Ext] prepareEstimatedGasUsage #
      - modifiers: onlyGovernance
   - [Ext] executeEstimatedGasUsage #
   - [Ext] resetGasUsage #
      - modifiers: onlyGovernance
   - [Ext] addUsableToken #

      - modifiers: onlyGovernance
```

```
   - [Ext] getEthRequiredForGas
   - [Ext] getUserPositions
   - [Ext] getSupportedProtocols
   - [Ext] usersWithPositions
   - [Ext] getUsableTokens
   - [Ext] getTopUpHandler
   - [Pub] execute #
   - [Pub] prepareTopUpHandler #
      - modifiers: onlyGovernance
   - [Pub] getHealthFactor
   - [Pub] getHandler
   - [Pub] getEstimatedGasUsage
   - [Pub] getActionFee
   - [Pub] getFeeHandler
   - [Pub] getPosition
   - [Pub] isUsable
   - [Int] _updateTopUpHandler #
   - [Int] _payFees #
   - [Int] _lockFunds #
   - [Int] _removePosition #
   - [Int] _removeUserPosition #
   - [Int] _approve #
   - [Int] _calcExchangeAmount
   - [Int] _getHandler
   - [Int] _isSwappable
   - [Int] _getProtocolKey

+  CTokenRegistry (ICTokenRegistry)
   - [Pub] <Constructor> #
   - [Ext] fetchCToken #
   - [Ext] getCToken
   - [Pub] getCToken
   - [Int] _updateCTokenMapping #
   - [Int] _isCTokenUsable

+  AaveHandler (ITopUpHandler)
   - [Pub] <Constructor> #
   - [Ext] topUp ($)
   - [Ext] getUserFactor
   - [Int] _approve #

+  CompoundHandler (ITopUpHandler, ExponentialNoError)
   - [Pub] <Constructor> #
   - [Ext] topUp ($)
   - [Ext] getUserFactor
   - [Int] _repayAnyDebt #
   - [Int] _approve #
   - [Int] _getAccountBorrowsAndSupply

+  GasBank (IGasBank)

   - [Pub] <Constructor> #
```

```
   - [Ext] depositFor ($)
   - [Ext] withdrawFrom #
   - [Ext] withdrawUnused #
   - [Ext] balanceOf
   - [Pub] withdrawFrom #
   - [Int] _withdrawFrom #

+  VestedEscrowRevocable (IVestedEscrowRevocable, VestedEscrow)
   - [Pub] <Constructor> #
      - modifiers: VestedEscrow
   - [Ext] claim #
   - [Ext] revoke #
   - [Ext] vestedOf
   - [Ext] balanceOf
   - [Ext] lockedOf
   - [Pub] claim #
      - modifiers: nonReentrant

+  AmmConvexGauge (IAmmConvexGauge, AmmGauge, CvxMintAmount)
   - [Pub] <Constructor> #
      - modifiers: AmmGauge
   - [Ext] claimRewards #
   - [Ext] setInflationRecipient #
      - modifiers: onlyGovernance
   - [Ext] deactivateInflationRecipient #
      - modifiers: onlyGovernance
   - [Ext] claimableRewards
   - [Ext] allClaimableRewards
   - [Pub] stakeFor #
   - [Pub] unstakeFor #
   - [Pub] poolCheckpoint #
   - [Int] _userCheckpoint #

+  FeeBurner (IFeeBurner)
   - [Pub] <Constructor> #
   - [Ext] <Fallback> ($)
   - [Pub] burnToTarget ($)
   - [Int] _depositInPool #
   - [Int] _approve #
   - [Int] _swapperRouter

+  EscrowTokenHolder
   - [Pub] <Constructor> #

+  VestedEscrow (IVestedEscrow, ReentrancyGuard)
   - [Pub] <Constructor> #
   - [Ext] setAdmin #
   - [Ext] setFundAdmin #
   - [Ext] initializeUnallocatedSupply #
   - [Ext] fund #

      - modifiers: nonReentrant
```

```
   - [Ext] claim #
   - [Ext] vestedSupply
   - [Ext] lockedSupply
   - [Ext] vestedOf
   - [Ext] balanceOf
   - [Ext] lockedOf
   - [Pub] claim #
      - modifiers: nonReentrant
   - [Int] _claimUntil #
   - [Int] _computeVestedAmount
   - [Int] _totalVestedOf
   - [Int] _totalVested
   - [Int] _balanceOf

+  Minter (IMinter, Authorization, ReentrancyGuard)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] setToken #
      - modifiers: onlyGovernance
   - [Ext] startInflation #
      - modifiers: onlyGovernance
   - [Ext] executeInflationRateUpdate #
   - [Ext] mint #
      - modifiers: nonReentrant
   - [Ext] mintNonInflationTokens #
      - modifiers: onlyGovernance
   - [Ext] getLpInflationRate
   - [Ext] getKeeperInflationRate
   - [Ext] getAmmInflationRate
   - [Int] _executeInflationRateUpdate #
   - [Int] _mint #

+  KeeperGauge (IKeeperGauge, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] kill #
      - modifiers: onlyInflationManager
   - [Ext] reportFees #
   - [Ext] advanceEpoch #
      - modifiers: onlyInflationManager
   - [Ext] claimRewards #
   - [Ext] claimableRewards
   - [Pub] poolCheckpoint #
   - [Pub] claimRewards #
   - [Int] _mintRewards #
   - [Int] _calcTotalClaimable

+  InflationManager (Authorization, IInflationManager, Preparable)
   - [Pub] <Constructor> #
      - modifiers: Authorization

   - [Ext] setMinter #
```

```
    - modifiers: onlyGovernance
- [Ext] advanceKeeperGaugeEpoch #
    - modifiers: onlyGovernance
- [Ext] mintRewards #
    - modifiers: onlyGauge
- [Ext] deactivateWeightBasedKeeperDistribution #
    - modifiers: onlyGovernance
- [Ext] checkpointAllGauges #
- [Ext] prepareKeeperPoolWeight #
    - modifiers: onlyGovernance
- [Ext] executeKeeperPoolWeight #
- [Ext] batchPrepareKeeperPoolWeights #
    - modifiers: onlyGovernance
- [Ext] whitelistGauge #
    - modifiers: onlyRole
- [Ext] batchExecuteKeeperPoolWeights #
- [Ext] removeStakerVaultFromInflation #
    - modifiers: onlyRole
- [Ext] prepareLpPoolWeight #
    - modifiers: onlyRoles2
- [Ext] executeLpPoolWeight #
- [Ext] batchPrepareLpPoolWeights #
    - modifiers: onlyRoles2
- [Ext] batchExecuteLpPoolWeights #
- [Ext] prepareAmmTokenWeight #
    - modifiers: onlyRoles2
- [Ext] executeAmmTokenWeight #
- [Ext] batchPrepareAmmTokenWeights #
    - modifiers: onlyRoles2
- [Ext] batchExecuteAmmTokenWeights #
- [Ext] setKeeperGauge #
    - modifiers: onlyGovernance
- [Ext] removeKeeperGauge #
    - modifiers: onlyGovernance
- [Ext] setAmmGauge #
    - modifiers: onlyGovernance
- [Ext] removeAmmGauge #
    - modifiers: onlyGovernance
- [Ext] addGaugeForVault #
- [Ext] getAllAmmGauges
- [Ext] getLpRateForStakerVault
- [Ext] getKeeperRateForPool
- [Ext] getAmmRateForToken
- [Ext] getKeeperWeightForPool
- [Ext] getAmmWeightForToken
- [Ext] getLpPoolWeight
- [Ext] getKeeperGaugeForPool
- [Ext] getAmmGaugeForToken
- [Pub] isInflationWeightManager
- [Int] _executeKeeperPoolWeight #

- [Int] _executeLpPoolWeight #
```

```
   - [Int] _executeAmmTokenWeight #
   - [Int] _removeKeeperGauge #
   - [Int] _ensurePoolExists
   - [Int] _getKeeperGaugeKey
   - [Int] _getAmmGaugeKey
   - [Int] _getLpStakerVaultKey

+  AmmGauge (Authorization, IAmmGauge)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] kill #
   - [Ext] claimRewards #
   - [Ext] stake #
   - [Ext] unstake #
   - [Ext] getAmmToken
   - [Ext] isAmmToken
   - [Ext] claimableRewards
   - [Pub] stakeFor #
   - [Pub] unstakeFor #
   - [Pub] poolCheckpoint #
   - [Int] _userCheckpoint #

+  RewardHandler (IRewardHandler, Preparable, Authorization)
   - [Pub] <Constructor> #
      - modifiers: Authorization
   - [Ext] <Fallback> ($)
   - [Ext] burnFees #
   - [Int] _approve #


($) = payable function
# = non-constant function
```

License

This report falls under the terms described in the included LICENSE.