AUSD0 PSM Security Audit

Released December 5, 2025

Performed By

Carter Snay	csnay@iol.unh.edu
James Kahn	jkahn@iol.unh.edu
Oliver Willis	owillis@iol.unh.edu

University of New Hampshire
Interoperability Labs

+1-603-862-0090 | www.iol.unh.edu

Table of Contents

1 Legal Notice

The Interoperability Labs Blockchain team makes every effort to identify as many vulnerabilities in the code as possible within the given time period but assumes no responsibility for the findings presented in this document. A security audit by the team does not constitute an endorsement of the underlying business or product. The audit was time-boxed, and the review focused solely on the security aspects of the Solidity implementation of the contracts.

2 Executive Summary

The UNH Interoperability Labs conducted a security assessment for AUSD0 PSM from December 3, 2024 to December 5, 2024. During this assessment, The UNH Interoperability Labs reviewed AUSD0 PSM code for security vulnerabilities, design issues and general weaknesses.

During this initial assessment, 2 informational findings were identified by the team.

2.1 About LayerZero Labs and AUSD0 PSM

LayerZero is an omnichain interoperability protocol designed to facilitate seamless messaging of arbitrary data between blockchains. It achieves this by leveraging a combination of on-chain endpoints, a decentralized network of verifiers, and executors to securely transmit messages across chains. The protocol enables cross-chain applications to maintain atomicity and composability across multiple networks.

The AUSD0 PSM is part of a smart contract system that enables atomic swaps and cross-chain bridging of AUSD using LayerZero's omnichain messaging. It integrates with Agora Stable Swap on Ethereum to allow users to convert USDC/USDT into AUSD and transfer it to other chains, or reverse the process by sending AUSD back to Ethereum and swapping it into USDC/USDT. This design ensures permissionless, token-agnostic operations while maintaining robust retry and refund mechanisms for failed transactions.

2.2 Review Timeline

- December 3, 2024: Initial Audit Scope Review
- December 4, 2025: Manual Review Begins
- December 5, 2025: Draft Report & Final Report Delivered

2.3 Scope

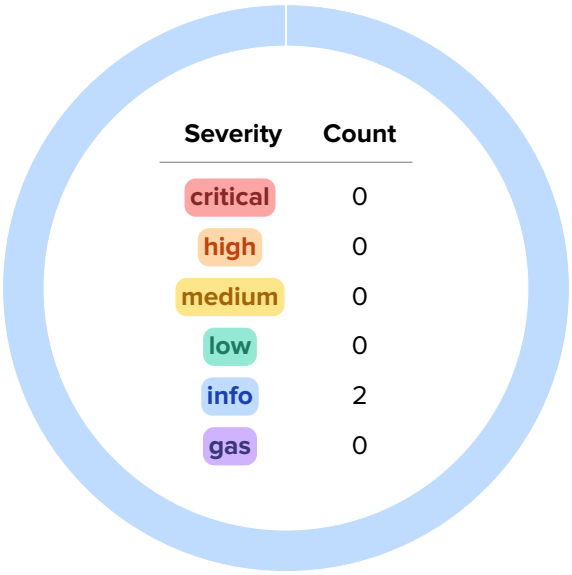
Project Name	AUSD0 PSM		
URL	`https://github.com/LayerZero-Labs/ausd0-internal`		
Language	Solidity		
Scope	Repo	`https://github.com/LayerZero-Labs/ausd0-internal/`	
	Hash	~~3ec5bb3c1e592568e8636dbaf2fb520b91fc0952~~	Oct 28, 2025
		~~e2f2687910fe3bc17b3bb742c0e6d0a6b7d223fe~~	Dec 4, 2025
		d3970249bf186794cdf451c119660aca3dc13a77	Dec 5, 2025

2.3.1 Files in Scope

```
contracts/
├── interfaces/
│   └── IPsmSwapAdapter.sol
└── PsmSwapAdapter.sol
```

3 Vulnerabilities

In summary, we identified two informational findings.



Severity	Count
critical	0
high	0
medium	0
low	0
info	2
gas	0

3.1 Findings Summary

ID	Severity	Title	Status
I-01	info	Comment fixes in `PsmSwapAdapter.sol`	RESOLVED
I-02	info	Comment fixes in `PsmSwapAdapter.sol`	RESOLVED

3.2 Detailed Findings

3.2.1 Info

[I-01] Comment fixes in `PsmSwapAdapter.sol`

Category	Target
Comment Inconsistency	`PsmSwapAdapter.sol`

Description

Incorrect formatting in both code comment and NatSpec.

Recommended mitigation

PsmSwapAdapter.sol — diff

```
55     // Approve OFT to spend max amount of OFT tokens in order to prevent
56 -   // USDT-style approve approve failures if non-zero allowance is left on the OFT due to "dust" removal.
57 +   // USDT-style approve failures if non-zero allowance is left on the OFT due to "dust" removal.
58     // The rest of the code now assumes that OFT has enough allowance granted by this contract.
59     // Ignore whether OFT requires approval or not, as sometimes people misconfigure it.
```

PsmSwapAdapter.sol — diff

```
207     /**
208      * @dev Internal helper that executes a swap of tokens via Agora Stable Swap PSM.
209 +    * @param _tokenIn The input token address
210 +    * @param _tokenOut The output token address
211      * @param _to The address to transfer tokens to
212      * @param _amountIn The amount of tokens to swap
213      * @param _amountOutMin The minimum amount of tokens to receive
214 -    * @param _tokenIn The input token address
215 -    * @param _tokenOut The output token address
216      * @param _deadline The deadline for the swap operation (timestamp)
217      * @return amountOut The amount of tokens received
```

Resolution

LayerZero has acknowledged this, and has resolved these typos in the following commit:

e2f2687910fe3bc17b3bb742c0e6d0a6b7d223fe

[I-02] Comment fixes in `PsmSwapAdapter.sol`

Category	Target
Comment Inconsistency	`PsmSwapAdapter.sol`

Description

Incorrect formatting in code comment and NatSpec.

`minAmountLD is set to 0` is not enforced by the implementation.

Recommended mitigation

PsmSwapAdapter.sol diff

```
137    function _handleCompose(
138         bytes32 _guid,
139         uint256 _amountIn,
140         bytes memory _composeMsg,
141         address _executor,
142         bytes calldata _tokenOutBytes
143     ) internal returns (uint256 amountOut) {
144 -       // ComposeMsg is encodePacked of three addresses, it's 60 bytes: 20 each.
145 +       // _composeMsg is encodePacked of three addresses, it's 60 bytes: 20 each.
146         // Parse directly from bytes using assembly. No left-padding (addresses are packed).
147         address token;
148         address to;
149         address withdrawer;
```

PsmSwapAdapter.sol diff

```
248      /**
249       * @dev Internal helper that constructs the parameters needed for cross-chain token transfers.
250 -     * minAmountLD is set to 0 because it plays the same role as _swapParam.amountOutMin.
251       * OFT tokens use 6 decimals on all chains, hence we will not remove any dust on funds received after
         the swap.
252       * If caller wants to restrict minimum amount to transfer, they can set it to _swapParam.amountOutMin.
253       * @param _swapParam The swap parameters containing destination information
254       * @param _amountLD The amount to send (in local decimals)
255       * @param _extraOptions Additional LayerZero options
256       * @return sendParam The constructed SendParam for the OFT operation
257       */
```

Resolution

LayerZero has acknowledged this, and has resolved these typos in the following commit:

d3970249bf186794cdf451c119660aca3dc13a77

Appendix A: Our Methodology

The Interoperability Labs audit team follows a comprehensive methodology in ensuring the security and reliability of smart contracts and Web3 protocols. While the specific testing procedures performed vary between the project and protocol, the tooling and manual review process remains the same to ensure thorough analysis has been completed on all items within the defined scope of the audit. Throughout the security review, the audit team maintains communication with the development team, providing feedback on identified vulnerabilities and optimizations. The following sections provide an overview of our systematic audit process and methodology.

Risk Classification

Impact →

Likelihood ↓

	Informational	Low	Medium	High	Critical
Very Unlikely	Info	Low	Low	Medium	Critical
Unlikely	Info	Low	Low	Medium	Critical
Possible	Info	Low	Medium	High	Critical
Likely	Info	Low	Medium	High	Critical
Very Likely	Low	Medium	High	Critical	Critical

We use the PricewaterhouseCoopers-style matrix to provide comprehensive risk assessment. See the documentation for more details.

Review Phases

Phase I: Initial Scoping

- Independent review of project documentation to understand the business logic of the project.
- Identification of critical components and key areas of focus and possible areas of exploitation.
- Ensure that the project's documentation is accurate, complete, understandable, and there is alignment between the code and the documentation.
- Discussion with the development team to clarify objectives, expectations, and known issues.

Phase II: Codebase Review

- **Static Analysis**: Automated tools to scan for common vulnerabilities with Slither and Aderyn.
- **Manual Review**: In-depth inspection of the code by the audit team to identify issues, including unsafe coding practices from known previous exploits.
- **Function State Machine Diagramming**: Generate a flow diagram illustrating the intended transaction paths, as well as unintended and potentially exploitable paths.

Phase III: Local Testing

Methods:

- **Unit Testing**: Validate individual functions for correctness.
- **Integration Testing**: Ensure that different components interact as expected.

Techniques Used in This Audit:

- **Unit Testing**
- **Manual Review**
- **Function State Machine Diagramming**

Proof of Vulnerability

Objective: Prove how a found exploit can be executed.

Activities:

- Perform controlled attacks on a local fork of the protocol to show how an exploit can be executed.
- Test edge cases and unexpected scenarios discovered in the diagramming phase.

By following a structured and comprehensive methodology, we aim to provide actionable insights to strengthen the security and reliability of the protocol. This ensures security, resilience, and long-term success for the protocol.

Appendix B: The Interoperability Labs

The University of New Hampshire Interoperability Labs (UNH-IOL) is the foremost independent testing facility for data networking companies worldwide.

We accelerate the launch of innovative products by providing standards and compliance testing to ensure that devices meet industry standards. Whether it's traditional Ethernet or advanced technologies like 5G, blockchain, and autonomous vehicles, our services provide comprehensive testing for device interoperability, conformance, and certification.

Our state-of-the-art laboratory and 36 years of extensive experience make it a strategic resource for industry startups and Fortune 500 companies needing collaboration, innovation, and standards development to help them shape the future of networking.

Join us and become a part of a community driving the next generation of networking technology. Together, we can shape the future of networking.

https://www.iol.unh.edu/membership

University of New Hampshire Interoperability Laboratory

21 Madbury Rd., Ste 100 Durham, NH 03824-4716

+1-603-862-0090 | www.iol.unh.edu

Contact our Blockchain Team: `blockchain@iol.unh.edu`