CERTIK

Preliminary Comments

Sovryn 2

Nov 17th, 2021

Table of Contents

Summary

This report has been prepared for Sovryn to discover issues and vulnerabilities in the source code of the Sovryn 2 project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Sovryn 2
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/DistributedCollective/sovryn-perpetual-swap
Commit	f8805bc70e86bf75ac46c282ac44d848cb140537

Audit Summary

Delivery Date	Nov 17, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	2	2	0	0	0	0
Major	6	6	0	0	0	0
Medium	3	3	0	0	0	0
Minor	5	5	0	0	0	0
Informational	10	10	0	0	0	0
Discussion	2	2	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
CDT	cdf/CDFTable.sol	af94a4649ae553f6ecaa3c1b82511852af1b2cd0bfa26cabbbc29729baa9c6df
ICD	cdf/ICDFTable.sol	fd4362a6fd8eaf34271210397241f25c1b051604b891cf390f0a874edb1d1a0b
IPF	interface/IPriceFeedsExt.sol	cab1a871332d984cf507018c14a25ce02ea3ee4bfad7071b0015334913f0d085
IST	interface/IShareToken.sol	ff14b3a58b0ce0f690e0547f0a692450e83e9c2302c4eefb25e075c67802c5cf
ISF	interface/IShareTokenFactory.sol	4e614bb69e3b1230425aa7dd48d0192afa07756b76f518b27b5628eff4f743cf
ISO	interface/ISpotOracle.sol	3bf1b5b8996b1f7a43313d4d8a84722d26a482f0fc70c4b6a5c61e4e9a0f4efb
ABD	libraries/ABDKMath64x64.sol	41e6f66ba58b84fe02391789642842a4b17df9975b55d1b7e9030506ed9a8eb3
OFC	libraries/OrderFlags.sol	f71a1d0459af5bd97d1f02711019f8be44184efdcad8babefbe9649d6c62e118
QSC	libraries/QuickSort.sol	5abdf0205a15bdd930521263475253915099 85a3ef548858cf712b83604d795e
AOC	oracle/AbstractOracle.sol	df5756b14f8069b2ab89994b37dd93f00b5c8888467cca35cf1b3c9aca083c69
OFK	oracle/OracleFactory.sol	8ec527f8bdce51a0b7731d1b9f15911061999f188c4e69b9924e92ebe09aa2d8
OII	oracle/OracleInterfaceID.sol	2c949d0f209fbd540a4287782029293614fe1f12003520266d166e78a81af1f2
SOC	oracle/SpotOracle.sol	884ece4897c176ba0babf64c53b5360f1bc988ec201142a9574681c913831b97
PSC	perpetual/core/PerpStorage.sol	4e97d052cd6b60f7d24bef282f5944911495a159d2d86f2c80d60dc61750094e
PMP	perpetual/core/PerpetualManagerProxy.sol	4f7a6df6eafc1298a44d86a6b66face21a662b61c8584fc869a6f25cb0fd8b3e
AMM	perpetual/functions/AMMPerpLogic.sol	17e535ec0b669bfe2eed9cdc2336cf80a04146c2147f94b5cb7a852624db7af1

ID	File	SHA256 Checksum
PBF	perpetual/functions/PerpetualBaseFunctions.sol	51ba860f08275fc780bae9376eded5e3fca4df6fcc15a6e2e4bcfc0e14fda8eb
PRF	perpetual/functions/PerpetualRebalanceFunctions.sol	43cfa40a6ddef7c36fed1178f187fd2e9f9da3d149037d6981cc0e15f0ecaca9
PUF	perpetual/functions/PerpetualUpdateFunctions.sol	21e1acc41a5ea4796734843b0fb09b310014ed3460ba7111aaa483f906eaf5db
IAM	perpetual/interfaces/IAMMPerpLogic.sol	a18b156ada50f6e44953d145d3350275edfb0f963fd36b46f18383faf0ce7a34
IFL	perpetual/interfaces/IFunctionList.sol	986fff854c86ce515ce1bb3090220937a034aaf4cc26c22271842713007c954a
IPD	perpetual/interfaces/IPerpetualDepositManager.sol	fcddf00549f31ef99b2c2c9842f492de46359f30e962555dc63f0b0ab4be65b5
IPC	perpetual/interfaces/IPerpetualFactory.sol	95963fbeb4d1d220b8f99d1fc3103efbcef8314fb8aed7ac6f782e5207405441
IPG	perpetual/interfaces/IPerpetualGetter.sol	5a3466002e6b8ed85c5576e547bd26ceecf51cf056804d0aaded04b6ceb8b6bb
IPL	perpetual/interfaces/IPerpetualLiquidator.sol	186bf2d1948a0f75b31049b545f770bd1a1031a02389ca42f53364862ab10e02
IPM	perpetual/interfaces/IPerpetualManager.sol	9a1dd83f165496131895bbd50b8af9c09c6865549f9cc70fbde938b959ce3253
IPS	perpetual/interfaces/IPerpetualSettlement.sol	8d1022f99f7d2b621893c7e0e4c96447eefac10d6717f94d5e2828e91da24ab7
IPT	perpetual/interfaces/IPerpetualTradeLogic.sol	af9178c0b44674b3912fb912318563e09b728060de5897bffb82bd83fd5a8005
IPK	perpetual/interfaces/IPerpetualTradeManager.sol	281e3fd3614bf164b325f5743e79c8753dd7b2a7ea473849f0e5730f48cd33cf
IPP	perpetual/interfaces/IPerpetualTreasury.sol	a2ae96b3ade2331cbbaf90a9ebe1724e6ac0fdb0645590d70d616957c482e50b
IPU	perpetual/interfaces/IPerpetualUpdateLogic.sol	22ac9204aa0fd8153138f8720948734775768027ed7d3daf1024736e7843a0c5
IPW	perpetual/interfaces/IPerpetualWithdrawManager.sol	4555a15ed611dfee274296824e4ebe96d047e17d93cf7dc96f2b6b92e4786205

ID	File	SHA256 Checksum
ISV	perpetual/interfaces/ISOVLibraryEvents.sol	77326d7bb0de6786bd2cc6024007caccb8d0adef1dfd188ff866935ca87af1fa
PDM	perpetual/modules/PerpetualDepositManager.sol	bc79282f85000b826dd8d8362859966a9fa1792ecf4682dc6b79883cfc97653c
PFC	perpetual/modules/PerpetualFactory.sol	9eef66c152eecbd3b9d7d69e4212f8f8a557260fcba97c344a2219ddfe8388c2
PGC	perpetual/modules/PerpetualGetter.sol	065b716c6cc82a60706f140cc77780a000a403aaf2a06a9ffc1179032ab33c47
PLC	perpetual/modules/PerpetualLiquidator.sol	bba754b755ab6d5c39334dd93a9f72ab53475922e1ba15dcb7641ea4a94b1cf8
PSK	perpetual/modules/PerpetualSettlement.sol	c81bfccd43db9444e8096e962bef13d471e97dd801ccc55bc0018a59fb3d739e
PTL	perpetual/modules/PerpetualTradeLogic.sol	55614887b8b5f6f0530e4f2438ecb33185bce04465890099ceaa4a6304db6647
PTM	perpetual/modules/PerpetualTradeManager.sol	34a69e1ae2aae99c0f8aa5f1b1e535992bf0422fa6ad983b570806b23e8baff5
PTC	perpetual/modules/PerpetualTreasury.sol	4801e90afdc9b9a219cb9533f48199c00ef02434ec03c9b9352bfb88d73b52ed
PUL	perpetual/modules/PerpetualUpdateLogic.sol	b39762a88a78b0c7fe898385da1e04a96d8e98d4818413c8a97f4a89086e0af5
PWM	perpetual/modules/PerpetualWithdrawManager.sol	077dc4cfd4ae8b76c9af34877ac798afa6a418a07045c017e222f31c842ee874
STC	perpetual/token/ShareToken.sol	6e56fb853eba74c55f13562892f45105643498f5491b37569d0386058effbc6c
STF	perpetual/token/ShareTokenFactory.sol	5449dbdadcff9757dac0d50c28e556830f4218294a08a529cfcbe8216ada1af1

Executive Summary

It should be noted that the system design includes a number of economic arguments and assumptions. These were explored to the extent that they clarified the intention of the code base, but we did not audit the mechanism design itself. Note that financial models of blockchain protocols need to be resilient to attacks. It needs to pass simulations and verifications to guarantee the security of the overall protocol. The correctness of the financial model is not in the scope of the audit.

Additionally, as of the date of publishing, the contents of this document reflect the current understanding of known quality and security patterns regarding smart contracts and compilers. Given the size and complexity of the project, the findings detailed here are not to be considered exhaustive, and further testing and auditing are recommended after the issues covered are fixed.

The modules use the proxy pattern and can be upgraded through administrator actions.

The owner of `ShareToken` has the responsibility to notify users with the following capability:

- mint/burn any amount of token without any restriction.

The owner of `PerpetualFactory` has the responsibility to notify users with the following capability:

- one-time set the `shareTokenFactory` address through `initialize()`
- create a new liquidity pool through `createLiquidityPool()`
- create a perpetual through `createPerpetual()`
- set the `ammPerpLogic` address through `setAMMPerpLogic()`

The owner of `AMMPerpLogic` has the responsibility to notify users with the following capability:

- set the cumulative probability reference used in risk evaluation

The owner of `OracleFactory` has the responsibility to notify users with the following capability:

- deploy oracle contract for currency pair through `createOracle()`
- set oracle contract for currency pair through `addOracles()`
- set the given array of oracles as a route for the given currency pair through `addRoute()`

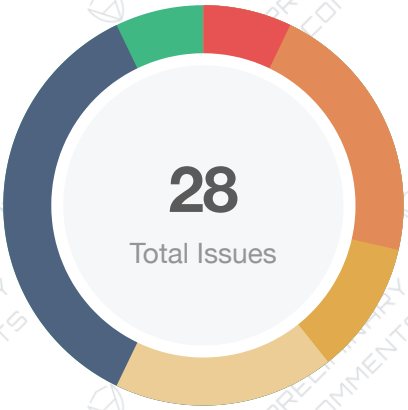
The owner of `CDFTable` has the responsibility to notify users with the following capability:

- add a reference to the cumulative probability table
- remove a route for the given currency pair through `removeRoute()`

The owner of `SpotOracle` has the responsibility to notify users with the following capability:

- set the market to "closed" through `setMarketClosed()`
- set the market to "terminated" through `setTerminated()`

Findings



Critical	2 (7.14%)
Major	6 (21.43%)
Medium	3 (10.71%)
Minor	5 (17.86%)
Informational	10 (35.71%)
Discussion	2 (7.14%)

ID	Title	Category	Severity	Status
GLOBAL-01	Potential Risk of `delegatecall`	Logical Issue	Medium	Pending
GLOBAL-02	Transparency of Parameters	Control Flow	Discussion	Pending
ABD-01	Redundant Statements	Volatile Code	Informational	Pending
ABD-02	Missing Error Messages	Coding Style	Informational	Pending
AMM-01	Division Before Multiplication	Mathematical Operations	Informational	Pending
AMM-02	Centralization Risk	**Centralization / Privilege**	**Major**	Pending
AMM-03	Function Visibility Optimization	Gas Optimization	Informational	Pending
CDT-01	Redundant Statements	Gas Optimization	Informational	Pending
CDT-02	Non-optimal Recursive Key Range	Logical Issue	Minor	Pending
CDT-03	Redundant Branch	Logical Issue	Minor	Pending
CDT-04	Centralization Risk	**Centralization / Privilege**	**Major**	Pending
OFK-01	Centralization Risk	**Centralization / Privilege**	**Major**	Pending
PBF-01	Division Before Multiplication	Mathematical Operations	Informational	Pending
PBF-02	Redundant Statements	Volatile Code	Informational	Pending
PBF-03	Inconsistent Conditional	Logical Issue	Discussion	Pending
PFC-01	Wrong Parameter Used	Logical Issue	Critical	Pending

ID	Title	Category	Severity	Status
PFC-02	Volatile Access	Control Flow	Medium	Pending
PFC-03	Centralization Risk	**Centralization / Privilege**	**Major**	Pending
PMP-01	Unused `internal` Function	Volatile Code	Minor	Pending
PRF-01	Incorrect `_famount` Sign	Logical Issue	Critical	Pending
PSC-01	Redundant Data Structure	Gas Optimization	Informational	Pending
PTL-01	Redundant Statements	Gas Optimization	Informational	Pending
PTL-02	Redundant Check for `traderAddr`	Logical Issue	Minor	Pending
PUF-01	Division Before Multiplication	Mathematical Operations	Informational	Pending
PUL-01	Lack of Access Control	Logical Issue	Medium	Pending
SOC-01	Third Party Dependencies	Volatile Code	Minor	Pending
SOC-02	Centralization Risk	**Centralization / Privilege**	**Major**	Pending
STC-01	Centralization Risk	**Centralization / Privilege**	**Major**	Pending

GLOBAL-01 | Potential Risk of `delegatecall`

Category	Severity	Location	Status
Logical Issue	Medium	Global	Pending

Description

DelegateCall, as the name implies, is a calling mechanism of how caller contract calls target contract function but when target contract executed its logic, the context is not on the user who executes caller contract but on caller contract. So all developers should be aware of the risk that the `target` address may do harm to the current contract. In addition, it may also cause function clashing. Refer to https://forum.openzeppelin.com/t/beware-of-the-proxy-learn-how-to-exploit-function-clashing/1070.

Recommendation

We advise the client to be careful with the function and only use it on credible contracts. We also advise the client to introduce the `Proxy` pattern from `openzeppelin` to avoid function clashing.

GLOBAL-02 | Transparency of Parameters

Category	Severity	Location	Status
Control Flow	Discussion	Global	Pending

Description

It appears that many key parameters used to margin rate and optimal fund sizes are customizable during initialization, including but not limited to:

- `CDFTable`
- `fTargetAMMPoolSize`
- `fMarginRateBeta`

Recommendation

We would like to inquire if any these parameters are not disclosed to the user and why.

ABD-01 | Redundant Statements

Category	Severity	Location	Status
Volatile Code	Informational	projects/sovryn2/libraries/ABDKMath64x64.sol (49d7330): 39	Pending

Description

The linked statements do not affect the functionality of the codebase and appear to be either leftovers from test code or older functionality.

Recommendation

We advise that they are removed to better prepare the code for production environments.

ABD-02 | Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	projects/sovryn2/libraries/ABDKMath64x64.sol (49d7330)	Pending

Description

The **require** can be used to check for conditions and throw an exception if the condition is not met. It is better to provide a string message containing details about the error that will be passed back to the caller.

Recommendation

We advise the client to add error messages in the `ABDKMath64x64` library.

AMM-01 | Division Before Multiplication

Category	Severity	Location	Status
Mathematical Operations	Informational	projects/sovryn2/perpetual/functions/AMMPerpLogic.sol (49d7330): 223, 104, 386	Pending

Description

Mathematical operations in the aforementioned line performs divisions before multiplications which can incur loss of precision.

Recommendation

We recommend applying multiplications before divisions.

AMM-02 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	projects/sovryn2/perpetual/functions/AMMPerpLogic.sol (49d7330) : 46	Pending

Description

In the contract `AMMPerpLogic`, the role `owner` has the authority over the following function:

- `setCDFTable()`

Any compromise to the `owner` account may allow the hacker to take advantage of this and

- set the cumulative probability reference used in risk evaluation and potentially manipulate the mark price

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

AMM-03 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	Informational	projects/sovryn2/perpetual/functions/AMMPerpLogic.sol (49d7330): 60, 104, 262, 302, 332, 360, 386	Pending

Description

The aforementioned functions are declared as `public`, contain array function arguments, and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

Recommendation

We advise that the functions' visibility specifiers are set to `external` and the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

CDT-01 | Redundant Statements

Category	Severity	Location	Status
Gas Optimization	Informational	projects/sovryn2/cdf/CDFTable.sol (49d7330): 8	Pending

Description

`ISpotOracle` and `console` are never used in contract `PerpetualTradeLogic` therefore it is unnecessary to import them. Besides `console` is imported in various contracts not limited to the linked ones.

These statements do not affect the functionality of the codebase and appear to be either leftover from test code or older functionality.

Recommendation

We advise that they be removed to better prepare the code for production environments.

CDT-02 | Non-optimal Recursive Key Range

Category	Severity	Location	Status
Logical Issue	Minor	projects/sovryn2/cdf/CDFTable.sol (49d7330): 115~116	Pending

Description

Excluding the special situation that `centerIndex == uiLength - 2`, `fCenterKey` would equal `fEndKey`, the right boundary of the central interval in which `_fKey` would be compared. In that case, the previous if branches would have included the circumstance `_fkey <= fCenter` and current code logic is redundant and add an additional interval to the next search range.

Recommendation

We advise the client to use `_fKey < fStartKey` instead.

CDT-03 | Redundant Branch

Category	Severity	Location	Status
Logical Issue	Minor	projects/sovryn2/cdf/CDFTable.sol (49d7330): 97~99	Pending

Description

It seems that the only situation for `centerIndex == 0` is when `uiStartIndex` is zero and `uiEndIndex` is one, which would fail the while conditional on line 95.

Recommendation

We would like to advise the client to remove this redundant `if` branch.

CDT-04 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	projects/sovryn2/cdf/CDFTable.sol (49d7330): 24	Pending

Description

In the contract `CDFTable`, the role `owner` has the authority over the following function:

- `addRows()`

Any compromise to the `owner` account may allow the hacker to take advantage of this and

- add a reference to the cumulative probability table to potentially manipulate risk evaluation

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

OFK-01 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	projects/sovryn2/oracle/OracleFactory.sol (49d7330): 39~49, 68, 106, 175	Pending

Description

In the contract `OracleFactory`, the role `owner` has the authority over the following function:

- `createOracle()`
- `addOracles()`
- `addRoute()`
- `removeRoute()`

Any compromise to the `owner` account may allow the hacker to take advantage of this and

- deploy oracle contract for currency pair through `createOracle()`
- set oracle contract for currency pair through `addOracles()`
- set the given array of oracles as a route for the given currency pair through `addRoute()`
- remove a route for the given currency pair through `removeRoute()`

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

PBF-01 | Division Before Multiplication

Category	Severity	Location	Status
Mathematical Operations	Informational	projects/sovryn2/perpetual/functions/PerpetualBaseFunctions.sol (49d7330): 245	Pending

Description

Mathematical operations in the aforementioned line performs divisions before multiplications which can incur loss of precision.

Recommendation

We recommend applying multiplications before divisions.

PBF-02 | Redundant Statements

Category	Severity	Location	Status
Volatile Code	Informational	projects/sovryn2/perpetual/functions/PerpetualBaseFunctions.sol (49d7330): 18~25	Pending

Description

The linked statements do not affect the functionality of the codebase and appear to be either leftovers from test code or older functionality.

Recommendation

We advise that they are removed to better prepare the code for production environments.

PBF-03 | Inconsistent Conditional

Category	Severity	Location	Status
Logical Issue	Discussion	projects/sovryn2/perpetual/functions/PerpetualBaseFunctions.sol (49d7330): 153	Pending

Description

It seems that the linked conditional should be `_fAmount<=0` in parallel to `_transferFromUserToVault()`.

Recommendation

We advise the client to review the functionality of underlying codes.

PFC-01 | Wrong Parameter Used

Category	Severity	Location	Status
Logical Issue	Critical	projects/sovryn2/perpetual/modules/PerpetualFactory.sol (49d7330): 174~175	Pending

Description

In `_createPerpetual()`, the parameters are first checked as items in argument arrays then assigned to respective slots in the perpetual structure. There is an error during assignment when `_baseParams[7]` is assigned twice to `perpetual.fLiquidationPenaltyRate` and `perpetual.fMinimalSpread`. Subsequently `_baseParams[8]` is wrongly assigned to `perpetual.fDustSizeQC` and `_baseParams[9]` is unused.

Recommendation

We advise the client to correct assignments as following:

```
1      perpetual.fMinimalSpread = _baseParams[8];
2      perpetual.fDustSizeQC = _baseParams[9];
```

PFC-02 | Volatile Access

Category	Severity	Location	Status
Control Flow	Medium	projects/sovryn2/perpetual/modules/PerpetualFactory.sol (49d7330): 209	Pending

Description

Currently anyone can initialize a liquidity pool through `runLiquidityPool()` which contradicts with the purpose of operator specified in the comments above.

Recommendation

We advise the review the functionality of `runLiquidityPool()`.

PFC-03 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	**Major**	projects/sovryn2/perpetual/modules/PerpetualFactory.sol (49d7330): 16, 31, 87, 224	Pending

Description

In the contract `PerpetualFactory`, the role `owner` has the authority over the following functions:

- `initialize()`
- `createLiquidityPool()`
- `createPerpetual()`
- `SetAMMPerpLogic()

Any compromise to the `owner` account may allow the hacker to take advantage of this and

- one-time set the `shareTokenFactory` address through `initialize()`
- create a new liquidity pool through `createLiquidityPool()`
- create a perpetual through `createPerpetual()`
- set the `ammPerpLogic` address through `setAMMPerpLogic()`

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

PMP-01 | Unused `internal` Function

Category	Severity	Location	Status
Volatile Code	Minor	projects/sovryn2/perpetual/core/PerpetualManagerProxy.sol (49d7330): 32	Pending

Description

This internal function is not called anywhere in the contract.

Recommendation

We advise the client to review the functionality of `_implementation` and remove it if unnecessary.

PRF-01 | Incorrect `_famount` Sign

Category	Severity	Location	Status
Logical Issue	Critical	projects/sovryn2/perpetual/functions/PerpetualRebalanceFunctions.sol (49d7330): 101	Pending

Description

The `_fAmount` is a negative value and should be converted to positive before use in `_transferFromPoolToAMMMargin()`.

Recommendation

We advise the client to check the signs for the parameter `_fAmount`.

PSC-01 | Redundant Data Structure

Category	Severity	Location	Status
Gas Optimization	Informational	projects/sovryn2/perpetual/core/PerpStorage.sol (49d7330): 31~35	Pending

Description

The `OrderType` enumeration is not used anywhere in the project.

Recommendation

We advise the client to remove the aforementioned code if there is no further plan to use this data type.

PTL-01 | Redundant Statements

Category	Severity	Location	Status
Gas Optimization	Informational	projects/sovryn2/perpetual/modules/PerpetualTradeLogic.sol (49d7330): 6, 11	Pending

Description

`ISpotOracle` and `console` are never used in contract `PerpetualTradeLogic` therefore it is unnecessary to import them. Besides `console` is imported in various contracts not limited to the linked ones.

These statements do not affect the functionality of the codebase and appear to be either leftover from test code or older functionality.

Recommendation

We advise that they be removed to better prepare the code for production environments.

PTL-02 | Redundant Check for `traderAddr`

Category	Severity	Location	Status
Logical Issue	Minor	projects/sovryn2/perpetual/modules/PerpetualTradeLogic.sol (49d7330): 246~248	Pending

Description

Assigning zero to `fTradeDir` is contradictory to the require check above.

Recommendation

We advise the client to remove aforementioned codes.

PUF-01 | Division Before Multiplication

Category	Severity	Location	Status
Mathematical Operations	Informational	projects/sovryn2/perpetual/functions/PerpetualUpdateFunctions.sol (49d7330): 29	Pending

Description

Mathematical operations in the aforementioned line performs divisions before multiplications which can incur loss of precision.

Recommendation

We recommend applying multiplications before divisions.

PUL-01 | Lack of Access Control

Category	Severity	Location	Status
Logical Issue	Medium	projects/sovryn2/perpetual/modules/PerpetualUpdateLogic.sol (49d7330): 9, 14, 20	Pending

Description

The following functions can be called by anyone to update the sensitive stats of the contract:

- `updateAMMTargetPoolSize()`
- `updateFundingAndPricesBefore()`
- `updateFundingAndPricesAfter()`

Recommendation

We recommend adding proper access control to this function or checking the status of initialization in the deployment process.

SOC-01 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Minor	projects/sovryn2/oracle/SpotOracle.sol (49d7330): 70, 77	Pending

Description

The contract is serving as the underlying entity to interact with third-party `PriceFeedsExt` protocols. The scope of the audit treats 3rd party entities as black boxes and assumes their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd parties can possibly create severe impacts.

Recommendation

We understand that the business logic of `SpotOracle` requires interaction with `PriceFeedsExt`. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

SOC-02 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	projects/sovryn2/oracle/SpotOracle.sol (49d7330): 37, 51	Pending

Description

In the contract `SpotOracle`, the role `owner` has the authority over the following function:

- `setMarketClosed()`
- `setTerminated()`
-

Any compromise to the `owner` account may allow the hacker to take advantage of this and

- set the market to "closed" through `setMarketClosed()`
- set the market to "terminated" through `setTerminated()`

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

STC-01 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	**Major**	projects/sovryn2/perpetual/token/ShareToken.sol (49d7330): 12, 16	Pending

Description

In the contract `ShareToken`, the role `owner` has the authority over the following function:

- `mint()`
- `burn()`

Any compromise to the `owner` account may allow the hacker to take advantage of this and mint/burn any amount of token without any restriction.

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.