Perpetual Vault Protocol

The Perpetual Vault Protocol is a decentralized finance (DeFi) system that enables users to engage in leveraged trading on the GMX decentralized exchange. This protocol aims to simplify the process of managing leveraged positions while providing additional features such as automated position management and risk mitigation.

Users deposit USDC into the contract, and withdraw USDC from the contract. The USDC is used to open long/short positions on GMX perps. Each vaults represents exactly one perps market at a specified leverage. For example, we'll have a 1x ETH vault, 2x ETH vault, and 3x ETH vault. The leverage of each vault will stay consistent from start to finish. All short positions and long positions greater than 1x leverage involve opening a position on GMX perpetuals. If the position is 1x long, then we open a spot position by swapping thru any combination of GMX spot or Paraswap.

We implemented a Keeper system that executes actions via an asynchronous series of actions. We use enumerable sets to map users to their deposits. Each deposit can only be withdrawn in whole.

The strategy of signal changes (i.e. going from long to short or from short to long) is determined offchain and executed by the keeper.

1. What does the system do?

* Allows users to deposit collateral into a vault
* Manages leveraged long and short positions on GMX
* Automates position entry, exit, and management based on predefined signals
* Handles swaps between collateral and trading tokens
* Provides a way to compound gains and manage risks

2. What does it aim to achieve or what problem does it solve?

* Simplifies the process of leveraged trading for users who may not want to actively manage their positions

* Reduces the complexity of interacting directly with GMX for leveraged trading

* Allows for automated strategies to be implemented, potentially improving trading outcomes

* Provides a way for users to participate in leveraged trading with reduced active management

3. Who benefits or uses the protocol?

* Traders who want exposure to leveraged positions without active management

* Investors looking for automated trading strategies in the cryptocurrency market

* Users who want to engage in leveraged trading but find direct interaction with GMX complex

* DeFi enthusiasts looking for new ways to utilize their crypto assets

4. What are the limitations on values set by admins in the codebase?

* minDepositAmount = 1000

* maxDepositAmount = 100000

* minEth = 0.002 ether

* governanceFee = between 100 and 2000

* lockTime > 1 day

* callbackGasLimit = 2,000,000

* Vault leverage = Max leverage will be 3x leverage

5. What are the key invariants?

* Fair Share of Funding Fees: Depositors, in general, should not be able to claim more than their fair share of positive funding fees or pay more than their share of negative funding fees. The distribution of fees should be proportional to each depositor's share of the total

vault. There could be delays in claiming some funding fees. If the user withdraws prior to the ability to claim, then it would be ok not to receive his fair share.

* Total Shares Consistency: The sum of all individual depositor shares must always equal the totalShares variable in the contract. This ensures that the accounting of shares is accurate and that no shares are created or destroyed improperly.

* Depositor Share Value Preservation: The value of a depositor's shares should never decrease due to the actions of other depositors. Any losses or gains should be distributed proportionally based on share ownership. There are some exceptions, but there shouldn't be any material loss of depositor's share value.

* After all actions completed, nextAction, swapProgressData should be empty. PositionKey is 0 when no position

* Withdrawal Locks: No depositor should be able to withdraw their funds before the lockTime period has passed since their deposit. This lock period must be consistently enforced for all depositors.

[Website](https://www.gamma.xyz/)

[Twitter](https://x.com/gammastrategies)

[GitHub](https://github.com/GammaStrategies)

Actors

Detail which roles are included within your protocol, for example 'owner', 'borrower', 'organizer' etc. Draw clear links and outline the powers each actor should have and expected limitations to those powers. Please clearly detail your expected centralization risks.

1. Owner

Powers:

* Can set critical protocol parameters in PerpetualVault:

* Set keeper address

* Set treasury address

* Set min/max deposit amounts

* Set callback gas limit

* Set lock time

* Set deposit pause state

* Update vault reader address

* Set vault state in emergency situations

* Has control over GmxProxy settings:

* Update GMX contract addresses

* Set minimum ETH requirements

* Withdraw ETH from the contract

* Controls KeeperProxy configuration:

* Set price feed addresses and parameters

* Set threshold values

* Set maximum time windows

* Add/remove keeper addresses

2. Keeper

Powers:

* Execute position management operations:
* Call `run()` to open/close positions
* Execute `runNextAction()` for multi-step operations
* Cancel ongoing flows via `cancelFlow()`
* Cancel pending GMX orders via `cancelOrder()`
* Claim collateral rebates
* Must pass onlyKeeper modifier checks
* Responsible for monitoring and maintaining position health

3. Users (Depositors)

Powers:

* Deposit collateral tokens into the vault
* Request withdrawals of their deposited funds
* Cannot withdraw before lockup period (1 week by default)
* Must provide sufficient execution fees for operations

4. Treasury

Powers:

* Receives governance fees from profitable withdrawals
* Collects claimed rebates and other protocol fees

Expected Centralization Risks Assumed

1. Owner Privileges

* High degree of control over protocol parameters
* Can pause deposits
* Can update critical contract addresses
* Could potentially disrupt operations through parameter manipulation

2. Keeper Dependency

* System relies heavily on keeper for executing trades
* Single keeper point of failure if not properly distributed

* Malicious keeper could potentially front-run or delay transactions

* Assume that Keeper will always have enough gas to execute transactions. There is a pay execution fee function, but the assumption should be that there's more than enough gas to cover transaction failures, retries, etc

* There are two spot swap functionalies: (1) using GMX swap and (2) using Paraswap. We can assume that any swap failure will be retried until success.

3. Price Oracle Reliance

* Depends on Chainlink price feeds

* Owner controls price feed settings and thresholds

* Oracle manipulation risks if feeds are compromised

4. GMX Integration Risks

* Heavy dependency on GMX protocol functioning correctly

* Owner can update GMX-related addresses

* Changes in GMX protocol could impact system operations

* We can assume that the GMX keeper won't misbehave, delay, or go offline.

Risk Mitigation Features

1. Two-Step Ownership Transfer

* Uses OpenZeppelin's Ownable2StepUpgradeable for safer ownership transitions

2. Price Validation

* Implements checks against Chainlink oracle prices

* Includes grace periods and sequencer checks for L2

* Configurable thresholds for price deviations

3. Security Timeouts

* Mandatory lockup period for deposits

* Grace periods for oracle updates

* Execution time windows for orders

4. Reentrancy Protection

* Uses ReentrancyGuard for critical functions

* Implements GMX lock mechanism for position management