Pendle MarketV6 Audit Report

Oct 23, 2025

Table of Contents

Summary

This report has been prepared for MarketV6 smart contract, to discover issues and vulnerabilities in the source code of their Smart Contract 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.

Overview

Project Summary

Project Name	**MarketV6**
Codebase	**https://github.com/pendle-finance/pendle-core-v2**
Commit	**9fe9332079fc9960e4429bac1f6dee0fb3ecda60**
Language	**Solidity**

Audit Summary

Delivery Date	**Oct 23, 2025**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**5**

[WP-M1] In `PendleMarketFactoryV6Upg.initialize(_owner, ...)`, when `_owner` is not `msg.sender`, the `setTreasuryAndFeeReserve()` call will revert due to the `onlyOwner` modifier.

Medium

Issue Description

`PendleMarketFactoryV6Upg` may need to add a private/internal `_setTreasuryAndFeeReserve()` function that doesn't require owner permission in `initialize()`.

https://github.com/pendle-finance/pendle-core-v2/blob/c34284095a2c800d9e3797e4d8ae5affc1fda3f0/contracts/core/Market/PendleMarketFactoryV6Upg.sol#L64-L67

```
64      function initialize(address _owner, address _treasury, uint8
    _reserveFeePercent) external initializer {
65          __BoringOwnableV2_init(_owner);
66          setTreasuryAndFeeReserve(_treasury, _reserveFeePercent);
67      }
```

https://github.com/pendle-finance/pendle-core-v2/blob/c34284095a2c800d9e3797e4d8ae5affc1fda3f0/contracts/core/Market/PendleMarketFactoryV6Upg.sol#L120-L129

```
120     function setTreasuryAndFeeReserve(address newTreasury, uint8
    newReserveFeePercent) public onlyOwner {
    @@ 121,128 @@
129     }
```

PendleYieldContractFactoryUpg has similar issues.

https://github.com/pendle-finance/pendle-core-v2/blob/c34284095a2c800d9e3797e4d8ae5affc1fda3f0/contracts/core/YieldContracts/PendleYieldContractFactoryUpg.sol

```
@@ 1,39 @@

/// @dev If this contract is ever made upgradeable, please pay attention to the numContractDeployed variable
contract PendleYieldContractFactoryUpg is BoringOwnableUpgradeableV2, IPYieldContractFactory {

@@ 43,89 @@

    function initialize(
        uint96 _expiryDivisor,
        uint128 _interestFeeRate,
        uint128 _rewardFeeRate,
        address _treasury,
        address _owner
    ) external initializer {
        __BoringOwnableV2_init(_owner);
        setExpiryDivisor(_expiryDivisor);
        setInterestFeeRate(_interestFeeRate);
        setRewardFeeRate(_rewardFeeRate);
        setTreasury(_treasury);
    }

@@ 105,166 @@

    function setExpiryDivisor(uint96 newExpiryDivisor) public onlyOwner {

@@ 169,172 @@

    }

    function setInterestFeeRate(uint128 newInterestFeeRate) public onlyOwner {

@@ 176,180 @@

    }

    function setRewardFeeRate(uint128 newRewardFeeRate) public onlyOwner {

@@ 184,188 @@

    }

    function setTreasury(address newTreasury) public onlyOwner {
```

```
@@ 192,195 @@
196       }
197   }
```

Status

ⓘ Acknowledged

[WP-M2] `PendleLpOracleLib` is incompatible with older markets that don't have the `market.reentrancyGuardEntered()` function

Medium

Issue Description

Consider these options:

- Use `try market.VERSION() returns (uint256 marketVersion)` and only call `market.reentrancyGuardEntered()` for markets where `marketVersion >= 6`. (It's more secure to use `reentrancyGuardEntered()` when it is available)
- Or keep the old `PendleLpOracleLib` and add a new `PendleLpOracleLibV2`. (Saves some gas)

https://github.com/pendle-finance/pendle-core-v2/blob/c34284095a2c800d9e3797e4d8ae5affc1fda3f0/contracts/oracles/PtYtLpOracle/PendleLpOracleLib.sol#L1-L87

```
@@ 1,4 @@

5
6   library PendleLpOracleLib {
@@ 7,42 @@

43
44      function _getLpToAssetRateRaw(
45          IPMarket market,
46          uint32 duration,
47          uint256 pyIndex
48      ) private view returns (uint256 lpToAssetRateRaw) {
49          require(market.reentrancyGuardEntered() == false, "ReentrancyGuard: reentrant call");
@@ 50,72 @@
73      }
74
@@ 75,86 @@
87  }
```

Status

✓ Fixed

[WP-I3] Outdated comment about PendleYieldContractFactory's upgradeability

Informational

Issue Description

In the old code when this comment was added, PT and YT were deployed using `create`, and their addresses depended on `numContractDeployed`.

In the current version, PT and YT are deployed using `create2`, and their addresses only depend on factory address, `salt`, and `code`, not on `numContractDeployed`.

```
41  /// @dev If this contract is ever made upgradeable, please pay attention to the
    numContractDeployed variable
42  contract PendleYieldContractFactoryUpg is BoringOwnableUpgradeableV2,
    IPYieldContractFactory {
    @@ 43,101 @@
102
103     /**
104      * @notice Create a pair of (PT, YT) from any SY and valid expiry. Anyone can
    create a yield contract
105      * @dev It's intentional to make expiry an uint32 to guard against fat fingers.
    uint32.max is year 2106
106      */
107     function createYieldContract(
108         address SY,
109         uint32 expiry,
110         bool doCacheIndexSameBlock
111     ) external returns (address PT, address YT) {
112         if (MiniHelpers.isTimeInThePast(expiry) || expiry % expiryDivisor != 0)
    revert Errors.YCFactoryInvalidExpiry();
113
114         if (getPT[SY][expiry] != address(0)) revert
    Errors.YCFactoryYieldContractExisted();
115
116         IStandardizedYield _SY = IStandardizedYield(SY);
117
118         (, , uint8 assetDecimals) = _SY.assetInfo();
119
```

```
120         string memory syCoreName = _stripSYPrefix(_SY.name());
121         string memory syCoreSymbol = _stripSYPrefix(_SY.symbol());
122 
123         PT = Create2.deploy(
124             0,
125             bytes32(block.chainid),
126             abi.encodePacked(
127                 type(PendlePrincipalToken).creationCode,
128                 abi.encode(
129                     SY,
130                     PT_PREFIX.concat(syCoreName, expiry, " "),
131                     PT_PREFIX.concat(syCoreSymbol, expiry, "-"),
132                     assetDecimals,
133                     expiry
134                 )
135             )
136         );
137 
138         YT = BaseSplitCodeFactory._create2(
139             0,
140             bytes32(block.chainid),
141             abi.encode(
142                 SY,
143                 PT,
144                 YT_PREFIX.concat(syCoreName, expiry, " "),
145                 YT_PREFIX.concat(syCoreSymbol, expiry, "-"),
146                 assetDecimals,
147                 expiry,
148                 doCacheIndexSameBlock
149             ),
150             ytCreationCodeContractA,
151             ytCreationCodeSizeA,
152             ytCreationCodeContractB,
153             ytCreationCodeSizeB
154         );
155 
156         IPPrincipalToken(PT).initialize(YT);
157 
    @@ 158,163 @@
164     }
165 
```

```
@@ 166,201 @@
}
```

```
/// @dev If this contract is ever made upgradeable, please pay attention to the numContractDeployed variable
contract PendleYieldContractFactory is BoringOwnableUpgradeableV2, IPYieldContractFactory {
@@ 43,198 @@
}
```

https://github.com/pendle-finance/pendle-core-v2/blob/5fba212c784ddda25a0711822dff0a1d2f9c8afa/contracts/core/PendleYieldContractFactory.sol#L37-L148

```
/// @dev If this contract is ever made upgradeable, please pay attention to the numContractDeployed variable
contract PendleYieldContractFactory is PermissionsV2Upg, MiniDeployer, IPYieldContractFactory {
@@ 39,48 @@

    /// number of contracts that have been deployed from this address
    /// must be increased everytime a new contract is deployed
    uint256 public numContractDeployed;

@@ 54,75 @@

    /**
     * @notice Create a pair of (PT, YT) from any SCY and valid expiry. Anyone can create a yield contract
     */
    function createYieldContract(address SCY, uint256 expiry)
        external
        returns (address PT, address YT)
    {
        require(expiry > block.timestamp, "invalid expiry");

        require(expiry % expiryDivisor == 0, "must be multiple of divisor");

```

```
88          require(getPT[SCY][expiry] == address(0), "PT_EXISTED");
89
90          ISuperComposableYield _SCY = ISuperComposableYield(SCY);
91
92          (, , uint8 assetDecimals) = _SCY.assetInfo();
93
94          address predictedPTAddress = LibRLP.computeAddress(address(this),
    ++numContractDeployed);
95          address predictedYTAddress = LibRLP.computeAddress(address(this),
    ++numContractDeployed);
96
97          PT = address(
98              new PendlePrincipalToken(
99                  SCY,
100                 predictedYTAddress,
101                 PT_PREFIX.concat(_SCY.name(), expiry, " "),
102                 PT_PREFIX.concat(_SCY.symbol(), expiry, "-"),
103                 assetDecimals,
104                 expiry
105             )
106         );
107
108         require(PT == predictedPTAddress, "internal error 1");
109
110         YT = _deployWithArgs(
111             pendleYtCreationCodePointer,
112             abi.encode(
113                 SCY,
114                 predictedPTAddress,
115                 YT_PREFIX.concat(_SCY.name(), expiry, " "),
116                 YT_PREFIX.concat(_SCY.symbol(), expiry, "-"),
117                 assetDecimals,
118                 expiry
119             )
120         );
121
122         require(YT == predictedYTAddress, "internal error 2");
123

    @@ 124,129 @@

130     }
131
```

```
@@ 132,147 @@
148 }
```

https://github.com/pendle-finance/pendle-core-v2/blob/5fba212c784ddda25a0711822dff0a1d2f9c8afa/contracts/libraries/helpers/MiniDeployer.sol#L11-L21

```
11      function _deployWithArgs(address creationCodePointer, bytes memory args)
12          internal
13          returns (address deployedContract)
14      {
15          require(creationCodePointer != address(0), "zero pointer");
16          bytes memory finalCreationCode =
    abi.encodePacked(SSTORE2.read(creationCodePointer), args);
17          assembly {
18              deployedContract := create(0, add(finalCreationCode, 32),
    mload(finalCreationCode))
19          }
20          require(deployedContract != address(0), "deployment failed");
21      }
```

Status

✓ Fixed

[WP-I4] `PendleGaugeController` may need to be upgraded to add `marketFactory5` to support markets created by the new `PendleMarketFactoryV6Upg`

Informational

Issue Description

By the way, a new `PendlePoolDeployHelperV2` contract need to be deployed with parameters set to the new marketFactory and new yieldContractFactory addresses.

https://github.com/pendle-finance/pendle-core-v2/blob/4b2ed578b712087e27e8f81a6860e2c02c7cef39/contracts/LiquidityMining/GaugeController/PendleGaugeControllerBaseUpg.sol#L42-L81

```
42      IPMarketFactory public immutable marketFactory;
43      IPMarketFactory public immutable marketFactory2;
44      IPMarketFactory public immutable marketFactory3;
45      IPMarketFactory public immutable marketFactory4;
46
    @@ 47,48 @@
49
50      mapping(address => bool) public isValidMarket;
51      uint256[99] private __gap;
52
53      modifier onlyPendleMarket() {
54          if (isValidMarket[msg.sender]) {
55              _;
56          } else if (
57              marketFactory.isValidMarket(msg.sender) ||
58              marketFactory2.isValidMarket(msg.sender) ||
59              marketFactory3.isValidMarket(msg.sender) ||
60              marketFactory4.isValidMarket(msg.sender)
61          ) {
62              isValidMarket[msg.sender] = true;
63              _;
64          } else {
65              revert Errors.GCNotPendleMarket(msg.sender);
66          }
```

```
67      }
68
69      constructor(
70          address _pendle,
71          address _marketFactory,
72          address _marketFactory2,
73          address _marketFactory3,
74          address _marketFactory4
75      ) {
76          pendle = _pendle;
77          marketFactory = IPMarketFactory(_marketFactory);
78          marketFactory2 = IPMarketFactory(_marketFactory2);
79          marketFactory3 = IPMarketFactory(_marketFactory3);
80          marketFactory4 = IPMarketFactory(_marketFactory4);
81      }
```

We found that the current GaugeController has all the 4 marketFactory addresses set.

Status

✓ Fixed

[WP-I5] In future upgrades of PendleMarketFactoryV6Upg, any changes to the `getMarketConfig` interface will break compatibility with previously deployed Market contracts.

Informational

Issue Description

Since markets are not upgradeable, when upgrading PendleMarketFactoryUpg to a new version, backward compatibility must be maintained for factory interfaces that are referenced by previously deployed market contracts.

```
107  function getMarketConfig(
108          address market,
109          address router
110      ) external view returns (address _treasury, uint80 _overriddenFee, uint8
     _reserveFeePercent) {
111          (_treasury, _reserveFeePercent) = (treasury, reserveFeePercent);
112          _overriddenFee = overriddenFee[router][market];
113      }
```

```
295  function readState(address router) public view returns (MarketState memory market)
     {
296          market.totalPt = _storage.totalPt;
297          market.totalSy = _storage.totalSy;
298          market.totalLp = totalSupply().Int();
299
300          uint80 overriddenFee;
301
302          (market.treasury, overriddenFee, market.reserveFeePercent) =
     IPMarketFactory(factory).getMarketConfig(
303              address(this),
304              router
305          );
306
307          market.lnFeeRateRoot = overriddenFee == 0 ? lnFeeRateRoot : overriddenFee;
308          market.scalarRoot = scalarRoot;
309          market.expiry = expiry;
```

```
310
311             market.lastLnImpliedRate = _storage.lastLnImpliedRate;
312         }
```

During upgrades of `PendleYieldContractFactoryUpg` , the new implementation must maintain backwards compatibility with all previously deployed versions of PT and YT.

Since PendleMarketFactory uses PendleYieldContractFactory's interface, upgrades to `PendleYieldContractFactoryUpg` must also ensure the new implementation remains compatible with PendleMarketFactory.

Status

ⓘ Acknowledged

Appendix

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by WatchPug; however, WatchPug does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Smart Contract technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.