Report

v. 1.0

Customer

Term Structure Labs

Smart Contract Audit

TermMax. Phase I

7th July 2025

Report prepared by

ABDK Consulting

Contents

1 Changelog

#	Date	Author	Description
0.1	07.07.25	A. Zveryanskaya	Initial Draft
0.2	07.07.25	A. Zveryanskaya	Minor revision
1.0	07.07.25	A. Zveryanskaya	Release

2 Introduction

The following document provides the result of the audit performed by ABDK Consulting (Mikhail Vladimirov and Dmitry Khovratovich) at the customer request. The audit goal is a general review of the smart contracts structure, critical/major bugs detection and issuing the general recommendations.

Term Structure is a decentralized bond protocol powered by zkTrue-up, a platform that enables peer-to-peer lending and borrowing with fixed interest rates.

3 Project scope

We were asked to review:

- Original Code
- Code with Fixes

Files:

extensions/pendle/			
	PendleHelper.sol		
lib/			
	OnlyProxyCall.sol	TransferUtilsV2.sol	
router/			
	IERC20SwapAdapter.sol		
router/swapAdapters/			
	ERC20 SwapAdapterV2.sol	ERC4626 VaultAdapterV2.sol	Kyberswap V2AdapterV2.sol
	OdosV2AdapterV2.sol	PendleSwapV3AdapterV2.sc	UniswapV3AdapterV2.sol

4 Methodology

The methodology is not a strict formal procedure, but rather a selection of methods and tactics combined differently and tuned for each particular project, depending on the project structure and technologies used, as well as on client expectations from the audit.

- **General Code Assessment**. The code is reviewed for clarity, consistency, style, and for whether it follows best code practices applicable to the particular programming language used. We check indentation, naming convention, commented code blocks, code duplication, confusing names, confusing, irrelevant, or missing comments etc. At this phase we also understand overall code structure.
- **Entity Usage Analysis**. Usages of various entities defined in the code are analysed. This includes both: internal usages from other parts of the code as well as potential external usages. We check that entities are defined in proper places as well as their visibility scopes and access levels are relevant. At this phase, we understand overall system architecture and how different parts of the code are related to each other.
- **Access Control Analysis**. For those entities, that could be accessed externally, access control measures are analysed. We check that access control is relevant and done properly. At this phase, we understand user roles and permissions, as well as what assets the system ought to protect.
- **Code Logic Analysis**. The code logic of particular functions is analysed for correctness and efficiency. We check if code actually does what it is supposed to do, if that algorithms are optimal and correct, and if proper data types are used. We also make sure that external libraries used in the code are up to date and relevant to the tasks they solve in the code. At this phase we also understand data structures used and the purposes they are used for.

We classify issues by the following severity levels:

- **Critical issue** directly affects the smart contract functionality and may cause a significant loss.
- **Major issue** is either a solid performance problem or a sign of misuse: a slight code modification or environment change may lead to loss of funds or data. Sometimes it is an abuse of unclear code behavior which should be double checked.
- **Moderate issue** is not an immediate problem, but rather suboptimal performance in edge cases, an obviously bad code practice, or a situation where the code is correct only in certain business flows.
- **Recommendations** contain code style, best practices and other suggestions.

5 Our findings

We found 1 critical, 6 major, and a few less important issues. All identified Critical and Major issues have been fixed.

Fixed 6 out of 6 issues

6 Critical Issues

CVF-1 FIXED

- **Category** Flaw
- **Source** OnlyProxyCall.sol

Description This check doesn't prevent direct calls.

Recommendation The check should be: require (address(this) != addressThis, Only-CallableThroughProxy());

```
17 require(msg.sender != address(this), OnlyCallableThroughProxy());
```

7 Major Issues

CVF-2 FIXED

- **Category** Unclear behavior
- **Source** ERC20SwapAdapterV2.sol

Description This contract looks like a mock without any actual logic.

Recommendation Implement the logic or clearly identify this contract as a mock.

Client Comment *This contract is an abstract base for ERC20 swap adapters in the TermMax protocol.I add the comments.*

```
12 * @notice This contract facilitates ERC20 token swaps with
      ↪ additional features.
```

CVF-3 FIXED

- **Category** Flaw
- **Source** ERC4626VaultAdapterV2.sol

Description This value should be rounded up. Currently, the scaled minimum output amount could be zero for a non-zero input amount.

```
33 minTokenOut = (minTokenOut * amount) / inAmount;
```

CVF-5 FIXED

- **Category** Flaw
- **Source** OdosV2AdapterV2.sol

Description This value should be rounded up.

```
60 tokenInfo.outputMin = (tokenInfo.outputMin * amountIn) / tokenInfo.
      ↪ inputAmount;
```

CVF-6 FIXED

- **Category** Flaw
- **Source** PendleSwapV3AdapterV2.sol

Description This value should be rounded up.

```
39 minTokenOut = (minTokenOut * amount) / inAmount;
```

CVF-7 FIXED

- **Category** Flaw
- **Source** UniswapV3AdapterV2.sol

Description This value should be rounded up.

```
32 amountOutMinimum = (amountOutMinimum * amount) / inAmount;
```

8 Moderate Issues

CVF-4 INFO

- **Category** Suboptimal
- **Source** IERC20SwapAdapter.sol

Description This function doesn't allow specifying the minimum output amount, which makes it hard to use this function safely.

Recommendation Add the "minTokenOutAmt" argument.

Client Comment *This parameter is implemented independently in each adapter. I want to avoid duplication of parameters because the swap data will contain this data.*

```
16 function swap(address receipient, address tokenIn, address tokenOut,
      ↪  uint256 tokenInAmt, bytes memory swapData)
```

CVF-8 FIXED

- **Category** Suboptimal
- **Source** ERC4626VaultAdapterV2.sol

Description This is not required when redeeming own tokens.

Recommendation Remove this call.

```
42 tokenIn.safeIncreaseAllowance(address(tokenIn), amount);
```

CVF-9 FIXED

- **Category** Overflow/Underflow
- **Source** OdosV2AdapterV2.sol

Description Phantom overflow is possible here. i.e. a situation when the final calculation result would fit into the destination type, while certain intermediary calculation overflows.

Recommendation Use the "mulDiv" function.

```
59 tokenInfo.outputQuote = (tokenInfo.outputQuote * amountIn) /
      ↪ tokenInfo.inputAmount;
60 tokenInfo.outputMin = (tokenInfo.outputMin * amountIn) / tokenInfo.
      ↪ inputAmount;
```

CVF-10 INFO

- **Category** Flaw
- **Source** PendleHelper.sol

Description Not all fields of “input” are initialized.

Recommendation Initialize all the fields and let the compiler check that all fields were initialized like this: return TokenInput (...);

Client Comment *The purpose of this operation is to avoid version compatibility issues with Pendle here.*

```
24 input.tokenIn = tokenIn;
   input.netTokenIn = netTokenIn;
   input.tokenMintSy = tokenIn;
```

CVF-11 INFO

- **Category** Flaw
- **Source** PendleHelper.sol

Description Not all fields of “output” are initialized.

Recommendation Initialize all the fields and let the compiler check that all fields were initialized like this: return TokenOutput (...);”

Client Comment *The purpose of this operation is to avoid version compatibility issues with Pendle here.*

```
36 output.tokenOut = tokenOut;
   output.minTokenOut = minTokenOut;
   output.tokenRedeemSy = tokenOut;
```

CVF-12 FIXED

- **Category** Unclear behavior
- **Source** TransferUtilsV2.sol

Description It is unclear why transferring exactly max uint256 token is prohibited.

Recommendation Remove these checks or clearly explain why they are needed.

Client Comment *It is to check the transferring uint256 maxium of atoken. I add the comments.*

```
13 if (value == type(uint256).max) {
       revert CanNotTransferUintMax();
```

```
23 if (value == type(uint256).max) {
       revert CanNotTransferUintMax();
```

CVF-13 FIXED

- **Category** Suboptimal
- **Source** TransferUtilsV2.sol

Description These checks optimise marginal use cases, but make most common use cases more expensive.

Recommendation Revisit these check to ensure they are really necessary.

```
16 if (from == to || value == 0) {
```

```
26 if (to == address(this) || value == 0) {
```

```
33 if (value == 0 || spender == address(this)) {
```

```
40 if (value == 0 || spender == address(this)) {
```

```
47 if (spender == address(this)) {
```

CVF-14 FIXED

- **Category** Overflow/Underflow
- **Source** UniswapV3AdapterV2.sol

Description Phantom overflow is possible here, i.e. a situation when the final calculation result would fit into the destination type, while certain intermediary calculation overflows.

Recommendation Use the “mulDiv” function.

```
32 amountOutMinimum = (amountOutMinimum * amount) / inAmount;
```

9 Recommendations

CVF-15 FIXED

- **Category** Procedural
- **Source** ERC20SwapAdapterV2.sol

Description Consider specifying as “^0.8.0” unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-16 FIXED

- **Category** Unclear behavior
- **Source** ERC20SwapAdapterV2.sol

Description It is unclear how this error could be triggered, as there is no way to specify the minimum output amount for a swap.

Recommendation Remove this error or clearly explain where it could be triggered.

```
20 /// @notice Error for less than min token out
   error LessThanMinTokenOut(uint256 actual, uint256 expected);
```

CVF-17 FIXED

- **Category** Unclear behavior
- **Source** ERC20SwapAdapterV2.sol

Description It is unclear how this error could be triggered as there is no way to perform a swap without specifying the exact input amount.

Recommendation Remove this error of clearly explain where it could be triggered.

```
23 error ExceedMaxTokenIn(uint256 actual, uint256 expected);
```

CVF-18 FIXED

- **Category** Unclear behavior
- **Source** ERC20SwapAdapterV2.sol

Description This function looks like it has to be implemented in inherited contracts, however it has an empty body, so Compiler won't require this function to be overridden.

Recommendation Declare this function with no body, rather than with an empty body.

```
37 function _swap(address receipient, IERC20 tokenIn, IERC20 tokenOut,
      ↪ uint256 tokenInAmt, bytes memory swapData)

41 {}
```

CVF-19 FIXED

- **Category** Procedural
- **Source** ERC4626VaultAdapterV2.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-20 FIXED

- **Category** Documentation
- **Source** ERC4626VaultAdapterV2.sol

Description It is a good practice to put a comment into an empty block to explain why the block is empty.

```
21 constructor() {}
```

CVF-21 FIXED

- **Category** Suboptimal
- **Source** ERC4626VaultAdapterV2.sol

Description This constructor is redundant, as it contains no logic.

Recommendation Remove it.

```
21 constructor() {}
```

CVF-22 FIXED

- **Category** Procedural
- **Source** IERC20SwapAdapter.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-23 INFO

- **Category** Bad datatype
- **Source** IERC20SwapAdapter.sol

Description The type for the token arguments should be "IERC20".

Client Comment *The purpose of changing to address is to facilitate the adaptation of different types of token logic*

```
16 function swap(address receipient, address tokenIn, address tokenOut,
   ↪  uint256 tokenInAmt, bytes memory swapData)
```

CVF-24 INFO

- **Category** Suboptimal
- **Source** IERC20SwapAdapter.sol

Description This function should emit some event and this event should be declared in this interface.

Client Comment *There is currently no need to track the swap chain, and for the purpose of saving gas consumption, we are not considering adding events for the time being.*

```
16 function swap(address receipient, address tokenIn, address tokenOut,
   ↪  uint256 tokenInAmt, bytes memory swapData)
```

CVF-25 FIXED

- **Category** Procedural
- **Source** KyberswapV2AdapterV2.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-26 INFO

- **Category** Procedural
- **Source** KyberswapV2AdapterV2.sol

Description This interface should be moved into a separate file named "IKyberScalingHelper.sol".

Client Comment *We don't consider doing this because no other contracts will use it.*

```
7 interface IKyberScalingHelper {
```

CVF-27 FIXED

- **Category** Documentation
- **Source** KyberswapV2AdapterV2.sol

Description The semantics of this function is unclear.

Recommendation Explain in a documentation comment.

```
8  function getScaledInputData(bytes calldata inputData, uint256
       ↪ newAmount)
```

```
11     returns (bool isSuccess, bytes memory data);
```

CVF-28 FIXED

- **Category** Suboptimal
- **Source** KyberswapV2AdapterV2.sol

Description This error could be made more useful by including the original error into it.

```
22 error KyberScalingFailed();
```

CVF-29 INFO

- **Category** Bad datatype
- **Source** KyberswapV2AdapterV2.sol

Description The type for this variable should be more specific.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
24 address public immutable router;
```

CVF-30 INFO

- **Category** Bad datatype
- **Source** KyberswapV2AdapterV2.sol

Description The type for this variable should be “IKyberScalingHelper”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
25 address public immutable KYBER_SCALING_HELPER;
```

CVF-31 INFO

- **Category** Bad datatype
- **Source** KyberswapV2AdapterV2.sol

Description The type for the “router_” argument should be more specific.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
27 constructor(address router_, address scalingHelper_) {
```

CVF-32 INFO

- **Category** Bad datatype
- **Source** KyberswapV2AdapterV2.sol

Description The type for the “scalingHelper_” argument should be “IKyberScalingHelper”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
27 constructor(address router_, address scalingHelper_) {
```

CVF-33 FIXED

- **Category** Documentation
- **Source** KyberswapV2AdapterV2.sol

Description The address in the comment is confusing.

Recommendation Remove the comment.

```
29 KYBER_SCALING_HELPER = scalingHelper_; // 0
      ↪ x2f577A41BeC1BE1152AeEA12e73b7391d15f655D
```

CVF-34 FIXED

- **Category** Procedural
- **Source** OdosV2AdapterV2.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-35 INFO

- **Category** Procedural
- **Source** OdosV2AdapterV2.sol

Description This interface should be declared in a separate file named "IOdos-RouterV2.sol".

Client Comment *We don't consider doing this because no other contracts will use it.*

```
6 interface IOdosRouterV2 {
```

CVF-36 FIXED

- **Category** Bad naming
- **Source** OdosV2AdapterV2.sol

Description A struct name are usually starts with a capital letter.

```
7 struct swapTokenInfo {
```

CVF-37 INFO

- **Category** Bad datatype
- **Source** OdosV2AdapterV2.sol

Description The type for these fields should be “IERC20”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
8  address inputToken;
```

```
11 address outputToken;
```

CVF-38 FIXED

- **Category** Suboptimal
- **Source** OdosV2AdapterV2.sol

Description This error could be made more useful by adding certain parameters into it.

```
30 error InvalidOutputToken();
```

CVF-39 INFO

- **Category** Bad datatype
- **Source** OdosV2AdapterV2.sol

Description The argument type should never be “IOdosRouterV2”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
34 constructor(address router_) {
```

CVF-40 FIXED

- **Category** Suboptimal
- **Source** OdosV2AdapterV2.sol

Description This check makes the “outputToken” field redundant.

```
53 if (tokenInfo.outputToken != address(tokenOut)) {
```

CVF-41 FIXED

- **Category** Readability
- **Source** OdosV2AdapterV2.sol

Description Brackets are redundant.

```
59 tokenInfo.outputQuote = (tokenInfo.outputQuote * amountIn) /
      ↪ tokenInfo.inputAmount;
60 tokenInfo.outputMin = (tokenInfo.outputMin * amountIn) / tokenInfo.
      ↪ inputAmount;
```

CVF-42 FIXED

- **Category** Procedural
- **Source** OnlyProxyCall.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular file.

```
2 pragma solidity ^0.8.27;
```

CVF-43 INFO

- **Category** Procedural
- **Source** OrderManagerV2.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular version.

Client Comment *This is optional, we only consider using "^0.8.0" in lib or interface.*

```
2 pragma solidity ^0.8.27;
```

CVF-44 FIXED

- **Category** Procedural
- **Source** PendleHelper.sol

Description Consider specifying as "^0.8.0" unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-45 INFO

- **Category** Procedural
- **Source** PendleHelper.sol

Description We didn't review these files.

Client Comment *This is the Interface from pendle and can be excluded.*

```
4 import "@pendle/core-v2/contracts/interfaces/IPAllActionV3.sol";
  import "@pendle/core-v2/contracts/interfaces/IPMarket.sol";
```

CVF-46 INFO

- **Category** Unclear behavior
- **Source** PendleHelper.sol

Description It is unclear where the "SwapData" type is imported from.

Recommendation Import explicitly as: import { SwapData } from "@pendle/core-v2/contracts/router/swap-aggregator/IPSwapAggregator.sol";

Client Comment *These types are all derived from pendle and we do not consider making further declarations.*

```
9 SwapData public emptySwap;
```

CVF-47 INFO

- **Category** Unclear behavior
- **Source** PendleHelper.sol

Description It is unclear where the “LimitOrderData”, “ApproxParams”, “TokenInput”, and “TokenOutput” types are imported from.

Recommendation Import explicitly as: import { ApproxParams, LimitOrderData, TokenInput, TokenOutput } from "@pendle/core-v2/contracts/interfaces /IPAllActionTypeV3.sol";

Client Comment *These types are all derived from pendle and we do not consider making further declarations.*

```
12 LimitOrderData public emptyLimit;
```

```
15 ApproxParams public defaultApprox = ApproxParams(0, type(uint256).
   ↪ max, 0, 256, 1e14);
```

```
22     returns (TokenInput memory input)
```

```
34     returns (TokenOutput memory output)
```

CVF-48 INFO

- **Category** Bad naming
- **Source** PendleHelper.sol

Description The semantics of the field values are unclear.

Recommendation Specify field names like this: ApproxParams({guessMin: 0, guessMax: type(uint256).max, guessOfffchain: 0, maxIteration: 256, eps: 1e14});

Client Comment *These types are all derived from pendle and we do not consider making further declarations.*

```
15 ApproxParams public defaultApprox = ApproxParams(0, type(uint256).
   ↪ max, 0, 256, 1e14);
```

CVF-49 INFO

- **Category** Bad datatype
- **Source** PendleHelper.sol

Description The type for the “tokenIn” argument should be “IERC20”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
19 function createTokenInputStruct(address tokenIn, uint256 netTokenIn)
```

CVF-50 INFO

- **Category** Bad datatype
- **Source** PendleHelper.sol

Description The type for the “tokenOut” argument should be “IERC20”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
31 function createTokenOutputStruct(address tokenOut, uint256
   ↪ minTokenOut)
```

CVF-51 FIXED

- **Category** Procedural
- **Source** PendleSwapV3AdapterV2.sol

Description Consider specifying as “^0.8.0” unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-52 INFO

- **Category** Procedural
- **Source** PendleSwapV3AdapterV2.sol

Description We didn't review these files.

Client Comment *This is the Interface from pendle and can be excluded.*

```
4 import {IPAllActionV3} from "@pendle/core-v2/contracts/interfaces/
    ↪ IPAllActionV3.sol";
  import {IPMarket, IPPrincipalToken, IPYieldToken} from "@pendle/core
    ↪ -v2/contracts/interfaces/IPMarket.sol";
```

CVF-53 INFO

- **Category** Bad datatype
- **Source** PendleSwapV3AdapterV2.sol

Description The argument type should be "IPAllActionV3".

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
18 constructor(address router_) {
```

CVF-54 INFO

- **Category** Bad datatype
- **Source** PendleSwapV3AdapterV2.sol

Description The type for the first decoded value should be "IPMarket" rather than "address".

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
30 abi.decode(swapData, (address, uint256, uint256));
```

CVF-55 FIXED

- **Category** Readability
- **Source** PendleSwapV3AdapterV2.sol

Description Brackets are redundant.

```
39 minTokenOut = (minTokenOut * amount) / inAmount;
```

CVF-56 FIXED

- **Category** Overflow/Underflow
- **Source** PendleSwapV3AdapterV2.sol

Description Phantom overflow is possible here, i.e. a situation when the final calculation result would fit into the destination type, while certain intermediary calculation overflows.

Recommendation Use the “mulDiv” function.

```
39 minTokenOut = (minTokenOut * amount) / inAmount;
```

CVF-57 FIXED

- **Category** Procedural
- **Source** UniswapV3AdapterV2.sol

Description Consider specifying as “^0.8.0” unless there is something special regarding this particular version.

```
2 pragma solidity ^0.8.27;
```

CVF-58 INFO

- **Category** Bad datatype
- **Source** UniswapV3AdapterV2.sol

Description The argument type should ge “ISwapRouter”.

Client Comment *Thanks for your suggestion, we will not change it for now because it is optional.*

```
16 constructor(address router_) {
```

CVF-59 FIXED

- **Category** Readability
- **Source** UniswapV3AdapterV2.sol

Description Brackets are redundant.

```
32 amountOutMinimum = (amountOutMinimum * amount) / inAmount;
```

ABDK
Consulting

About us

Established in 2016, is a leading service provider in the space of blockchain development and audit. It has contributed to numerous blockchain projects, and co-authored some widely known blockchain primitives like Poseidon hash function.

The ABDK Audit Team, led by Mikhail Vladimirov and Dmitry Khovratovich, has conducted over 40 audits of blockchain projects in Solidity, Rust, Circom, C++, JavaScript, and other languages.

Contact

Email

dmitry@abdkconsulting.com

Website

abdk.consulting

Twitter

twitter.com/ABDKconsulting

LinkedIn

linkedin.com/company/abdk-consulting