wstEth L2 Token Audit Report

Prepared by Cyfrin

Version 2.0

Lead Auditors

Giovanni Di Siena

0kage

Assisting Auditors

Hans

June 10, 2024

Contents

1 About Cyfrin

Cyfrin is a Web3 security company dedicated to bringing industry-leading protection and education to our partners and their projects. Our goal is to create a safe, reliable, and transparent environment for everyone in Web3 and DeFi. Learn more about us at cyfrin.io.

2 Disclaimer

The Cyfrin team makes every effort to find as many vulnerabilities in the code as possible in the given time but holds no responsibility for the findings in this document. A security audit by the team does not endorse the underlying business or product. The audit was time-boxed and the review of the code was solely on the security aspects of the solidity implementation of the contracts.

3 Risk Classification

	Impact: High	**Impact: Medium**	**Impact: Low**
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4 Protocol Summary

The `WstEthL2Token` is an upgradeable ERC-20 token following the UUPS proxy pattern, designed to be used with the Wormhole Native Token Transfers (NTT) protocol. It implements the `INttToken` interface and exposes the relevant functions to be called by the `NttManager` contract. This design allows the Lido wstETH token to be natively transferred between Ethereum and other supported chains without the need for multiple third-party bridge tokens.

5 Audit Scope

Cyfrin conducted an audit `WstEthL2Token` based on the code present in the repository commit hash e92b7e4.

The following files were included in the scope of the audit:

- src/WstEthL2Token.sol
- script/lib/Upgrades.sol
- test/token/WstEthL2Token.t.sol
- test/token/WstEthL2TokenHarness.sol
- test/token/WstEthL2TokenV2Fake.sol

6 Executive Summary

Over the course of 5 days, the Cyfrin team conducted an audit on the wstEth L2 Token smart contracts provided by Wormhole Foundation. In this period, a total of 5 issues were found.

A number of informational findings were raised by this review of the `WstEthL2Token` contract, with no material security vulnerabilities identified.

Summary

Project Name	wstEth L2 Token
Repository	example-wormhole-axelar-wsteth
Commit	e92b7e4843c3...
Audit Timeline	Jun 3rd - Jun 7th
Methods	Manual Review

Issues Found

Critical Risk	0
High Risk	0
Medium Risk	0
Low Risk	0
Informational	5
Gas Optimizations	0
Total Issues	5

Summary of Findings

[I-1] Include the `UUPSUpgradeable` initializer comment	Resolved
[I-2] Consider also applying the `onlyMinter` modifier to `WstEthL2Token::burnFrom` even though the current implementation always reverts	Acknowledged
[I-3] `INttToken::InsufficientBalance` custom error is unused	Acknowledged
[I-4] Unchained `ERC20Upgradeable` and `OwnableUpgradeable` initializers can be called directly	Acknowledged
[I-5] `WstEthL2Token::initialize` has no access controls	Acknowledged

7 Findings

7.1 Informational

7.1.1 Include the `UUPSUpgradeable` initializer comment

While a comment was included for the `ERC20Burnable` initializer, the corresponding comment for the `UUPSUpgradeable` initializer was not. It has since been added in the current scope following a suggestion prior to the review start date.

Wormhole Foundation: Fixed in the current scope.

Cyfrin: Acknowledged.

7.1.2 Consider also applying the `onlyMinter` modifier to `WstEthL2Token::burnFrom` even though the current implementation always reverts

`WstEthL2Token::burnFrom` currently reverts as the method is not implemented and should not be called, so while this poses no direct security risk, the `onlyMinter` modifier should be applied to this function similar to the other permissioned functions. This could help to avoid access control issues in future, assuming the contract is upgraded to implement this function, since only the `NttManager` should be able to burn funds.

Wormhole Foundation: Acknowledged.

Cyfrin: Acknowledged.

7.1.3 `INttToken::InsufficientBalance` custom error is unused

`INttToken` exposes a custom error `InsufficientBalance()`, which is currently unused and could be added to `WstEthL2Token::burn` to provide a more verbose error message, albeit at the expense of gas optimization.

```
function burn(uint256 _value) public override(INttToken, ERC20BurnableUpgradeable) onlyMinter {
+   if (_value > balanceOf(_msgSender())) {
+       revert InsufficientBalance(balanceOf(_msgSender()), _value);
+   }
    ERC20BurnableUpgradeable.burn(_value);
}
```

Wormhole Foundation: Acknowledged.

Cyfrin: Acknowledged.

7.1.4 Unchained `ERC20Upgradeable` and `OwnableUpgradeable` initializers can be called directly

While this comment correctly identifies that ERC20 extensions do not linearize calls to the `ERC20Upgradeable` initializer, it is sufficient and still recommended to call the unchained `ERC20Upgradeable`/`OwnableUpgradeable` equivalents directly.

Wormhole Foundation: Acknowledged.

Cyfrin: Acknowledged.

7.1.5 `WstEthL2Token::initialize` has no access controls

Care should be taken when deploying and/or upgrading `WstEthL2Token` due to the lack of access controls on its initializer function (and any potential future reinitializers). This does not pose a direct security risk when deployed/upgraded and initialized in an atomic transaction using the script provided, but it is pertinent to note nonetheless.

Wormhole Foundation: Acknowledged.

Cyfrin: Acknowledged.