Zellic

Endpoint V2 (VerifierNetwork)

Smart Contract Security Assessment

August 25, 2023

Prepared for:

Ryan Zarick and Isaac Zhang

LayerZero Labs

Prepared by:

Jasraj Bedi and Aaron Esau

Zellic Inc.

Contents

About Zellic

Zellic was founded in 2020 by a team of blockchain specialists with more than a decade of combined industry experience. We are leading experts in smart contracts and Web3 development, cryptography, web security, and reverse engineering. Before Zellic, we founded perfect blue, the top competitive hacking team in the world. Since then, our team has won countless cybersecurity contests and blockchain security events.

Zellic aims to treat clients on a case-by-case basis and to consider their individual, unique concerns and business needs. Our goal is to see the long-term success of our partners rather than simply provide a list of present security issues. Similarly, we strive to adapt to our partners' timelines and to be as available as possible. To keep up with our latest endeavors and research, check out our website zellic.io or follow @zellic_io on Twitter. If you are interested in partnering with Zellic, please contact us at hello@zellic.io.

1 Executive Summary

Zellic conducted a security assessment for LayerZero Labs from July 26th to August 22nd, 2023. During this engagement, Zellic reviewed Endpoint V2 (VerifierNetwork) for security vulnerabilities, design issues, and general weaknesses in security posture. This assessment was conducted as a part of a larger assessment for Endpoint V2, but included reviews of patches up until commit 7a3ce889 for VerifierNetwork.

1.1 Goals of the Assessment

In a security assessment, goals are framed in terms of questions that we wish to answer. These questions are agreed upon through close communication between Zellic and the client. In this assessment, we sought to answer the following questions:

- Can an attacker trick the VerifierNetwork into delivering a message improperly?
- What centralization risks does Endpoint V2 (VerifierNetwork) have?

1.2 Non-goals and Limitations

We did not assess the following areas that were outside the scope of this engagement:

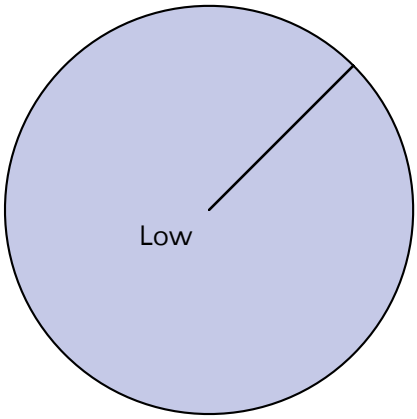
- Front-end components
- Infrastructure relating to the project
- Key custody

1.3 Results

During our assessment on the scoped Endpoint V2 (VerifierNetwork) contracts, we discovered two findings, all of which were low impact.

Breakdown of Finding Impacts

Impact Level	Count
Critical	0
High	0
Medium	0
Low	2
Informational	0



2 Introduction

2.1 About Endpoint V2 (VerifierNetwork)

LayerZero is a generic messaging protocol. VerifierNetwork is a modular smart contract implementing an N/M threshold validator network for LayerZero oracles. It is implemented as a basic primitive that can perform arbitrary calls after a quorum has been reached by the registered signers. As such, it is compatible with both Endpoint V1 and Endpoint V2.

2.2 Methodology

During a security assessment, Zellic works through standard phases of security auditing, including both automated testing and manual review. These processes can vary significantly per engagement, but the majority of the time is spent on a thorough manual review of the entire scope.

Alongside a variety of tools and analyzers used on an as-needed basis, Zellic focuses primarily on the following classes of security and reliability issues:

Basic coding mistakes. Many critical vulnerabilities in the past have been caused by simple, surface-level mistakes that could have easily been caught ahead of time by code review. Depending on the engagement, we may also employ sophisticated analyzers such as model checkers, theorem provers, fuzzers, and so on as necessary. We also perform a cursory review of the code to familiarize ourselves with the contracts.

Business logic errors. Business logic is the heart of any smart contract application. We examine the specifications and designs for inconsistencies, flaws, and weaknesses that create opportunities for abuse. For example, these include problems like unrealistic tokenomics or dangerous arbitrage opportunities. To the best of our abilities, time permitting, we also review the contract logic to ensure that the code implements the expected functionality as specified in the platform's design documents.

Integration risks. Several well-known exploits have not been the result of any bug within the contract itself; rather, they are an unintended consequence of the contract's interaction with the broader DeFi ecosystem. Time permitting, we review external interactions and summarize the associated risks: for example, flash loan attacks, oracle price manipulation, MEV/sandwich attacks, and so on.

Code maturity. We look for potential improvements in the code base in general. We look for violations of industry best practices and guidelines and code quality standards. We also provide suggestions for possible optimizations, such as gas optimization, upgradeability weaknesses, centralization risks, and so on.

For each finding, Zellic assigns it an impact rating based on its severity and likelihood. There is no hard-and-fast formula for calculating a finding's impact. Instead, we assign it on a case-by-case basis based on our judgment and experience. Both the severity and likelihood of an issue affect its impact. For instance, a highly severe issue's impact may be attenuated by a low likelihood. We assign the following impact ratings (ordered by importance): Critical, High, Medium, Low, and Informational.

Zellic organizes its reports such that the most important findings come first in the document, rather than being strictly ordered on impact alone. Thus, we may sometimes emphasize an "Informational" finding higher than a "Low" finding. The key distinction is that although certain findings may have the same impact rating, their *importance* may differ. This varies based on various soft factors, like our clients' threat models, their business needs, and so on. We aim to provide useful and actionable advice to our partners considering their long-term goals, rather than a simple list of security issues at present.

2.3 Scope

The engagement involved a review of the following targets:

Endpoint V2 (VerifierNetwork) Contracts

Repository	https://github.com/LayerZero-Labs/monorepo
Version	monorepo: `d5ae9e51e56b3e0d8b514715360dcc1ec8ed5621`
Program	packages/layerzero-v2/evm/messagelib/contracts/uln/VerifierNetwork.sol
Type	Solidity
Platform	EVM-compatible

2.4 Project Overview

Zellic was contracted to perform a security assessment of Endpoint V2 for a total of six person-weeks. The assessment was conducted over the course of four calendar weeks.

Contact Information

The following project manager was associated with the engagement:

Chad McDonald, Engagement Manager
chad@zellic.io

The following consultants were engaged to conduct the assessment:

Jasraj Bedi, Engineer
jazzy@zellic.io

Aaron Esau, Engineer
aaron@zellic.io

2.5 Project Timeline

The key dates of the engagement are detailed below.

July 26, 2023	Start of primary review period
August 22, 2023	End of primary review period

3 Detailed Findings

3.1 Centralization risk on `execute` function

- **Target**: VerifierNetwork
- **Category**: Business Logic
- **Likelihood**: Low
- **Severity**: Medium
- **Impact**: Low

Description

The `execute` function restricts the callers to only the admin role:

```
function execute(ExecuteParam[] calldata _params)
    external onlyRole(ADMIN_ROLE) {
    for (uint i = 0; i < _params.length; ++i) {
        // ...
        }
    }
}
```

However, this restriction is unnecessary because the function requires a quorum of valid signatures. If a quorum is reached, there should be no need for the quorum-ed signatures to be sent by a trusted party. This can instead be made trustless.

Impact

If an admin is unable to call `execute`, this will halt all the operations of the ULN. It would not be able to deliver any messages to the endpoint, even if all of the signers were online.

Recommendations

The function should be able to be called permissionlessly to ensure the signatures may always be submitted.

Remediation

LayerZero Labs, after discussing with Zellic has decided that this issue does not warrant a fix at the current time

3.2 Potential replay across chains

- **Target**: VerifierNetwork
- **Category**: Business Logic
- **Likelihood**: Low
- **Severity**: Low
- **Impact**: Low

Description

As LayerZero is a cross-chain application, VerifierNetwork might be deployed across multiple chains. There exists a possibility of message replay if signers are shared between multiple instances of VerifierNetwork. This is because there is no unique identifier pinning the VerifierNetwork the message can be executed at.

Impact

A message can be replayed between instances of VerifierNetwork if the signers/quorum is shared.

As the signed message includes the target address, calls to `onlySelf(orAdmin)` functions cannot be replayed. Furthermore, calls to ULN functions such as verify would not be useful to an attacker as well.

Recommendations

Add an identifier to VerifierNetwork that is checked as part of the signature.

Remediation

LayerZero labs acknowled the issue and has fixed it in commit 175c08bd

4 Threat Model

This assessment was conducted as part of the larger assessment for Endpoint V2. Please refer to the Endpoint V2 report for a detailed threat model.

5 Assessment Results

At the time of our assessment, the reviewed code was not deployed to the Ethereum Mainnet.

During our assessment on the scoped Endpoint V2 (VerifierNetwork) contracts, we discovered two findings, all of which were low impact. LayerZero Labs acknowledged all findings and implemented fixes.

5.1 Disclaimer

This assessment does not provide any warranties about finding all possible issues within its scope; in other words, the evaluation results do not guarantee the absence of any subsequent issues. Zellic, of course, also cannot make guarantees about any code added to the project after the version reviewed during our assessment. Furthermore, because a single assessment can never be considered comprehensive, we always recommend multiple independent assessments paired with a bug bounty program.

For each finding, Zellic provides a recommended solution. All code samples in these recommendations are intended to convey how an issue may be resolved (i.e., the idea), but they may not be tested or functional code. These recommendations are not exhaustive, and we encourage our partners to consider them as a starting point for further discussion. We are happy to provide additional guidance and advice as needed.

Finally, the contents of this assessment report are for informational purposes only; do not construe any information in this report as legal, tax, investment, or financial advice. Nothing contained in this report constitutes a solicitation or endorsement of a project by Zellic.