Morpho: Morpho Token Upgradeable

Security Review

Cantina Managed review by:

Saw-mon and Natalie, Lead Security Researcher

M4rio.eth, Security Researcher

November 5, 2024

Contents

1 Introduction

1.1 About Cantina

Cantina is a security services marketplace that connects top security researchers and solutions with clients. Learn more at cantina.xyz

1.2 Disclaimer

Cantina Managed provides a detailed evaluation of the security posture of the code at a particular moment based on the information available at the time of the review. While Cantina Managed endeavors to identify and disclose all potential security issues, it cannot guarantee that every vulnerability will be detected or that the code will be entirely secure against all possible attacks. The assessment is conducted based on the specific commit and version of the code provided. Any subsequent modifications to the code may introduce new vulnerabilities that were absent during the initial review. Therefore, any changes made to the code require a new security review to ensure that the code remains secure. Please be advised that the Cantina Managed security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.3 Risk assessment

Severity	Description
Critical	*Must* fix as soon as possible (if already deployed).
High	Leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
Medium	Global losses <10% or losses to only a subset of users, but still unacceptable.
Low	Losses will be annoying but bearable. Applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.
Gas Optimization	Suggestions around gas saving practices.
Informational	Suggestions around best practices or readability.

1.3.1 Severity Classification

The severity of security issues found during the security review is categorized based on the above table. Critical findings have a high likelihood of being exploited and must be addressed immediately. High findings are almost certain to occur, easy to perform, or not easy but highly incentivized thus must be fixed as soon as possible.

Medium findings are conditionally possible or incentivized but are still relatively likely to occur and should be addressed. Low findings a rare combination of circumstances to exploit, or offer little to no incentive to exploit but are recommended to be addressed.

Lastly, some findings might represent objective improvements that should be addressed but do not impact the project's overall security (Gas and Informational findings).

2 Security Review Summary

The Morpho Protocol is a decentralized, noncustodial lending protocol implemented for the Ethereum Virtual Machine.

From Oct 23rd to Oct 25th the Cantina team conducted a review of morpho-token-upgradeable on commit hash 1bf203b8.

The Cantina team reviewed Morpho's morpho-token-upgradeable changes holistically on commit hash daa7ba28 and determined that all issues were resolved and no new issues were identified.

The team identified a total of **7** issues in the following risk categories:

- Critical Risk: 0
- High Risk: 0
- Medium Risk: 0
- Low Risk: 0
- Gas Optimizations: 0
- Informational: 7

3 Findings

3.1 Informational

3.1.1 A prefix can be added to the namespaced storage layout's id

Severity: Informational

Context: DelegationToken.sol#L32-L34

Description: To better isolate and avoid possible future name collisions between different projects it might make sense to add a prefix to the namespaced storage layout's id (as specified in EIP 7201). Currently it is only `DelegationToken`.

Recommendation: It might make sense to change it to something like `morpho.storage.DelegationToken`:

```
// keccak256(abi.encode(uint256(keccak256("morpho.storage.DelegationToken")) - 1)) & ~bytes32(uint256(0xff))
bytes32 internal constant DelegationTokenStorageLocation = ...;

/* STORAGE LAYOUT */

/// @custom:storage-location erc7201:morpho.storage.DelegationToken
```

Morpho: Fixed in PR 75.

Cantina Managed: Fixed.

3.1.2 Missing function to get the current implementation

Severity: Informational

Context: *(No context files were provided by the reviewer)*

Description: The `DelegationToken` is using `UUPSUpgradable` proxy which uses `ERC1967` as proxy storage slots. Currently there is no way to get the current implementation of the proxy.

Recommendation: To get the current implementation of a proxy the `ERC1967Utils` library offers a function called `ERC1967Utils.getImplementation()`. Consider wrapping this in a public function:

```
function getImplementation() external view returns (address){
    return ERC1967Utils.getImplementation();
}
```

Morpho: Fixed here in PR 76.

Cantina Managed: Fixed.

3.1.3 Misleading comments related with `ERC20Wrapper` and `ERC20WrapperBundler` inside the Wrapper

Severity: Informational

Context: Wrapper.sol#L41, Wrapper.sol#L51, Wrapper.sol#L62

Description: Throughout the `Wrapper` contract we can notice 2 instances where the comments are misleading:

1. Currently, the `Wrapper` contains 2 functions `withdrawTo` and `depositFor` which are compliant with the `ERC20Wrapper`. The contract itself is not fully compliant, only these 2 functions, therefore the comments on L41 and L51 should be more specific:

```
/// @dev Compliant to `ERC20Wrapper.depositTo` function from OZ for convenience.
/// ...
/// @dev Compliant to `ERC20Wrapper.withdrawTo` function from OZ for convenience.
```

2. On line 62, `To ease wrapping via the bundler contract` it states that the `Wrapper` is supposed to be used with the `ERC20WrapperBundler` for wrapping. One might think that it can be used as well for unwrapping due to the presence of `withdrawTo` function. That is not true because the `erc20WrapperWithdrawTo` function in the Bundler requires additional modifications:

balanceOf should be implemented in the Wrapper, approval should be implemented in the ERC20WrapperBundler.erc20WrapperWithdrawTo. Consider adding extra context on the withdrawTo function specifying that is not compatible with the ERC20WrapperBundler.erc20WrapperWithdrawTo

Recommendation: Consider adding the extra context mentioned above in the comments.

Morpho: Acknowledged:

1. The contract is not expected to be fully compliant with ERC20Wrapper.
2. It is mentioned in Wrapper.sol#L61 that the bundler is expected to ease the wrapping process, but nothing is said about unwrapping.

Cantina Managed: Acknowledged.

3.1.4 LEGACY_MORPHO and NEW_MORPHO could be defined as IERC20

Severity: Informational

Context: Wrapper.sol#L15, Wrapper.sol#L20

Description: Currently, the LEGACY_MORPHO and NEW_MORPHO are defined as address and then when they are used a wrap to IERC20 interface is performed:

```
IERC20(LEGACY_MORPHO).transferFrom(msg.sender, address(this), value);
```

Recommendation: Consider defining them as IERC20 by default to avoid the extra wrapping in IERC20

Morpho: Acknowledged. We prefer to stay with the current version.

Cantina Managed: Acknowledged.

3.1.5 Missing version field in the MorphoTokenEthereum and MorphoTokenOptimism

Severity: Informational

Context: MorphoTokenEthereum.sol#L20, MorphoTokenOptimism.sol#L30

Description: When an upgradable contract is defined, it is customed to define a version field in the implementation which will specify the current version. The field is usually incremented with every upgrade. The MorphoTokenEthereum and MorphoTokenOptimism are currently missing this field.

Recommendation: Consider adding a string public constant version = "1"; field for the first version of the implementation.

Morpho: Acknowledged. We don't plan on deploying other versions, so we'd rather not to touch to the version parameter.

Cantina Managed: Acknowledged.

3.1.6 Typo in the DelegationToken

Severity: Informational

Context: DelegationToken.sol#L134

Description: The DelegationToken _update function contains a typo in the comment:

```
/// @dev Emits a {IDelegates-DelegateVotesChanged} event.
```

The function actually emits DelegatedVotingPowerChanged and there is no IDelegates interface, the interface is called IDelegation

Recommendation: Consider modifying the comment to reflect the correct event emitted:

```
/// @dev Emits a {DelegatedVotingPowerChanged} event.
```

Morpho: Fixed in PR 77.

Cantina Managed: Fixed.

3.1.7 Use init functions provided by the Open-Zeppelin contracts

Severity: Informational

Context: MorphoTokenEthereum.sol#L30-L35, MorphoTokenOptimism.sol#L65-L70

Description: In this context one can use the `__Ownable_init(owner)` instead of inlining the same logic using:

```
require(owner != address(0), ZeroAddress());

// ...

_transferOwnership(owner);
```

Recommendation: Apply the following patch:

```
diff --git a/src/MorphoTokenEthereum.sol b/src/MorphoTokenEthereum.sol
index 6e0dde3..67d4833 100644
--- a/src/MorphoTokenEthereum.sol
+++ b/src/MorphoTokenEthereum.sol
@@ -27,12 +27,10 @@ contract MorphoTokenEthereum is DelegationToken {
     /// @param owner The new owner.
     /// @param wrapper The wrapper contract address to migrate legacy MORPHO tokens to the new one.
     function initialize(address owner, address wrapper) external initializer {
-        require(owner != address(0), ZeroAddress());
-
+        __Ownable_init(owner);
         __ERC20_init(NAME, SYMBOL);
         __ERC20Permit_init(NAME);

-        _transferOwnership(owner);
         _mint(wrapper, 1_000_000_000e18); // Mint 1B to the wrapper contract.
     }

diff --git a/src/MorphoTokenOptimism.sol b/src/MorphoTokenOptimism.sol
index 8383588..1cd2d48 100644
--- a/src/MorphoTokenOptimism.sol
+++ b/src/MorphoTokenOptimism.sol
@@ -62,12 +62,9 @@ contract MorphoTokenOptimism is DelegationToken, IOptimismMintableERC20 {
     /// @notice Initializes the contract.
     /// @param owner The new owner.
     function initialize(address owner) external initializer {
-        require(owner != address(0), ZeroAddress());
-
+        __Ownable_init(owner);
         __ERC20_init(NAME, SYMBOL);
         __ERC20Permit_init(NAME);
-
-        _transferOwnership(owner);
     }

     /// @dev Allows the StandardBridge on this network to mint tokens.
```

Morpho: Fixed in commit 220dd096.

Cantina Managed: Fixed.