LayerZero VerifierNetwork Security Analysis Report and Formal Verification Properties

certora

29.08.2023

Table of Contents

Table of Contents 1
Summary 2
- Summary of findings 2
- Disclaimer 3

Main Issues Discovered 4

Issue-01: 4
Issue-02: 4
Issue-03: 4
Issue-04: 5
Note-05: 5
Note-06: 5
Notations 6
VerifierNetwork.sol properties 6

Summary

This document describes the specification and verification of the new **LayerZero Verifier Network contract** using the Certora Prover and manual code review findings. The work was undertaken from **18th August 2023** to **28th August 2023**. The latest commit that was reviewed is 796c167.

The following contracts list is included in the **scope**:

```
packages/layerzero-v2/evm/messagelib/contracts/uln/VerfierNetwork.sol
packages/layerzero-v2/evm/messagelib/contracts/Worker.sol
packages/layerzero-v2/evm/messagelib/contracts/uln/MultiSig.sol
@openzeppelin/contracts/access/AccessControl.sol
@openzeppelin/contracts/utils/Context.sol
```

The contracts were verified for Solidity version 0.8.19.

The Certora Prover demonstrated that the implementation of the Solidity contracts above is correct with respect to the formal rules written by the Certora team. In addition, the team performed a manual audit of all Solidity contracts. During the verification process and the manual audit, the Certora Prover discovered bugs in the Solidity contracts code, as listed below.

Summary of findings

The table below summarizes the issues discovered during the audit, categorized by severity.

Severity	Total discovered	Total fixed	Total acknowledged
High	0	0	0
Medium	0	0	0
Low	4	0	0
Informational	2		
Total	**6**	**0**	**0**

Disclaimer

The Certora Prover takes a contract and a specification as input and formally proves that the contract satisfies the specification in all scenarios. Notably, the guarantees of the Certora Prover are scoped to the provided specification and the Certora Prover does not check any cases not covered by the specification.

Even though we hope this information is helpful, we provide no warranty of any kind, explicit or implied. The contents of this report should not be construed as a complete guarantee that the contract is secure in all dimensions. In no event shall Certora or any of its employees be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the results reported here.

Main Issues Discovered

Issue-01: Contract is left with no admins

Severity: Medium
Probability: Low
Category: Loss of control
File(s): `VerifierNetwork.sol`
Bug description: There is no admin.
Exploit scenario: The `revokeRole` function is called by the last admin with parameters `ADMIN_ROLE` and their own address, thus renouncing their own role, hence the contract is left with zero admins.
Property violated: AtLeastOneAdmin
Implications: The quorum will have to choose a new admin.
LayerZero's response: "quorum can grant new admin with quorumChangeAdmin."

Issue-02: Zero address is admin

Severity: Low
Probability: Low
Category: Invalid state
File(s): `VerifierNetwork.sol`
Bug description: Account with zero address could be an admin.
Exploit scenario: Can only happen during construction, in the `grantRole` or the `quorumChangeAdmin` function.
Property violated: adminIsNonZero
Implications: Giving admin rights to zero address has no benefits other than saving gas for the missing check. In this case, we can always set a new admin with `quorumChangeAdmin` and revoke the admin role from the zero address if needed.
LayerZero's response: "we check if address is zero in constructor before setting. won't check for grantRole, if such action is performed, suppose it is intentional."

Issue-03: Contradiction of ALLOWLIST and DENYLIST ROLES

Severity: Low
Probability: Low
Category: State contradiction
File(s): `VerifierNetwork.sol`
Bug description: Account can have both `DENYLIST` and `ALLOWLIST` roles.
Exploit scenario: By calling `grantRole` with one of the said roles when the other is already acquired by the user.
Property violated: cannotBeBothAllowedAndDenied
Implications: Currently this is not a major problem, but if we checked for instance for `ALLOWLIST` first in `Worker:onlyAcl`, some entity could get approval even when on `DENYLIST`. The issue might become severe if a future check for the account being on those lists is reversed in order, where the `ALLOWLIST` role is checked first.
LayerZero's response: "`ALLOWLIST` and `DENYLIST` are intended to work together with `onlyAcl` and `allowlistSize`, will add a comment that the roles should only be used with `onlyAcl`."

Issue-04: Grieving by admin

Severity: Low
Probability: Low
Category: Grieving
File(s): `VerifierNetwork.sol`
Bug description: Contract's admin can reduce confirmations for a specific message by replaying an old confirmation message for that message.
Implications: Minor annoyance to the signers.
LayerZero's response: Grieving interfaces are fine. The quorum can swap the bad admin out.

Note-05: Admins can revoke each other

Severity: Informational
Probability: High
Category: Fragile governance
File(s): `VerifierNetwork.sol`
Bug description: Admins can revoke other admins which could be a design flaw depending on the way you want to handle admin roles.
LayerZero's response: Acknowledged.

Note-06: Function name is misleading

Severity: Informational
Probability: High
Category: Misleading documentation
File(s): `VerifierNetwork.sol`
Bug description: Documentation for `quorumChangeAdmin` as well as its name says its purpose is to change admin, but it actually adds admin and doesn't remove any existing admin.
LayerZero's response: Acknowledged.

Notations

✅Indicates the rule is formally verified.
❌Indicates the rule is violated.
⏳Indicates the rule is timing out.

VerifierNetwork.sol properties

Assumptions

- Loop unrolling: We assume any loop can have at most 3 iterations.
- View functions filtering: Rules checking state changes of all available functions do not check view functions.
- Optimistic fallback: all fallback functions were assumed to be empty.
- The following functions have been ignored during this verification, we assume that those never revert and can return arbitrary values:
  - `ILayerZeroPriceFeed`
    - `estimateFeeByEid(uint32,uint,uint)`
    - `estimateFeeOnSend(uint32,uint,uint)`
  - `IMessageLib`
    - `withdrawFee(address, uint)`
  - `ILayerZeroUltraLightNodeV2`
    - `withdrawNative(address, uint)`
  - `IUltraLightNode`
    - `deliver(bytes, bytes32)`
    - `deliverable(bytes, bytes32)`
    - `verify(bytes, bytes32, uint64)`

Properties

1. ✅ Signers can only be changed with the `setSigner` function. (onlySetSignerCanChangeSigners)
2. ❌ No account can be both on `DENYLIST` and `ALLOWLIST`. (cannotBeBothAllowedAndDenied) - Issue 03
3. ❌ The zero address cannot be admin. (adminIsNonZero) - Issue 02
4. ✅ Quorum is never zero. (quorumIsNonzero)
5. ✅ There cannot be zero signers. (signerSizeIsNonZero)
6. ❌ There is always at least one admin. (AtLeastOneAdmin, violated by revokeRole) - Issue 01
7. ✅ `allowListSize` is equal to the number of users that have role `ALLOWLIST`. (allowListSizeEqualsListCount)

8. ✅ The `execute` function fails after passed `expiration`. (executeExpiredFails)
9. ✅ After successful execution of `grantRole(role, account)` the account has the role. (grantRoleWorks)
10. ✅ Only an admin can grant the admin role. (grantRoleProtected)
11. ✅ After successful execution of `revokeRole(role, account)` the account does not have the role. (revokeRoleWorks)
12. ✅ Admin changes only with `quorumChangeAdmin`, `grantRole` or `revokeRole` . (changeAdminOnlyWithRoleChangingFunctions)
13. ✅ The `verifySignatures` function doesn't revert. (verifySignaturesDoesNotRevert)
14. ✅ The `execute` function does not execute when there isn't a quorum of distinct signatures. (executeOnlyWithEnoughSignatures)
15. ✅ The `execute` function does not execute any job that was already executed. (executeDoesntExecuteDuplicateJobsSamePatch, executeDoesntExecuteDuplicateJobsDifferentPatch)