StandX EVM Audit Report

Apr 22, 2025

Table of Contents

Summary 2

Overview 3

Issues 4

[WP-L3] `Settler` inherits `PausableUpgradeable` but is unused 4

[WP-I8] Anti-pattern: Unnecessary balance/allowance checks before transferFrom 5

[WP-I10] Consider making the accounts with roles enumerable to better track and verify permission assignments. 6

[WP-I17] The token definition in `IGateway` interface differs from the Gateway implementation code 8

[WP-N19] Unused `Gateway.state.usdcToken` 11

Appendix 13

Disclaimer 14

Summary

This report has been prepared for StandX 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	**StandX**
Codebase	**https://github.com/standx-labs/stand_evm_contract_audit**
Commit	**fdbbe9880e8112e74c05048208ffdbda17112453**
Language	**Solidity**

Audit Summary

Delivery Date	**Apr 22, 2025**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**5**

[WP-L3] `Settler` inherits `PausableUpgradeable` but is unused

Low

Issue Description

```
11  contract Settler is
12      Initializable,
13      AccessControlUpgradeable,
14      PausableUpgradeable,
15      UUPSUpgradeable
16  {
```

The contract inherits `PausableUpgradeable` but does not use the `whenNotPaused` modifier, and there are no public methods for pause and unpause functionality.

[WP-I8] Anti-pattern: Unnecessary balance/allowance checks before transferFrom

Informational

Issue Description

```
137         require(
138             baseToken.token.balanceOf(msg.sender) >= depositAmount,
139             GatewayLib.TokenBalance()
140         );
141         require(
142             baseToken.token.allowance(msg.sender, address(this)) >=
143                 depositAmount,
144             GatewayLib.TokenAllowance()
145         );
```

The anti-pattern is performing explicit balanceOf and allowance checks before calling `transferFrom` . The `transferFrom` function itself already includes these checks internally. Performing them beforehand is redundant and wastes gas.

Reference: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/ERC20.sol#L189

[WP-I10] Consider making the accounts with roles enumerable to better track and verify permission assignments.

Informational

Issue Description

`AccessControl` makes it difficult to check which members currently have a role. This is especially concerning for high-privilege roles like `DEFAULT_ADMIN_ROLE` and `UPGRADER_ROLE`.

For example, consider having Gateway inherit from `AccessControlEnumerableUpgradeable`.

```
@@ 1,5 @@
6  import {AccessControlUpgradeable} from
   "@openzeppelin-upgradeable/access/AccessControlUpgradeable.sol";
@@ 7,17 @@
18
19 contract Gateway is
20     IGateway,
21     Initializable,
22     AccessControlUpgradeable,
23     PausableUpgradeable,
24     ReentrancyGuardUpgradeable,
25     UUPSUpgradeable
26 {
27     bytes32 public constant PAUSER_ROLE = keccak256("PAUSER_ROLE");
28     bytes32 public constant UPGRADER_ROLE = keccak256("UPGRADER_ROLE");
@@ 29,52 @@
53
54     function setState(
55         GatewayLib.State memory newState
56     ) public onlyRole(DEFAULT_ADMIN_ROLE) nonReentrant {
57         state = newState;
58     }
59
@@ 60,66 @@
67
68     function updateUserFlag(
```

```
69          address userAddress,
70          bool enabled,
71          bool canClaim
72      ) public onlyRole(DEFAULT_ADMIN_ROLE) nonReentrant returns (bool) {
73          (UserLib.User storage userInfo, ) = _getOrNewUserNoCheck(userAddress);
74          userInfo.enabled = enabled;
75          userInfo.canClaimReward = canClaim;
76          return true;
77      }
78
79      function updateUser(
80          UserLib.User memory userInfo
81      ) public onlyRole(DEFAULT_ADMIN_ROLE) nonReentrant {
82          UserLib.User storage user = UserLib.getUser(userInfoMap, userInfo.user);
83          require(user.user == userInfo.user, GatewayLib.InvalidUser());
84          require(!user.enabled, GatewayLib.InvalidUser());
85          uint256 claimNonce = userInfo.claimNonce;
86          UserLib.updateUser(userInfoMap, userInfo);
87          // We know storage user is a pointer.
88          UserLib.User storage user_new = UserLib.getUser(
89              userInfoMap,
90              userInfo.user
91          );
92          require(claimNonce == user_new.claimNonce, GatewayLib.InvalidNonce());
93      }
94
95      function updateTokenList(
96          TokenLib.Token memory baseToken
97      ) public onlyRole(DEFAULT_ADMIN_ROLE) nonReentrant {
98          TokenLib.setToken(tokenInfoMap, baseToken);
99          emit UpdateTokenList(baseToken.token);
100     }
101
102     function _authorizeUpgrade(
103         address newImplementation
104     ) internal override onlyRole(UPGRADER_ROLE) {}
105

@@ 106,362 @@

363 }
```

[WP-I17] The token definition in `IGateway` interface differs from the Gateway implementation code

Informational

Issue Description

- The `token` parameter in `deposit()` and the `targetToken` parameter in `withdrawRequest()` are defined as "USDT or USDC" in #C0_L21 and #C0_L31 respectively.
- The Gateway implementation allows any token that is `enabled` in the `Gateway.tokenInfoMap` storage.

The Anchor version implementation in price$_oracle.rsL23 - L38$ only allows USDT and USDC. This behavior has been changed in the Solidity version of `Gateway`, and it's possible that `IGateway` was not updated accordingly.

```
20      /// @notice Deposit tokens to mint DUSD
21      /// @param token Token to deposit (USDT or USDC)
22      /// @param amount Amount of tokens to deposit
23      /// @param oracle Price oracle data for conversion rate
24      function deposit(
25          IERC20 token,
26          uint256 amount,
27          PriceOracle.Oracle memory oracle
28      ) external;
29
30      /// @notice Request to withdraw tokens by burning DUSD
31      /// @param targetToken Token to receive (USDT or USDC)
32      /// @param dusdAmount Amount of DUSD to burn
33      function withdrawRequest(IERC20 targetToken, uint256 dusdAmount) external;
```

```
110     function deposit(
111         IERC20 token,
112         uint256 amount,
113         PriceOracle.Oracle memory oracle
114     ) public whenNotPaused nonReentrant {
115         _getOrNewUser(msg.sender);
116         TokenLib.Token storage baseToken = TokenLib.getToken(
```

```
117             tokenInfoMap,
118             address(token)
119         );
@@ 120,131 @@
132
133         depsoitAmountInDUSD = PriceOracle.calculateMintAmount(
134             oracle,
135             baseToken,
136             depsoitAmountInDUSD,
137             state.usdtToken,
138             state.priceOracle
139         );
140
@@ 141,154 @@
155     }
156
157     function withdrawRequest(
158         IERC20 targetToken,
159         uint256 dusdAmount
160     ) public whenNotPaused nonReentrant {
161         (UserLib.User storage user, ) = _getOrNewUser(msg.sender);
162
163         TokenLib.getToken(tokenInfoMap, address(targetToken));
@@ 164,194 @@
195     }
```

Token.sol

```
21     function getToken(
22         TokenMap.AddressToTokenMap storage map,
23         address tokenAddress
24     ) internal view returns (Token storage) {
25         Token storage token = TokenMap.get(map, tokenAddress);
26         require(address(token.token) == tokenAddress, "Invalid token");
27         require(token.enabled, "Invalid token");
28         return token;
29     }
```

PriceOracle.sol

```
52      function calculateMintAmount(
53          Oracle memory oracle,
54          TokenLib.Token storage baseToken,
55          uint256 amount,
56          IERC20 usdtToken,
57          address priceOracle
58      ) internal view returns (uint256) {
59          uint256 rate = oracle.price;
60          if (baseToken.token == usdtToken) {
61              rate = EXCHANGE_RATIO_BASE;
62          } else {
63              checkOracle(oracle, priceOracle, baseToken);
64          }
65          return applyExchangeRateAndFee(amount, rate, 0, true);
66      }
```

PriceOracle.sol

```
28      function checkOracle(
29          Oracle memory oracle,
30          address priceOracle,
31          TokenLib.Token storage baseToken
32      ) internal view returns (bool) {
    @@ 33,47 @@
48          require(oracle.token == baseToken.token, InvalidToken());
49          return true;
50      }
```

Status

✓ Fixed

[WP-N19] Unused `Gateway.state.usdcToken`

Issue Description

The state variable `usdcToken` is not used anywhere in the codebase.

https://github.com/standx-labs/stand_evm_contract_audit_watchpug/blob/8db2a6592eeb4a93e315bc8c82796ce63c9435ce/src/GatewayLib.sol#L36-L48

```
36      /// @notice Configuration state of the gateway
37      /// @dev Holds all the critical addresses and parameters for the gateway
38      struct State {
39          Settler settler; // Settler contract for DUSD operations
40          address vault; // Vault address for token storage
41          uint256 userCnt; // Total number of users
42          IERC20 dusdToken; // DUSD token address
43          IERC20 usdtToken; // USDT token address
44          IERC20 usdcToken; // USDC token address
45          address offlineSigner; // Address authorized to sign yield claims
46          address priceOracle; // Oracle for price feeds
47          uint256 protocolFee; // Protocol fee in basis points
48      }
```

https://github.com/standx-labs/stand_evm_contract_audit_watchpug/blob/8db2a6592eeb4a93e315bc8c82796ce63c9435ce/src/Gateway.sol#L19-L367

```
19  contract Gateway is
    @@ 20,25 @@
26  {
    @@ 27,33 @@
34
35      GatewayLib.State private state;
36
    @@ 37,366 @@
367 }
```

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.