mevETH Contract

MevEth.sol Git Source(opens in a new tab)

Contract that allows deposit of ETH, for a Liquid Staking Receipt (LSR) in return.

LSR is represented through an ERC4626 token and interface.

State Variables

stakingPaused

Inidicates if staking is paused.

bool

public stakingPaused;

initialized

Indicates if contract is initialized.

bool

public initialized;

feeDenominator

withdraw fee denominator

uint16

internal

constant feeDenominator =

10_000 ;

pendingStakingModuleCommittedTimestamp

Timestamp when pending staking module update can be finalized.

uint64

public pendingStakingModuleCommittedTimestamp;

pendingMevEthShareVaultCommittedTimestamp

Timestamp when pending mevEthShareVault update can be finalized.

uint64

public pendingMevEthShareVaultCommittedTimestamp;

MODULE_UPDATE_TIME_DELAY

Time delay before staking module or share vault can be finalized.

uint64

internal

constant MODULE_UPDATE_TIME_DELAY =

7

days ;

MAX_DEPOSIT

Max amount of ETH that can be deposited.

uint128

internal

constant MAX_DEPOSIT = type(uint128).max;

MIN_DEPOSIT

Min amount of ETH that can be deposited.

uint128

public

constant MIN_DEPOSIT =

0.01

ether ;

MIN_WITHDRAWAL

Min amount of ETH that can be withdrawn via the queue.

uint128

public MIN_WITHDRAWAL;

mevEthShareVault

The address of the MevEthShareVault.

address

public mevEthShareVault;

pendingMevEthShareVault

The address of the pending MevEthShareVault when a new vault has been committed but not finalized.

address

public pendingMevEthShareVault;

stakingModule

The staking module used to stake Ether.

IStakingModule public stakingModule;

pendingStakingModule

The pending staking module when a new module has been committed but not finalized.

IStakingModule public pendingStakingModule;

WETH9

WETH Implementation used by MevEth.

WETH public

immutable WETH9;

lastRewards

Last rewards payment by block number

uint256

internal lastRewards;

fraction

Struct used to accounting the ETH staked within MevEth.

Fraction public fraction;

CREAM_TO_MEV_ETH_PERCENT

The percent out of 1000 crETH2 can be redeemed for as mevEth

uint256

public

constant CREAM_TO_MEV_ETH_PERCENT =

1130 ;

creamToken

The canonical address of the crETH2 address

address

public

constant creamToken =

0x49D72e3973900A195A155a46441F0C08179FdB64 ;

lastDeposit

Sandwich protection mapping of last user deposits by block number

mapping (address

=>

uint256) lastDeposit;

depositedValidators

Deposited validators mapping to prevent double deposits

mapping (bytes

=>

bool) depositedValidators;

queueLength

The length of the withdrawal queue.

uint256

public queueLength;

requestsFinalisedUntil

mark the latest withdrawal request that was finalised

uint256

public requestsFinalisedUntil;

withdrawalAmountQueued

Withdrawal amount queued

uint256

public withdrawalAmountQueued;

withdrawalQueue

The mapping representing the withdrawal queue.

The index in the queue is the key, and the value is the WithdrawalTicket.

mapping (uint256 ticketNumber => WithdrawalTicket ticket) public withdrawalQueue;

Functions

constructor

Construction creates mevETH token, sets authority and weth address.

Pending staking module and committed timestamp will both be zero on deployment.

constructor (address authority ,

address weth) Auth (authority) ERC20 ("Mev Liquid Staking Receipt", "mevETH", 18); Parameters

Name Type Description authority address Address of the controlling admin authority. weth address Address of the WETH contract to use for deposits.

calculateNeededEtherBuffer

Calculate the needed Ether buffer required when creating a new validator.

function

calculateNeededEtherBuffer () public

view

returns (uint256); Returns

Name Type Description uint256 uint256 The required Ether buffer.

init

Initializes the MevEth contract, setting the staking module and share vault addresses.

This function can only be called once and is protected by the onlyAdmin modifier.

function

init (address initialShareVault ,

address initialStakingModule) external

onlyAdmin ; Parameters

Name Type Description initialShareVault address The initial share vault set during initialization. initialStakingModule address The initial staking module set during initialization.

_stakingUnpaused

Ensures that staking is not paused when invoking a specific function.

This check is used on the createValidator, deposit and mint functions.

function

_stakingUnpaused () internal

view ;

pauseStaking

Pauses staking on the MevEth contract.

This function is only callable by addresses with the admin role.

function

pauseStaking () external

onlyAdmin ;

unpauseStaking

Unauses staking on the MevEth contract.

This function is only callable by addresses with the admin role.

function

unpauseStaking () external

onlyAdmin ;

commitUpdateStakingModule

Starts the process to update the staking module. To finalize the update, the MODULE_UPDATE_TIME_DELAY must elapse and thefinalizeUpdateStakingModule function must be called.

This function is only callable by addresses with the admin role.

function

commitUpdateStakingModule (IStakingModule newModule) external

onlyAdmin ; Parameters

Name Type Description newModule IStakingModule The new staking module.

finalizeUpdateStakingModule

Finalizes the staking module update if a pending staking module exists.

This function is only callable by addresses with the admin role.

function

finalizeUpdateStakingModule () external

onlyAdmin ;

cancelUpdateStakingModule

Cancels a pending staking module update.

This function is only callable by addresses with the admin role.

function

cancelUpdateStakingModule () external

onlyAdmin ;

commitUpdateMevEthShareVault

Starts the process to update the share vault. To finalize the update, the MODULE_UPDATE_TIME_DELAY must elapse and the finalizeUpdateStakingModule function must be called.

This function is only callable by addresses with the admin role

function

commitUpdateMevEthShareVault (address newMevEthShareVault) external

onlyAdmin ; Parameters

Name Type Description newMevEthShareVault address The new share vault

finalizeUpdateMevEthShareVault

Finalizes the share vault update if a pending share vault exists.

This function is only callable by addresses with the admin role.

function

finalizeUpdateMevEthShareVault () external

onlyAdmin ;

cancelUpdateMevEthShareVault

Cancels a pending share vault update.

This function is only callable by addresses with the admin role.

function

cancelUpdateMevEthShareVault () external

onlyAdmin ;

createValidator

This function passes through the needed Ether to the Staking module, and the assosiated credentials with it

This function is only callable by addresses with the operator role and if staking is unpaused

function

createValidator (IStakingModule . ValidatorData

calldata newData ,

bytes32 latestDepositRoot) external

onlyOperator ; Parameters

Name Type Description newData IStakingModule.ValidatorData The data needed to create a new validator
latestDepositRoot bytes32

grantRewards

Grants rewards updating the fraction.elastic.

called from validator rewards updates

function

grantRewards () external

payable ;

grantValidatorWithdraw

Allows the MevEthShareVault or the staking module to withdraw validator funds from the contract.

Before updating the fraction, the withdrawal queue is processed, which pays out any pending withdrawals.

This function is only callable by the MevEthShareVault or the staking module.

function

grantValidatorWithdraw () external

payable ;

claim

Claim Finalised Withdrawal Ticket

function

claim (uint256 withdrawalId) external ; Parameters

Name Type Description withdrawalId uint256 Unique ID of the withdrawal ticket

processWithdrawalQueue

Processes the withdrawal queue, reserving any pending withdrawals with the contract's available balance.

function

processWithdrawalQueue (uint256 newRequestsFinalisedUntil) external

onlyOperator ;

setMinWithdrawal

function

setMinWithdrawal (uint128 newMinimum) public

onlyAdmin ;

asset

The underlying asset of the mevEth contract

function

asset () external

view

returns (address assetTokenAddress); Returns

Name Type Description assetTokenAddress address The address of the asset token

totalAssets

The total amount of assets controlled by the mevEth contract

function

totalAssets () external

view

returns (uint256 totalManagedAssets); Returns

Name Type Description totalManagedAssets uint256 The amount of eth controlled by the mevEth contract

convertToShares

Function to convert a specified amount of assets to shares based on the elastic and base.

function

convertToShares (uint256 assets) public

view

returns (uint256 shares); Parameters

Name Type Description assets uint256 The amount of assets to convert to shares Returns

Name Type Description shares uint256 The value of the given assets in shares

convertToAssets

Function to convert a specified amount of shares to assets based on the elastic and base.

function

convertToAssets (uint256 shares) public

view

returns (uint256 assets); Parameters

Name Type Description shares uint256 The amount of shares to convert to assets Returns

Name Type Description assets uint256 The value of the given shares in assets

maxDeposit

Function to indicate the maximum deposit possible.

function

maxDeposit (address) external

view

returns (uint256 maxAssets); Returns

Name Type Description maxAssets uint256 The maximum amount of assets that can be deposited.

previewDeposit

Function to simulate the amount of shares that would be minted for a given deposit at the current ratio.

function

previewDeposit (uint256 assets) external

view

returns (uint256 shares); Parameters

Name Type Description assets uint256 The amount of assets that would be deposited Returns

Name Type Description shares uint256 The amount of shares that would be minted,under ideal conditions only

_deposit

internal deposit function to process Weth or Eth deposits

function

_deposit (address receiver ,

uint256 assets ,

uint256 shares) internal ; Parameters

Name Type Description receiver address The address user whom should receive the mevEth out assets uint256 The amount of assets to deposit shares uint256 The amount of shares that should be minted

deposit

Function to deposit assets into the mevEth contract

function

deposit (uint256 assets ,

address receiver) external

payable

returns (uint256 shares); Parameters

Name Type Description assets uint256 The amount of WETH which should be deposited receiver address The address user whom should receive the mevEth out Returns

Name Type Description shares uint256 The amount of shares minted

maxMint

Function to indicate the maximum amount of shares that can be minted at the current ratio.

function

maxMint (address) external

view

returns (uint256 maxShares); Returns

Name Type Description maxShares uint256 The maximum amount of shares that can be minted

previewMint

Function to simulate the amount of assets that would be required to mint a given amount of shares at the current ratio.

function

previewMint (uint256 shares) external

view

returns (uint256 assets); Parameters

Name Type Description shares uint256 The amount of shares that would be minted Returns

Name Type Description assets uint256 The amount of assets that would be required,under ideal conditions only

mint

Function to mint shares of the mevEth contract

function

mint (uint256 shares ,

address receiver) external

payable

returns (uint256 assets); Parameters

Name Type Description shares uint256 The amount of shares that should be minted receiver address The address user whom should receive the mevEth out Returns

Name Type Description assets uint256 The amount of assets deposited

maxWithdraw

Function to indicate the maximum amount of assets that can be withdrawn at the current state.

function

maxWithdraw (address owner) external

view

returns (uint256 maxAssets); Parameters

Name Type Description owner address The address in question of who would be withdrawing Returns

Name Type Description maxAssets uint256 The maximum amount of assets that can be withdrawn

previewWithdraw

Function to simulate the amount of shares that would be allocated for a specified amount of assets.

function

previewWithdraw (uint256 assets) external

view

returns (uint256 shares); Parameters

Name Type Description assets uint256 The amount of assets that would be withdrawn Returns

Name Type Description shares uint256 The amount of shares that would be burned,under ideal conditions only

_withdraw

Function to withdraw assets from the mevEth contract

function

_withdraw (bool useQueue ,

address receiver ,

address owner ,

uint256 assets ,

uint256 shares) internal ; Parameters

Name Type Description useQueue bool Flag whether to use the withdrawal queue receiver address The address user whom should receive the mevEth out owner address The address of the owner of the mevEth assets uint256 The amount of assets that should be withdrawn shares uint256 shares that will be burned

_updateAllowance

internal function to update allowance for withdraws if necessary

function

_updateAllowance (address owner ,

uint256 shares) internal ; Parameters

Name Type Description owner address owner of tokens shares uint256 amount of shares to update

withdraw

Withdraw assets if balance is available

function

withdraw (uint256 assets ,

address receiver ,

address owner) external

returns (uint256 shares); Parameters

Name Type Description assets uint256 The amount of assets that should be withdrawn receiver address The address user whom should receive the mevEth out owner address The address of the owner of the mevEth Returns

Name Type Description shares uint256 The amount of shares burned

withdrawQueue

Withdraw assets or open queue ticket for claim depending on balance available

function

withdrawQueue (uint256 assets ,

address receiver ,

address owner) external

returns (uint256 shares); Parameters

Name Type Description assets uint256 The amount of assets that should be withdrawn receiver address The address user whom should receive the mevEth out owner address The address of the owner of the mevEth Returns

Name Type Description shares uint256 The amount of shares burned

maxRedeem

Function to simulate the maximum amount of shares that can be redeemed by the owner.

function

maxRedeem (address owner) external

view

returns (uint256 maxShares); Parameters

Name Type Description owner address The address in question of who would be redeeming their shares Returns

Name Type Description maxShares uint256 The maximum amount of shares they could redeem

previewRedeem

Function to simulate the amount of assets that would be withdrawn for a specified amount of shares.

function

previewRedeem (uint256 shares) external

view

returns (uint256 assets); Parameters

Name Type Description shares uint256 The amount of shares that would be burned Returns

Name Type Description assets uint256 The amount of assets that would be withdrawn,under ideal conditions only

redeem

Function to redeem shares from the mevEth contract

function

redeem (uint256 shares ,

address receiver ,

address owner) external

returns (uint256 assets); Parameters

Name Type Description shares uint256 The amount of shares that should be burned receiver address The address user whom should receive the wETH out owner address The address of the owner of the mevEth Returns

Name Type Description assets uint256 The amount of assets withdrawn

max

Returns the largest of two numbers.

function

max (uint256 a ,

uint256 b) internal

pure

returns (uint256);

min

Returns the smallest of two numbers.

function

min (uint256 a ,

uint256 b) internal

pure

returns (uint256);

redeemCream

Redeem Cream staked eth tokens for mevETH at a fixed ratio

function

redeemCream (uint256 creamAmount) external ; Parameters

Name Type Description creamAmount uint256 The amount of Cream tokens to redeem

receive

Only Weth withdraw is defined for the behaviour. Deposits should be directed to deposit / mint. Rewards via grantRewards and validator withdraws via grantValidatorWithdraw.

receive () external

payable ;

transfer

function

transfer (address to ,

uint256 amount) public

virtual

override

returns (bool);

transferFrom

function

transferFrom (address from ,

address to ,

uint256 amount) public

virtual

override

returns (bool);

Events

MevEthInitialized

Event emitted when the MevEth is successfully initialized.

event

MevEthInitialized (address

indexed mevEthShareVault, address

indexed stakingModule);

StakingPaused

Emitted when staking is paused.

event

StakingPaused ();

StakingUnpaused

Emitted when staking is unpaused.

event

StakingUnpaused ();

StakingModuleUpdateCommitted

Event emitted when a new staking module is committed. The MODULE_UPDATE_TIME_DELAY must elapse before the staking module update can be finalized.

event

StakingModuleUpdateCommitted (address

indexed oldModule, address

indexed pendingModule, uint64

indexed eligibleForFinalization);

StakingModuleUpdateFinalized

Event emitted when a new staking module is finalized.

event

StakingModuleUpdateFinalized (address

indexed oldModule, address

indexed newModule);

StakingModuleUpdateCanceled

Event emitted when a new pending module update is canceled.

event

StakingModuleUpdateCanceled (address

indexed oldModule, address

indexed pendingModule);

MevEthShareVaultUpdateCommitted

Event emitted when a new share vault is committed. To finalize the update, the MODULE_UPDATE_TIME_DELAY must elapse and the finalizeUpdateMevEthShareVault function must be called.

event

MevEthShareVaultUpdateCommitted (address

indexed oldVault, address

indexed pendingVault, uint64

indexed eligibleForFinalization);

MevEthShareVaultUpdateFinalized

Event emitted when a new share vault is finalized.

event

MevEthShareVaultUpdateFinalized (address

indexed oldVault, address

indexed newVault);

MevEthShareVaultUpdateCanceled

Event emitted when a new pending share vault update is canceled.

event

MevEthShareVaultUpdateCanceled (address

indexed oldVault, address

indexed newVault);

ValidatorCreated

Event emitted when a new validator is created

event

ValidatorCreated (address

indexed stakingModule, IStakingModule.ValidatorData newValidator);

Rewards

Event emitted when rewards are granted.

event

Rewards (address sender, uint256 amount);

ValidatorWithdraw

Emitted when validator withdraw funds are received.

event

ValidatorWithdraw (address sender, uint256 amount);

WithdrawalQueueOpened

Event emitted when a withdrawal ticket is added to the queue.

event

WithdrawalQueueOpened (address

indexed recipient, uint256

indexed withdrawalId, uint256 assets);

WithdrawalQueueClosed

event

WithdrawalQueueClosed (address

indexed recipient, uint256

indexed withdrawalId, uint256 assets);

CreamRedeemed

event

CreamRedeemed (address

indexed redeemer, uint256 creamAmount, uint256 mevEthAmount);

Structs

Fraction

Central struct used for share accounting + math.

struct

Fraction { uint128 elastic; uint128 base; }

WithdrawalTicket

Struct representing a withdrawal ticket which is added to the withdrawal queue.

struct

WithdrawalTicket { bool claimed; address receiver; uint128 amount; uint128 accumulatedAmount; } Last updated onFebruary 7, 2024 Authmanager Mevethrateprovider