HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: 1A1Z
Date: June 23th, 2022

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for 1A1Z.
Approved By	Andrew Matiukhin | CTO Hacken OU
Type of Contracts	AMM
Platform	EVM
Language	Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Website	https://website.com
Timeline	01.03.2022 - 20.06.2022
Changelog	21.03.2022 - Initial Review 29.03.2022 - Revising 26.04.2022 - Revising 04.05.2022 - Revising 23.06.2022 - Revising

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Introduction 4

Scope 4

Executive Summary 6

AS-IS overview 7

Severity Definitions 27

Findings 28

Disclaimers 32

Appendix A. Smart Contracts Security Issues 33

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by 1A1Z (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contracts.

Scope

The scope of the project is smart contracts in the repository:

Repository:
https://github.com/DistributedCollective/sovryn-perpetual-swap
Commit:
b7383fc6884146bc59bad672683dcaeb9ba9ecef
Technical Documentation: Yes (README.md, SovrynPerpetualsV1.pdf)
JS tests: Yes (test)
Contracts:

```
perpetual/token/ShareToken.sol
perpetual/token/ShareTokenFactory.sol
perpetual/core/PerpStorage.sol
perpetual/core/PerpetualManagerProxy.sol
perpetual/limitorder/LimitOrderBookBeacon.sol
perpetual/limitorder/LimitOrderBook.sol
perpetual/limitorder/LimitOrderBookFactory.sol
perpetual/functions/PerpetualHashFunctions.sol
perpetual/functions/PerpetualRebalanceFunctions.sol
perpetual/functions/AMMPerpLogic.sol
perpetual/functions/PerpetualTradeFunctions.sol
perpetual/functions/PerpetualUpdateFunctions.sol
perpetual/functions/PerpetualBaseFunctions.sol
perpetual/functions/PerpetualWithdrawFunctions.sol
perpetual/modules/PerpetualPoolFactory.sol
perpetual/modules/PerpetualSettlement.sol
perpetual/modules/PerpetualLimitTradeManager.sol
perpetual/modules/PerpetualLiquidator.sol
perpetual/modules/PerpetualUpdateLogic.sol
perpetual/modules/PerpetualTradeLogic.sol
perpetual/modules/PerpetualRelayRecipient.sol
perpetual/modules/PerpetualWithdrawManager.sol
perpetual/modules/PerpetualRebalanceLogic.sol
perpetual/modules/PerpetualMarginLogic.sol
perpetual/modules/PerpetualTreasury.sol
perpetual/modules/PerpetualGetter.sol
perpetual/modules/PerpetualTradeManager.sol
perpetual/modules/PerpetualTradeLimits.sol
perpetual/modules/PerpetualWithdrawAllManager.sol
perpetual/modules/PerpetualOrderManager.sol
perpetual/modules/PerpetualDepositManager.sol
perpetual/modules/PerpetualFactory.sol
perpetual/modules/PerpetualMarginViewLogic.sol
perpetual/interfaces/IPerpetualLimitTradeManager.sol
perpetual/interfaces/IPerpetualWithdrawManager.sol
perpetual/interfaces/IPerpetualPoolFactory.sol
```

```
perpetual/interfaces/IAMMPerpLogic.sol
perpetual/interfaces/IPerpetualTreasury.sol
perpetual/interfaces/IPerpetualWithdrawAllManager.sol
perpetual/interfaces/IPerpetualTradeManager.sol
perpetual/interfaces/IPerpetualFactory.sol
perpetual/interfaces/IPerpetualRelayRecipient.sol
perpetual/interfaces/IPerpetualOrderManager.sol
perpetual/interfaces/IPerpetualUpdateLogic.sol
perpetual/interfaces/ISOVLibraryEvents.sol
perpetual/interfaces/IPerpetualOrder.sol
perpetual/interfaces/IPerpetualSettlement.sol
perpetual/interfaces/IPerpetualRebalanceLogic.sol
perpetual/interfaces/IPerpetualMarginViewLogic.sol
perpetual/interfaces/IPerpetualMarginLogic.sol
perpetual/interfaces/IFunctionList.sol
perpetual/interfaces/IPerpetualLiquidator.sol
perpetual/interfaces/IPerpetualTradeLimits.sol
perpetual/interfaces/IPerpetualGetter.sol
perpetual/interfaces/IPerpetualManager.sol
perpetual/interfaces/IPerpetualTradeLogic.sol
perpetual/interfaces/IPerpetualDepositManager.sol
interface/IChainLinkPriceFeed.sol
interface/AggregatorV3Interface.sol
interface/IPriceFeedsExt.sol
interface/IShareTokenFactory.sol
interface/IShareToken.sol
interface/ISpotOracle.sol
gsn/forwarder/IForwarder.sol
gsn/RbtcPaymaster.sol
gsn/utils/GsnEip712Library.sol
gsn/utils/MinLibBytes.sol
gsn/utils/GsnUtils.sol
gsn/utils/GsnTypes.sol
gsn/RbtcPaymasterTestnet.sol
gsn/BaseRelayRecipient.sol
gsn/BasePaymaster.sol
gsn/interfaces/IRelayHub.sol
gsn/interfaces/IRbtcPaymaster.sol
gsn/interfaces/IPaymaster.sol
gsn/interfaces/IStakeManager.sol
gsn/interfaces/IRelayRecipient.sol
oracle/OracleInterfaceID.sol
oracle/SpotOracle.sol
oracle/OracleFactory.sol
oracle/AbstractOracle.sol
libraries/QuickSort.sol
libraries/OrderFlags.sol
libraries/RSKAddrValidator.sol
libraries/ABDKMath64x64.sol
libraries/ConverterDec18.sol
libraries/EnumerableBytes4Set.sol
libraries/EnumerableSetUpgradeable.sol
libraries/Utils.sol
libraries/Bytes32Pagination.sol
cdf/CDFTable.sol
cdf/ICDFTable.sol
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ EIP standards violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency
Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency ▪ Kill-Switch Mechanism

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

The score measurements details can be found in the corresponding section of the methodology.

Documentation quality

The Customer provided superficial functional requirements and technical requirements. The total Documentation Quality score is **10** out of **10**.

Code quality

The total CodeQuality score is **9** out of **10**. The code is nicely commented. Unit tests were provided. There are NatSpec blocks but not full and not all.

Architecture quality

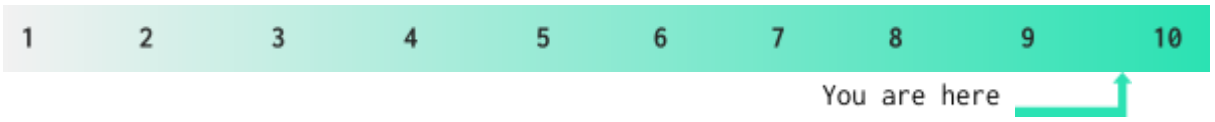
The architecture quality score is **9** out of **10**. The logic is implemented in split into corresponding files. The logic is clear while reading provided docs.

Security score

As a result of the audit, security engineers found **no new issues**. The security score is **10** out of **10**. All found issues are displayed in the “Issues overview” section.

Summary

According to the assessment, the Customer's smart contract has the following score: **9.8**



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

AS-IS overview

PerpStorage.sol

Description

The contract holds storage variables for manager contracts through the proxy.

Imports

PerpStorage contract has the following imports:

- @openzeppelin/contracts/access/Ownable.sol
- @openzeppelin/contracts/utils/ReentrancyGuard.sol
- ../../interface/IShareTokenFactory.sol
- ../../libraries/ABDKMath64x64.sol
- ./../functions/AMMPerpLogic.sol
- ../../libraries/EnumerableSetUpgradeable.sol
- ../../libraries/EnumerableBytes4Set.sol
- ../../gsn/BaseRelayRecipient.sol

Inheritance

PerpStorage contract has the following inheritance:

- Ownable
- ReentrancyGuard
- BaseRelayRecipient

Usages

PerpStorage contract has the following custom usages:

- using ABDKMath64x64 for int128;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;
- using EnumerableBytes4Set for EnumerableBytes4Set.Bytes4Set;

Structs

PerpStorage contract has the following structures defined:

- struct OraclePriceData {
  int128 fPrice;
  uint256 time;
  bool isInSignificant;
  }

- struct MarginAccount {
  int128 fLockedInValueQC;
  int128 fCashCC;
  int128 fPositionBC;
  int128 fUnitAccumulatedFundingStart;
  bytes32 positionId;
  }

- struct PerpetualData {
  bytes32 id;
  uint256 poolId;
  address oracleS2Addr;

```
    address oracleS3Addr;
    OraclePriceData currentPremiumRate;
    OraclePriceData currentMarkPremiumRate;
    int128 premiumRatesEMA;
    OraclePriceData settlementMarkPremiumRate;
    OraclePriceData settlementS2PriceData;
    OraclePriceData settlementS3PriceData;
    int128 fCurrentFundingRate;
    int128 fUnitAccumulatedFunding;
    PerpetualState state;
    int128 fOpenInterest;
    int128 fAMMFundCashCC;
    int128 fkStar;
    int128 fkStarSide;
    int128 fInitialMarginRateAlpha;
    int128 fMarginRateBeta;
    int128 fInitialMarginRateCap;
    int128 fMaintenanceMarginRateAlpha;
    int128 fTreasuryFeeRate;
    int128 fPnLPartRate;
    int128 fReferralRebateCC;
    int128 fLiquidationPenaltyRate;
    int128 fMinimalSpread;
    int128 fMinimalSpreadInStress;
    int128 fLotSizeBC;
    int128 fFundingRateClamp;
    int128 fMarkPriceEMALambda;
    int128 fSigma2;
    int128 fSigma3;
    int128 fRho23;
    AMMPerpLogic.CollateralCurrency eCollateralCurrency;
    int128[2] fStressReturnS2;
    int128[2] fStressReturnS3;
    int128 fDFCoverNRate;
    int128[2] fDFLambda;
    int128[2] fAMMTargetDD;
    int128 fAMMMinSizeCC;
    int128 fMinimalTraderExposureEMA;
    int128 fMinimalAMMExposureEMA;
    int128 fMaximalTradeSizeBumpUp;
    int128 fTargetAMMFundSize;
    bool isBaselineAMMFundState;
    int128 fTargetDFSize;
    int128[2] fCurrentAMMExposureEMA;
    int128 fCurrentTraderExposureEMA;
    int128 fTotalMarginBalance;
    int128 fMaxPositionBC;
}
```

- struct Checkpoint {
  uint128 timestamp;
  int128 amount;
  }

- struct LiquidityPoolData {
  uint256 id;
  bool isRunning;
  address treasuryAddress;
  address marginTokenAddress;
  address shareTokenAddress;
  uint256 iTargetPoolSizeUpdateTime;
  int128 fPnLparticipantsCashCC;
  int128 fAMMFundCashCC;
  int128 fDefaultFundCashCC;
  uint256 iPriceUpdateTimeSec;
  int128 fTargetAMMFundSize;
  int128 fTargetDFSize;
  uint256 iLastTargetPoolSizeTime;
  uint256 iLastFundingTime;
  uint256 iPnLparticipantWithdrawalPeriod;
  int128 fPnLparticipantWithdrawalPercentageLimit;
  int128 fPnLparticipantWithdrawalMinAmountLimit;
  int128 fRedemptionRate;
  uint256 iPerpetualCount;
  int128 fMaxTotalTraderFunds;
  }

Enums

PerpStorage contract has the following enums:

```
enum PerpetualState {
    INVALID,
    INITIALIZING,
    NORMAL,
    EMERGENCY,
    CLEARED
}
```

Events

PerpStorage contract has no custom events.

Modifiers

PerpStorage contract has no custom modifiers.

Fields

PerpStorage contract has no public fields and constants.

Functions

PerpStorage contract has no external/public functions.

PerpetualManagerProxy.sol

Description

The proxy contract, which is being used to call different modules.

Imports

PerpetualManagerProxy contract has the following imports:

- @openzeppelin/contracts/proxy/Proxy.sol
- @openzeppelin/contracts/proxy/Proxy.sol
- ./PerpStorage.sol
- ../interfaces/ISOVLibraryEvents.sol
- ../interfaces/IFunctionList.sol
- ../../libraries/EnumerableBytes4Set.sol
- ../../libraries/Utils.sol

Inheritance

PerpetualManagerProxy contract has the following inheritance:

- PerpStorage
- Proxy
- ISOVLibraryEvents

Usages

PerpetualManagerProxy contract has the following custom usages:

- using EnumerableBytes4Set for EnumerableBytes4Set.Bytes4Set;

Structs

PerpetualManagerProxy contract has no data structures defined.

Enums

PerpetualManagerProxy contract has no enums.

Events

PerpetualManagerProxy contract has the following events:

- event ProxyOwnershipTransferred(address indexed _oldOwner, address indexed _newOwner);
- event ImplementationChanged(bytes4 _sig, address indexed _oldImplementation, address indexed _newImplementation);

Modifiers

PerpetualManagerProxy contract has the following modifiers:

- onlyProxyOwner

Fields

PerpetualManagerProxy contract has no public fields and constants.

Functions

PerpetualManagerProxy contract has the following external/public functions:

- getImplementation(bytes4 _sig) external view returns (address)
- setImplementation(address _impl) external onlyProxyOwner
- setImplementationCrossModules(address _impl) external onlyProxyOwner
- getModuleImplementationAddress(string memory _moduleName) external view returns (address)
- setProxyOwner(address _owner) external onlyProxyOwner
- getProxyOwner() public view returns (address _owner)

PerpetualDepositManager.sol

Description

The contract that manages the deposits.

Imports

PerpetualDepositManager contract has the following imports:

- ../functions/PerpetualBaseFunctions.sol
- ../interfaces/IPerpetualDepositManager.sol
- ./../functions/PerpetualUpdateFunctions.sol
- ../interfaces/IFunctionList.sol
- ../../libraries/Utils.sol

Inheritance

PerpetualDepositManager contract has the following inheritance:

- PerpetualBaseFunctions
- PerpetualUpdateFunctions
- IFunctionList
- IPerpetualDepositManager

Usages

PerpetualDepositManager contract has the following custom usages:

- using ABDKMath64x64 for int128;
- using ConverterDec18 for int128;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;

Structs

PerpetualDepositManager contract has no data structures defined.

Enums

PerpetualDepositManager contract has no enums.

Events

PerpetualDepositManager contract has no custom events.

Modifiers

PerpetualDepositManager contract has no custom modifiers.

Fields

PerpetualDepositManager contract has no public fields and constants.

Functions

PerpetualDepositManager contract has the following external/public functions:

- deposit(bytes32 _iPerpetualId, int128 _fAmount) external override nonReentrant
- depositToDefaultFund(uint256 _poolId, int128 _fAmount) external override onlyOwner
- withdrawFromDefaultFund(uint256 _poolId, address _receiver, int128 _fAmount) external override onlyOwner
- transferEarningsToTreasury(uint256 _poolId, int128 _fAmount) external override onlyOwner
- getFunctionList() external pure virtual override returns (bytes4[] memory, bytes32)

PerpetualLimitTradeManager.sol

Description

The contract that manages the limit trades.

Imports

PerpetualLimitTradeManager contract has the following imports:

- ../interfaces/IFunctionList.sol
- ../../libraries/RSKAddrValidator.sol
- ../interfaces/IPerpetualLimitTradeManager.sol
- ../functions/PerpetualTradeFunctions.sol

Inheritance

PerpetualLimitTradeManager contract has the following inheritance:

- PerpetualTradeFunctions
- IFunctionList
- IPerpetualLimitTradeManager

Usages

PerpetualLimitTradeManager contract has no custom usages.

Structs

PerpetualLimitTradeManager contract has no data structures defined.

Enums

PerpetualLimitTradeManager contract has no enums.

Events

PerpetualLimitTradeManager contract has no custom events.

Modifiers

PerpetualLimitTradeManager contract has no custom modifiers.

Fields

PerpetualLimitTradeManager contract has no public fields and constants.

Functions

PerpetualLimitTradeManager contract has the following external/public functions:

- tradeBySig(Order memory _order, bytes memory signature) external override returns (int128)
- getFunctionList() external pure virtual override returns (bytes4[] memory, bytes32)

PerpetualOrderManager.sol

Description

The contract allows canceling orders.

Imports

PerpetualOrderManager contract has the following imports:

- ../interfaces/IFunctionList.sol
- ../../libraries/RSKAddrValidator.sol
- ../functions/PerpetualTradeFunctions.sol
- ../interfaces/IPerpetualOrderManager.sol

Inheritance

PerpetualOrderManager contract has the following inheritance:

- PerpetualTradeFunctions
- IFunctionList
- IPerpetualLimitTradeManager

Usages

PerpetualOrderManager contract has no custom usages.

Structs

PerpetualOrderManager contract has no data structures defined.

Enums

PerpetualOrderManager contract has no enums.

Events

PerpetualOrderManager contract has no custom events.

Modifiers

PerpetualOrderManager contract has no custom modifiers.

Fields

The contract has no public fields and constants.

Functions

PerpetualOrderManager contract has the following external/public functions:

- cancelOrder(Order memory _order, bytes memory _signature) external override
- getFunctionList() external pure virtual override returns (bytes4[] memory, bytes32)

PerpetualTradeManager.sol

Description

The contract that allows to trade orders.

Imports

PerpetualTradeManager contract has the following imports:

- ../interfaces/IPerpetualTradeManager.sol
- ../interfaces/IFunctionList.sol
- ../functions/PerpetualTradeFunctions.sol

Inheritance

PerpetualTradeManager contract has the following inheritance:

- PerpetualTradeFunctions
- IFunctionList
- IPerpetualTradeManager

Usages

PerpetualTradeManager contract has no custom usages.

Structs

PerpetualTradeManager contract has no data structures defined.

Enums

PerpetualTradeManager contract has no enums.

Events

PerpetualTradeManager contract has no custom events.

Modifiers

PerpetualTradeManager contract has no custom modifiers.

Fields

PerpetualTradeManager contract has no public fields and constants.

Functions

PerpetualTradeManager contract has the following external/public functions:

- trade(Order memory _order) external override returns (int128)
- getFunctionList() external pure virtual override returns (bytes4[] memory, bytes32)

PerpetualWithdrawManager.sol

Description

The contract allows withdrawing a specified amount margin from the trader's account of the perpetual.

Imports

PerpetualWithdrawManager contract has the following imports:

- ./../functions/PerpetualWithdrawFunctions.sol
- ./../interfaces/IPerpetualWithdrawAllManager.sol
- ./../interfaces/IFunctionList.sol
- ../../interface/ISpotOracle.sol
- ../../libraries/Utils.sol

Inheritance

PerpetualWithdrawManager contract has the following inheritance:

- PerpetualWithdrawFunctions
- IFunctionList
- IPerpetualWithdrawAllManager

Usages

PerpetualWithdrawManager contract has no custom usages.

Structs

PerpetualWithdrawManager contract has no data structures defined.

Enums

PerpetualWithdrawManager contract has no enums.

Events

PerpetualWithdrawManager contract has no custom events.

Modifiers

PerpetualWithdrawManager contract has no custom modifiers.

Fields

PerpetualWithdrawManager contract has no public fields and constants.

Functions

PerpetualWithdrawManager contract has the following external/public functions:

- withdraw(bytes32 _iPerpetualId, int128 _fAmount) external override nonReentrant updateFundingAndPrices(perpetualPoolIds[_iPerpetualId])
- getFunctionList() external pure virtual override returns (bytes4[] memory, bytes32)

PerpetualWithdrawAllManager.sol

Description

The contract allows withdrawing the entire margin from the trader's account of the perpetual.

Imports

PerpetualWithdrawAllManager contract has the following imports:

- ./../functions/PerpetualWithdrawFunctions.sol
- ./../interfaces/IPerpetualWithdrawAllManager.sol
- ./../interfaces/IFunctionList.sol
- ../../interface/ISpotOracle.sol
- ../../libraries/Utils.sol

Inheritance

PerpetualWithdrawAllManager contract has the following inheritance:

- PerpetualWithdrawFunctions
- IFunctionList
- IPerpetualWithdrawAllManager

Usages

PerpetualWithdrawAllManager contract has no custom usages.

Structs

PerpetualWithdrawAllManager contract has no data structures defined.

Enums

PerpetualWithdrawAllManager contract has no enums.

Events

PerpetualWithdrawAllManager contract has no custom events.

Modifiers

PerpetualWithdrawAllManager contract has no custom modifiers.

Fields

PerpetualWithdrawAllManager contract has no public fields and constants.

Functions

PerpetualWithdrawAllManager contract has the following external/public functions:

- withdrawAll(bytes32 _iPerpetualId) external override nonReentrant updateFundingAndPrices(perpetualPoolIds[_iPerpetualId])
- getFunctionList() external pure virtual override returns (bytes4[] memory, bytes32)

LimitOrderBook.sol

Description

Limit/Stop Order Book Proxy Contract. A new perpetual limit order book contract.

Imports

LimitOrderBook contract has the following imports:

- @openzeppelin/contracts-upgradeable/proxy/Initializable.sol
- @openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol
- @openzeppelin/contracts/cryptography/ECDSA.sol
- ../interfaces/IPerpetualManager.sol
- ../interfaces/IPerpetualOrder.sol
- ../../libraries/RSKAddrValidator.sol
- ../../libraries/Bytes32Pagination.sol
- ../functions/PerpetualHashFunctions.sol

Inheritance

LimitOrderBook contract has the following inheritance:

- IPerpetualOrder
- Initializable
- PerpetualHashFunctions

Usages

LimitOrderBook contract has the following usages:

- using Bytes32Pagination for bytes32[];

Structs

LimitOrderBook contract has no data structures defined.

Enums

LimitOrderBook contract has no enums.

Events

LimitOrderBook contract has the following events:

- event PerpetualLimitOrderCreated(bytes32 indexed perpetualId, address indexed trader, int128 limitPrice, int128 triggerPrice, bytes32 digest);

Modifiers

LimitOrderBook contract has no custom modifiers.

Fields

LimitOrderBook contract has the following fields and constants:

- bytes32[] public allDigests;
- mapping(address => bytes32[]) public digestsOfTrader;
- mapping(bytes32 => IPerpetualOrder.Order) public orderOfDigest;
- mapping(bytes32 => bytes) public orderSignature;
- mapping(bytes32 => bytes32) public nextOrderHash;
- mapping(bytes32 => bytes32) public prevOrderHash;
- bytes32 public lastOrderHash;
- uint256 public orderCount;
- IPerpetualManager public perpManager;
- bytes32 public perpetualId;
- IERC20Upgradeable public marginToken;

Functions

LimitOrderBook contract has the following external/public functions:

- initialize(address _perpetualManagerAddr, bytes32 _perpetualId) external initializer
- createLimitOrder(Order memory _order, bytes memory signature, uint256 _allowance) external
- executeLimitOrder(Order memory _order) external
- cancelLimitOrder(bytes32 _digest, bytes memory _signature) public
- numberOfDigestsOfTrader(address trader) external view returns (uint256)
- numberOfAllDigests() external view returns (uint256)
- numberOfOrderBookDigests() external view returns (uint256)
- limitDigestsOfTrader(address trader, uint256 page, uint256 limit) external view returns (bytes32[] memory)
- allLimitDigests(uint256 page, uint256 limit) external view returns (bytes32[] memory)
- getTrader(bytes32 digest) external view returns (address trader)
- getOrders(address trader, uint256 offset, uint256 limit) external view returns (Order[] memory orders)
- getSignature(bytes32 digest) public view returns (bytes memory signature)
- pollLimitOrders(bytes32 _startAfter, uint256 _numElements) external view returns (Order[] memory orders, bytes32[] memory orderHashes)

PerpetualBaseFunctions.sol

Description

The contract contains the base set of functions.

Imports

PerpetualBaseFunctions contract has the following imports:

- @openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol
- @openzeppelin/contracts-upgradeable/token/ERC20/SafeERC20Upgradeable.sol
- @openzeppelin/contracts/token/ERC20/ERC20.sol
- ../core/PerpStorage.sol
- ../interfaces/ISOVLibraryEvents.sol
- ../../libraries/ConverterDec18.sol
- ../../libraries/EnumerableSetUpgradeable.sol
- ../interfaces/IPerpetualRebalanceLogic.sol

Inheritance

PerpetualBaseFunctions contract has the following inheritance:

- PerpStorage
- ISOVLibraryEvents

Usages

PerpetualBaseFunctions contract has the following usages:

- using ABDKMath64x64 for int128;
- using ConverterDec18 for int128;
- using ConverterDec18 for int256;
- using SafeERC20Upgradeable for IERC20Upgradeable;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;

Structs

PerpetualBaseFunctions contract has no data structures defined.

Enums

PerpetualBaseFunctions contract has no enums.

Events

PerpetualBaseFunctions contract has no custom events.

Modifiers

PerpetualBaseFunctions contract has no custom modifiers.

Fields

PerpetualBaseFunctions contract has no public fields and constants.

Functions

PerpetualBaseFunctions contract has no external/public functions.

PerpetualHashFunctions.sol

Description

The contract contains the set of hash functions.

Imports

PerpetualHashFunctions contract has the following imports:

- @openzeppelin/contracts/cryptography/ECDSA.sol
- ../interfaces/IPerpetualOrder.sol

Inheritance

PerpetualHashFunctions contract does not inherit any contract.

Usages

PerpetualHashFunctions contract has the following usages:

- using ABDKMath64x64 for int128;
- using ConverterDec18 for int128;
- using ConverterDec18 for int256;
- using SafeERC20Upgradeable for IERC20Upgradeable;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;

Structs

PerpetualHashFunctions contract has no data structures defined.

Enums

PerpetualHashFunctions contract has no enums.

Events

PerpetualHashFunctions contract has no custom events.

Modifiers

PerpetualHashFunctions contract has no custom modifiers.

Fields

PerpetualHashFunctions contract has no public fields and constants.

Functions

PerpetualHashFunctions contract has no external/public functions.

PerpetualRebalanceFunctions.sol

Description

The contract contains the set of rebalancing functions.

Imports

PerpetualRebalanceFunctions contract has the following imports:

- @openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol
- @openzeppelin/contracts-upgradeable/token/ERC20/SafeERC20Upgradeable.sol
- @openzeppelin/contracts/token/ERC20/ERC20.sol
- ./PerpetualBaseFunctions.sol
- ../interfaces/IPerpetualTradeLogic.sol
- ../interfaces/IPerpetualUpdateLogic.sol
- ../interfaces/IPerpetualGetter.sol
- ../interfaces/IPerpetualMarginLogic.sol

Inheritance

PerpetualRebalanceFunctions contract has the following inheritance:

- PerpetualBaseFunctions

Usages

PerpetualRebalanceFunctions contract has the following usages:

- using ABDKMath64x64 for int128;

Structs

PerpetualRebalanceFunctions contract has no data structures defined.

Enums

PerpetualRebalanceFunctions contract has no enums.

Events

PerpetualRebalanceFunctions contract has no custom events.

Modifiers

PerpetualRebalanceFunctions contract has the following modifiers:

- onlyThis
- updateFundingAndPrices

Fields

PerpetualRebalanceFunctions contract has no public fields and constants.

Functions

PerpetualRebalanceFunctions contract has no external/public functions.

PerpetualTradeFunctions.sol

Description

The contract contains the set of trading functions.

Imports

PerpetualTradeFunctions contract has the following imports:

- @openzeppelin/contracts/proxy/Proxy.sol
- ../../libraries/OrderFlags.sol
- ../../interface/ISpotOracle.sol
- ../functions/PerpetualRebalanceFunctions.sol
- ../interfaces/IPerpetualTradeManager.sol
- ../interfaces/IPerpetualDepositManager.sol
- ../interfaces/IFunctionList.sol
- ../../libraries/RSKAddrValidator.sol
- ../../libraries/Utils.sol
- ../functions/PerpetualHashFunctions.sol
- ../interfaces/IPerpetualOrder.sol

Inheritance

PerpetualTradeFunctions contract has the following inheritance:

- PerpetualRebalanceFunctions
- PerpetualHashFunctions

Usages

PerpetualTradeFunctions contract has the following usages:

- using ABDKMath64x64 for int128;
- using ConverterDec18 for int128;
- using OrderFlags for uint32;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;

Structs

PerpetualTradeFunctions contract has no data structures defined.

Enums

PerpetualTradeFunctions contract has no enums.

Events

PerpetualTradeFunctions contract has no custom events.

Modifiers

PerpetualTradeFunctions contract has the following modifiers:

- onlyThis
- updateFundingAndPrices

Fields

PerpetualTradeFunctions contract has no public fields and constants.

Functions

PerpetualTradeFunctions contract has no external/public functions.

PerpetualUpdateFunctions.sol

Description

The contract contains the set of trading functions.

Imports

PerpetualUpdateFunctions contract has the following imports:

- @openzeppelin/contracts/proxy/Proxy.sol
- ../../libraries/OrderFlags.sol
- ../../interface/ISpotOracle.sol
- ../functions/PerpetualRebalanceFunctions.sol
- ../interfaces/IPerpetualTradeManager.sol
- ../interfaces/IPerpetualDepositManager.sol
- ../interfaces/IFunctionList.sol
- ../../libraries/RSKAddrValidator.sol
- ../../libraries/Utils.sol
- ../functions/PerpetualHashFunctions.sol
- ../interfaces/IPerpetualOrder.sol

Inheritance

PerpetualUpdateFunctions contract has the following inheritance:

- PerpetualRebalanceFunctions
- PerpetualHashFunctions

Usages

PerpetualUpdateFunctions contract has the following usages:

- using ABDKMath64x64 for int128;
- using ConverterDec18 for int128;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;

Structs

PerpetualUpdateFunctions contract has no data structures defined.

Enums

PerpetualUpdateFunctions contract has no enums.

Events

PerpetualUpdateFunctions contract has no custom events.

Modifiers

PerpetualUpdateFunctions contract has the following modifiers:

- onlyThis
- updateFundingAndPrices

Fields

PerpetualUpdateFunctions contract has no public fields and constants.

Functions

PerpetualUpdateFunctions contract has no external/public functions.

PerpetualWithdrawFunctions.sol

Description

The contract contains the set of withdrawal functions.

Imports

PerpetualWithdrawFunctions contract has the following imports:

- ../../interface/ISpotOracle.sol
- ./PerpetualRebalanceFunctions.sol

Inheritance

PerpetualWithdrawFunctions contract has the following inheritance:

- PerpetualRebalanceFunctions

Usages

PerpetualWithdrawFunctions contract has the following usages:

- using ABDKMath64x64 for int128;
- using EnumerableSetUpgradeable for EnumerableSetUpgradeable.AddressSet;

Structs

PerpetualWithdrawFunctions contract has no data structures defined.

Enums

PerpetualWithdrawFunctions contract has no enums.

Events

PerpetualWithdrawFunctions contract has no custom events.

Modifiers

PerpetualWithdrawFunctions contract has the following modifiers:

- onlyThis
- updateFundingAndPrices

Fields

PerpetualWithdrawFunctions contract has no public fields and constants.

Functions

PerpetualWithdrawFunctions contract has no external/public functions.

SpotOracle.sol

Description

The contract provides an ability to get the spot information, including the spot price.

Imports

SpotOracle contract has the following imports:

- @openzeppelin/contracts/math/SafeMath.sol
- ./AbstractOracle.sol
- ../interface/IPriceFeedsExt.sol
- ../interface/IChainLinkPriceFeed.sol
- ../libraries/ABDKMath64x64.sol
- ../libraries/QuickSort.sol
- ../libraries/ConverterDec18.sol

Inheritance

SpotOracle contract has the following inheritance:

- AbstractOracle

Usages

SpotOracle contract has the following usages:

- using ABDKMath64x64 for uint256;
- using ABDKMath64x64 for int128;
- using ConverterDec18 for int256;
- using QuickSort for uint256[];
- using SafeMath for uint256;

Structs

SpotOracle contract has no data structures defined.

Enums

SpotOracle contract has no enums.

Events

SpotOracle contract has no custom events.

Modifiers

SpotOracle contract has the following modifiers:

- onlyThis
- updateFundingAndPrices

Fields

SpotOracle contract has the following public fields and constants:

- uint256 public constant CHAIN_LINK_MULTIPLIER = 10**10;
- address[] public priceFeeds;
- bool[] public isIChainLink;
- bool private marketClosed;
- bool private terminated;

Functions

SpotOracle contract has no external/public functions:

- setMarketClosed(bool _marketClosed) external override onlyOwner
- isMarketClosed() external view override returns (bool)
- setTerminated(bool _terminated) external override onlyOwner
- isTerminated() external view override returns (bool)
- getSpotPrice() public view virtual override returns (int128, uint256)

- getBaseCurrency() external view override returns (bytes32)
- getQuoteCurrency() external view override returns (bytes32)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Findings

Critical

No critical severity issues were found.

High

No high severity issues were found.

Medium

1. **Tests coverage**

   The tests coverage is less than the best practices recommended. The coverage is about 85% for statements and 71% for branches. It is recommended to be at least 95% for statements and up to 100% for branches. The entire business logic cases and the negative scenarios should be covered.

   Scope: tests

   Recommendation: make sure tests cover at least 95% of statements and up to 100% of code branches.

   Status: Fixed (Revised Commit: d731dd3)

2. **Contracts that lock Ether.**

   Contract `PerpetualManagerProxy` extends the openzeppelin's `Proxy` contract which has defined `fallback` and `receive` **payable** functions. While the contract does not have any withdrawal functions, and if the set implementation contract would not have such, that will result that any Ether sent to the contract address would be lost forever.

   Contract: PerpetualManagerProxy.sol

   Functions: fallback, receive

   Recommendation: override the `_fallback` function to disallow ether receiving or add the withdrawal function to unlock locked ethers.

   Status: Fixed (Revised Commit: d731dd3)

Low

1. **Unused local variable.**

   Variable `fCenterKey` is never used. It does contain the int128 value from the `fKeys` mapping by the `centerIndex`. It is never used later in the code.

   Contract: CDFTable.sol

   Function: _getKeyRange

   Recommendation: remove unused variable.

   Status: Fixed (Revised Commit: d731dd3)

2. **Unused local variable.**

   Variable `spreads` is never used. It is declared as the `int128[2]` but neither initialized nor used in the code later.

   Contract: PerpetualBaseFunctions.sol

   Function: _updateInsurancePremium

   Recommendation: remove unused variable.

   Status: Fixed (Revised Commit: d731dd3)

3. **Unused function parameter.**

   Parameter `_ammVars` is never used in the function.

   Contract: AMMPerpLogic.sol

   Function: _calculateRiskNeutralDDNoQuanto

   Recommendation: remove a name for the unused parameter in the function definition.

   Status: Fixed (Revised Commit: d731dd3)

4. **Unused function parameter.**

   Parameter `_ammVars` is never used in the function.

   Contract: AMMPerpLogic.sol

   Function: _calculateStandardDeviationQuanto

   Recommendation: remove a name for the unused parameter in the function definition.

   Status: Fixed (Revised Commit: d731dd3)

5. **Unused function parameter.**

   Parameter `_perpetualId` is never used in the function.

   Contract: PerpetualBaseFunctions.sol

   Function: _checkWhitelist

   Recommendation: remove a name for the unused parameter in the function definition.

   Status: Fixed (Revised Commit: d731dd3)

6. **View function that should be pure.**

   View function that does neither reading any state variables nor external calls should be declared pure to save some gas.

   Contract: PerpetualHashFunctions.sol

   Function: _getDigest

   Recommendation: use keyword **pure** instead of **view**.

Status: Fixed (Revised Commit: d731dd3)

7. **View function that should be pure.**

   View function that does neither reading any state variables nor external calls should be declared pure to save some gas.

   Contract: PerpetualRelayRecipient.sol

   Function: versionRecipient

   Recommendation: use keyword **pure** instead of **view**.

   Status: Fixed (Revised Commit: d731dd3)

8. **View function that should be pure.**

   View function that does neither reading any state variables nor external calls should be declared pure to save some gas.

   Contract: PerpetualManagerProxy.sol

   Function: _getFunctionList

   Recommendation: use keyword **pure** instead of **view**.

   Status: Fixed (Revised Commit: d731dd3)

9. **A function that should be declared as a view.**

   A function that does not change any state should be declared as `view` to save gas.

   Contract: PerpetualTreasury.sol

   Function: _checkPoolState

   Recommendation: use keyword **view** to declare a function as `view`

   Status: Fixed (Revised Commit: d731dd3)

10. **A function that should be declared as a view.**

    A function that does not change any state should be declared as `view` to save gas.

    Contract: PerpetualTreasury.sol

    Function: _checkWithdrawalRestrictions

    Recommendation: use keyword **view** to declare a function as `view`

    Status: Fixed (Revised Commit: d731dd3)

11. **A function that should be declared as a view.**

    A function that does not change any state should be declared as `view` to save gas.

    Contract: PerpetualManagerProxy.sol

    Function: _checkClashing

Recommendation: use keyword **view** to declare a function as `view`

Status: Fixed (Revised Commit: d731dd3)

12. A function that should be declared as a view.

A function that does not change any state should be declared as `view` to save gas.

Contract: PerpetualWithdrawFunctions.sol

Function: _validateInputData

Recommendation: use keyword **view** to declare a function as `view`

Status: Fixed (Revised Commit: d731dd3)

13. Solidity compiler version.

An old solidity compiler version is used. It is always recommended to use the latest stable version of the compiler. Using old compiler forces to use the outdated openzeppelin libraries, which do not include the latest updates.

Contract: PerpetualWithdrawFunctions.sol

Function: _validateInputData

Recommendation: use the latest compilers version (i.e.: 0.8.11)

Status: Fixed (Revised Commit: 31720f4)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the audit cannot guarantee the explicit security of the audited smart contracts.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Appendix A. Smart Contracts Security Issues

Category	Test Name	Result	Details
SWC-136	Unencrypted Private Data On-Chain	Passed	
SWC-135	Code With No Effects	Passed	
SWC-134	Message call with hardcoded gas amount	Passed	
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed	
SWC-132	Unexpected Ether balance	Passed	
SWC-131	Presence of unused variables	Passed	
SWC-130	Right-To-Left-Override control character (U+202E)	Passed	
SWC-129	Typographical Error	Passed	
SWC-128	DoS With Block Gas Limit		Not applicable in Solidity 0.8.x
SWC-127	Arbitrary Jump with Function Type Variable	Passed	
SWC-126	Insufficient Gas Griefing		Not applicable in Solidity 0.8.x
SWC-125	Incorrect Inheritance Order	Passed	
SWC-124	Write to Arbitrary Storage Location	Passed	
SWC-123	Requirement Violation	Passed	
SWC-122	Lack of Proper Signature Verification	Passed	
SWC-121	Missing Protection against Signature Replay Attacks	Passed	
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed	
SWC-119	Shadowing State Variables Function Default Visibility	Passed	
SWC-118	Incorrect Constructor Name	Passed	
SWC-117	Signature Malleability	Passed	
SWC-116	Block values as a proxy for time	Passed	
SWC-115	Authorization through tx.origin	Passed	
SWC-114	Transaction Order Dependence	Passed	
SWC-113	DoS with Failed Call	Passed	

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

SWC-112	Delegatecall to Untrusted Callee	Passed	
SWC-111	Use of Deprecated Solidity Functions	Passed	
SWC-110	Assert Violation	Passed	
SWC-109	Uninitialized Storage Pointer		Not applicable in Solidity 0.8.x
SWC-108	State Variable Default Visibility	Passed	
SWC-107	Reentrancy	Passed	
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed	
SWC-105	Unprotected Ether Withdrawal	Passed	
SWC-104	Unchecked Call Return Value	Passed	
SWC-103	Floating Pragma	Passed	
SWC-102	Outdated Compiler Version	Passed	
SWC-101	Integer Overflow and Underflow		Not applicable in Solidity 0.8.x
SWC-100	Function Default Visibility	Passed	