OpenZeppelin | security

Mountain Protocol wUSDM Audit

October 23, 2023

Table of Contents

Summary

Type	DeFi
Timeline	From 2023-10-06 To 2023-10-10
Languages	Solidity

Total Issues	4 (4 resolved)
Critical Severity Issues	0 (0 resolved)
High Severity Issues	0 (0 resolved)
Medium Severity Issues	0 (0 resolved)
Low Severity Issues	0 (0 resolved)
Notes & Additional Information	4 (4 resolved)
Client Reported Issues	0 (0 resolved)

Scope

We audited the mountainprotocol/tokens repository at commit 97f99ee.

In scope were the following contracts:

```
contracts
└── wUSDM.sol
```

Update: *The fixes to our findings were finalised in the same repository at the 78c6556badd1c8db2de24dbcf2adca60436d9543 commit.*

System Overview

The wUSDM contract represents an ERC-4626 token vault. It allows users to deposit USDM stablecoin tokens in exchange for wUSDM tokens. The USDM tokens are rebasing, whereas the wUSDM tokens are non-rebasing. Moreover, the wUSDM tokens are intended for liquidity provision and seamless integration with various protocols within the DeFi ecosystem.

Additionally, the wUSDM contract incorporates the ERC-2612 permit functionality, allowing for the use of signatures to grant token allowances. The close relationship between the wUSDM and USDM contracts is also worth noting; the wUSDM contract leverages the account block list from the USDM contract in order to generally govern transfers, and more specifically to prevent any transfers from accounts present in said block list.

Furthermore, the wUSDM token transfers can be paused in two ways: either by being paused directly from within the wUSDM contract or in the event the USDM contract is paused.

Privileged Roles

The wUSDM contract implements the following privileged roles:

- `DEFAULT_ADMIN_ROLE`: can grant the `PAUSE_ROLE` and `UPGRADE_ROLE` roles.
- `PAUSE_ROLE`: can pause the wUSDM contract.
- `UPGRADE_ROLE`: can upgrade the wUSDM implementation contract.

Trust Assumptions

- The wUSDM contract depends on the USDM contract's account blocking and contract pausing capabilities. The USDM contract is trusted to adhere to its specification.
- The holders of the privileged roles `DEFAULT_ADMIN_ROLE`, `PAUSE_ROLE` and `UPGRADE_ROLE` are expected to be non-malicious, and to be acting in the protocol's best interest.

Notes & Additional Information

N-01 Incorrect Size of Storage Gap

The wUSDM contract implements the `__gap` storage variable to reserve space for adding new state variables in the future without compromising storage compatibility with existing deployments. The convention used by the OpenZeppelin contracts is to use a `__gap` array with the size of 50 subtracted by the number of the used storage slots. wUSDM uses two storage slots for the `USDM` variable and `_nonces` mapping but the size of the `__gap` array is `49`.

Consider changing the size of the `__gap` array to `48`.

Update: *Resolved in pull request #43 at commit a98c49f.*

N-02 Missing Named Parameters in Mapping

Since Solidity 0.8.18, developers can utilize named parameters in mappings. This means that mappings can take the form of `mapping(KeyType KeyName? => ValueType ValueName?)`. This updated syntax provides a more transparent representation of the mapping's purpose.

Consider adding named parameters to the `_nonces` mapping in the `wUSDM` contract to improve the readability and maintainability of the code.

Update: *Resolved in pull request #44 at commit efcb523.*

N-03 Missing Docstrings

Throughout the `wUSDM` contract there are several parts that do not have docstrings.

- Missing docstrings for the entire `IUSDM` interface.
- Missing `@return` for the `paused` function.
- Missing `@param from` for the `_beforeTokenTransfer` function.

- Missing `@param owner` and `@return` for the `_useNonce` function.

Consider thoroughly documenting all functions (and their parameters) that are part of any contract's public API. Functions implementing sensitive functionality, even if not public, should be clearly documented as well. When writing docstrings, consider following the Ethereum Natural Specification Format (NatSpec).

Update: *Resolved in pull request #47 at commit cf4a239.*

N-04 Overridden Hook Does Not Follow Rules of Hooks

The wUSDM contract that overrides the `_beforeTokenTransfer` hook function does not follow the guidelines on hooks overriding. The overridden hook is not marked with the `virtual` keyword and does not call the parent's hook using `super`.

Consider making the `_beforeTokenTransfer` hook function `virtual` and adding a call to the parent's hook.

Update: *Resolved in pull request #46 at commit 0a2dc4b.*

Conclusion

Our evaluation of the wUSDM codebase uncovered no significant issues. The code demonstrates simplicity and efficiency. In addition, it incorporates multiple components from the previously audited USDM contract, leveraging its security features. Although no vulnerabilities were identified, several recommendations were nonetheless made in order to enhance security best practices, improve code readability, and streamline future integrations. It is worth noting that throughout the audit, the Mountain Protocol team was very responsive.