Security Review Report
NM-0161 ZendToken

NETHERMIND SECURITY

(Nov 24, 2023)

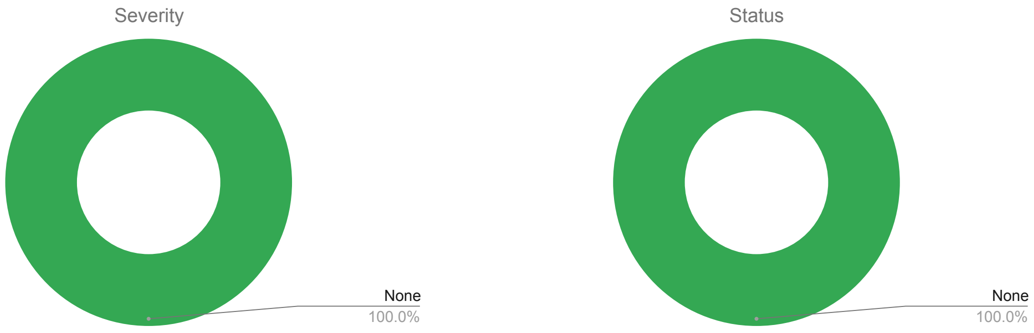
1 Executive Summary

This document presents the security review performed by Nethermind on the Zklend ZendToken. ZkLend is a Starknet native lending protocol where users can deposit assets as collateral and engage in overcolleralized borrowing. The ZendToken will be used as the governance token for the ZkLend protocol. It is a token compliant with the ERC20 specification and has been deployed on the Ethereum network. The token contract consists of a `TransparentUpgradeableProxy` and the underlying token implementation.

Throughout the investigation of the codebase, no issues have been identified.

The audit was performed using (a) manual analysis of the codebase, (b) automated analysis tools, (c) simulation of the smart contract, and (d) creation of test cases. Along this document, we report 0 Points of attention. The issues are summarized in Fig. 1.

This document is organized as follows. Section 2 presents the files in the scope of this audit. Section 3 presents verification of token behavior. Section 4 presents deployment details. Section 5 discusses the risk rating methodology adopted for this audit. Section 6 details the issues. Section 7 concludes the document.



(a)

(b)

Fig. 1: Distribution of issues: Critical (0), **High** (0), **Medium** (0), **Low** (0), **Undetermined** (0), **Informational** (0), **Best Practices** (0). **Distribution of status: Fixed** (0), **Acknowledged** (0), **Mitigated** (0), **Unresolved** (0), **Partially Fixed** (0)

Summary of the Audit

Audit Type	Security Review
Initial Report	Nov 24, 2023
Response from Client	-
Final Report	-
Methods	Manual Review, Automated Analysis
Repository	github.com/zkLend/zend-token
Commit Hash (Audit)	b55be38a544db2266c7d08f68a3261610b302d6a

2 Contracts

NOTE: This line count only represents the code written by the ZkLend team. When this token logic is flattened and the code written by the Openzeppelin team for the underlying ERC20 implementation is considered, the line count is 920.

	Contract	Lines of Code	Lines of Comments	Comments Ratio	Blank Lines	Total Lines
1	`src/ZendToken.sol`	9	1	7.1%	4	14
	Total	**9**	**1**	**7.1%**	**4**	**14**

3 Summary of Findings

After careful reveiw, no issues have been found in the codebase.

4 Verifying Token Behavior

To verify that the ZendToken will follow the intended behavior outlined in the ERC20 specification, a comprehensive test suite has been used. This test suite tests all possible ERC20 behaviors, including transfers (both `transfer` and `transferFrom`), balances and allowances. These tests are compatible with fuzzing, and each test has been simulated 40,000 times with random inputs to identify potential edge cases. All tests have passed, and are shown below:

```
Running 21 tests for test/ZendToken.t.sol:ZendTest
[PASS] testApprove(address,uint256) (runs: 40000, : 33382, ~: 34175)
[PASS] testApproveClear(address,uint256) (runs: 40000, : 25146, ~: 25463)
[PASS] testApproveEvent(address,uint256) (runs: 40000, : 34266, ~: 35077)
[PASS] testBalanceOfAtDeployment() (gas: 7767)
[PASS] testBalanceOfEmpty(address) (runs: 40000, : 10786, ~: 10786)
[PASS] testFailDoubleInitialize() (gas: 8083)
[PASS] testFailTransferFromMoreThanAllowance(address,address,uint256,uint256) (runs: 40000, : 32494, ~: 36956)
[PASS] testFailTransferFromMoreThanBalance(address,address,uint256,uint256) (runs: 40000, : 28347, ~: 40225)
[PASS] testFailTransferMoreThanBalance(address,uint256) (runs: 40000, : 12575, ~: 12575)
[PASS] testTotalSupply() (gas: 7608)
[PASS] testTransferEntireBalance(address) (runs: 40000, : 35445, ~: 35445)
[PASS] testTransferEvent(address,uint256) (runs: 40000, : 40004, ~: 41880)
[PASS] testTransferFromEntireBalance(address,address) (runs: 40000, : 51821, ~: 51819)
[PASS] testTransferFromEvent(address,uint256) (runs: 40000, : 39001, ~: 40914)
[PASS] testTransferFromIncreasesBalance(address,address,uint256) (runs: 40000, : 68296, ~: 70534)
[PASS] testTransferFromReducesApproval(address,address,uint256) (runs: 40000, : 68736, ~: 71026)
[PASS] testTransferFromReducesBalance(address,address,uint256) (runs: 40000, : 68408, ~: 70598)
[PASS] testTransferFromSelf() (gas: 30051)
[PASS] testTransferIncreasesBalance(address,uint256) (runs: 40000, : 38926, ~: 40866)
[PASS] testTransferReducesBalance(address,uint256) (runs: 40000, : 39200, ~: 41114)
[PASS] testTransferSelf(uint256) (runs: 40000, : 15221, ~: 15221)
Test result: ok. 21 passed; 0 failed; 0 skipped; finished in 19.35s
```

5 Deployment Details

The token has been deployed at address `0xb2606492712D311be8f41d940AFE8CE742A52D44` on Ethereum mainnet. This is the address of the proxy, which delegates all logic to the token implementation contract. The ZendToken implementation logic is stored at the address `0xa9a53bec7e830766668c2329db5444c12186b005`.

Upon deployment the function `__ZendToken_init` was called, which is a function that is only callable once. This minted 100,000,000 tokens (with 18 decimals) to a trusted Zklend owned address for later distribution. It is only possible for tokens to be minted through the function `__ZendToken_init`, meaning that no more tokens can be minted as long as the proxy implementation does not change.

The bytecode of the deployed implementation can be verified to match the compiled code provided in the repo by stripping Solidity version and IPFS metadata, then hashing both resulting binaries.

- **Deployed** `0x60efa49af5957756f5d26915bd66467a9e0b2d929024798ef397404af2ba4ea8`
- **Compiled** `0x60efa49af5957756f5d26915bd66467a9e0b2d929024798ef397404af2ba4ea8`

Given both runtime binaries match, it is proven that the code provided in the repository is the same as the code that is deployed on-chain.

6 About Nethermind

Nethermind is a Blockchain Research and Software Engineering company. Our work touches every part of the web3 ecosystem - from layer 1 and layer 2 engineering, cryptography research, and security to application-layer protocol development. We offer strategic support to our institutional and enterprise partners across the blockchain, digital assets, and DeFi sectors, guiding them through all stages of the research and development process, from initial concepts to successful implementation.

We offer security audits of projects built on EVM-compatible chains and Starknet. We are active builders of the Starknet ecosystem, delivering a node implementation, a block explorer, a Solidity-to-Cairo transpiler, and formal verification tooling. Nethermind also provides strategic support to our institutional and enterprise partners in blockchain, digital assets, and decentralized finance (DeFi). In the next paragraphs, we introduce the company in more detail.

Blockchain Security: At Nethermind, we believe security is vital to the health and longevity of the entire Web3 ecosystem. We provide security services related to Smart Contract Audits, Formal Verification, and Real-Time Monitoring. Our Security Team comprises blockchain security experts in each field, often collaborating to produce comprehensive and robust security solutions. The team has a strong academic background, can apply state-of-the-art techniques, and is experienced in analyzing cutting-edge Solidity and Cairo smart contracts, such as ArgentX and StarkGate (the bridge connecting Ethereum and StarkNet). Most team members hold a Ph.D. degree and actively participate in the research community, accounting for 240+ articles published and 1,450+ citations in Google Scholar. The security team adopts customer-oriented and interactive processes where clients are involved in all stages of the work.

Blockchain Core Development: Our core engineering team, consisting of over 20 developers, maintains, improves, and upgrades our flagship product - the Nethermind Ethereum Execution Client. The client has been successfully operating for several years, supporting both the Ethereum Mainnet and its testnets, and now accounts for nearly a quarter of all synced Mainnet nodes. Our unwavering commitment to Ethereum's growth and stability extends to sidechains and layer 2 solutions. Notably, we were the sole execution layer client to facilitate Gnosis Chain's Merge, transitioning from Aura to Proof of Stake (PoS), and we are actively developing a full-node client to bolster Starknet's decentralization efforts. Our core team equips partners with tools for seamless node set-up, using generated docker-compose scripts tailored to their chosen execution client and preferred configurations for various network types.

DevOps and Infrastructure Management: Our infrastructure team ensures our partners' systems operate securely, reliably, and efficiently. We provide infrastructure design, deployment, monitoring, maintenance, and troubleshooting support, allowing you to focus on your core business operations. Boasting extensive expertise in Blockchain as a Service, private blockchain implementations, and node management, our infrastructure and DevOps engineers are proficient with major cloud solution providers and can host applications in-house or on clients' premises. Our global in-house SRE teams offer 24/7 monitoring and alerts for both infrastructure and application levels. We manage over 5,000 public and private validators and maintain nodes on major public blockchains such as Polygon, Gnosis, Solana, Cosmos, Near, Avalanche, Polkadot, Aptos, and StarkWare L2. Sedge is an open-source tool developed by our infrastructure experts, designed to simplify the complex process of setting up a proof-of-stake (PoS) network or chain validator. Sedge generates docker-compose scripts for the entire validator set-up based on the chosen client, making the process easier and quicker while following best practices to avoid downtime and being slashed.

Cryptography Research: At Nethermind, our Cryptography Research team is dedicated to continuous internal research while fostering close collaboration with external partners. The team has expertise across a wide range of domains, including cryptography protocols, consensus design, decentralized identity, verifiable credentials, Sybil resistance, oracles, and credentials, distributed validator technology (DVT), and Zero-knowledge proofs. This diverse skill set, combined with strong collaboration between our engineering teams, enables us to deliver cutting-edge solutions to our partners and clients.

Smart Contract Development & DeFi Research: Our smart contract development and DeFi research team comprises 40+ world-class engineers who collaborate closely with partners to identify needs and work on value-adding projects. The team specializes in Solidity and Cairo development, architecture design, and DeFi solutions, including DEXs, AMMs, structured products, derivatives, and money market protocols, as well as ERC20, 721, and 1155 token design. Our research and data analytics focuses on three key areas: technical due diligence, market research, and DeFi research. Utilizing a data-driven approach, we offer in-depth insights and outlooks on various industry themes.

Our suite of L2 tooling: Warp is Starknet's approach to EVM compatibility. It allows developers to take their Solidity smart contracts and transpile them to Cairo, Starknet's smart contract language. In the short time since its inception, the project has accomplished many achievements, including successfully transpiling Uniswap v3 onto Starknet using Warp.

- **Voyager** is a user-friendly Starknet block explorer that offers comprehensive insights into the Starknet network. With its intuitive interface and powerful features, Voyager allows users to easily search for and examine transactions, addresses, and contract details. As an essential tool for navigating the Starknet ecosystem, Voyager is the go-to solution for users seeking in-depth information and analysis;

- **Horus** is an open-source formal verification tool for StarkNet smart contracts. It simplifies the process of formally verifying Starknet smart contracts, allowing developers to express various assertions about the behavior of their code using a simple assertion language;

- **Juno** is a full-node client implementation for Starknet, drawing on the expertise gained from developing the Nethermind Client. Written in Golang and open-sourced from the outset, Juno verifies the validity of the data received from Starknet by comparing it to proofs retrieved from Ethereum, thus maintaining the integrity and security of the entire ecosystem.

Learn more about us at nethermind.io.

General Advisory to Clients

As auditors, we recommend that any changes or updates made to the audited codebase undergo a re-audit or security review to address potential vulnerabilities or risks introduced by the modifications. By conducting a re-audit or security review of the modified codebase, you can significantly enhance the overall security of your system and reduce the likelihood of exploitation. However, we do not possess the authority or right to impose obligations or restrictions on our clients regarding codebase updates, modifications, or subsequent audits. Accordingly, the decision to seek a re-audit or security review lies solely with you.

Disclaimer

This report is based on the scope of materials and documentation provided by you to Nethermind in order that Nethermind could conduct the security review outlined in **1. Executive Summary** and **2. Audited Files**. The results set out in this report may not be complete nor inclusive of all vulnerabilities. Nethermind has provided the review and this report on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. This report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on this report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, Nethermind disclaims any liability in connection with this report, its content, and any related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. Nethermind does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and Nethermind will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.