SECURITY AUDIT OF

ANCIENT8 NATIVE BRIDGE

Public Report

Sep 11, 2024

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
Ether (ETH)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
ERC20	ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain.

EXECUTIVE SUMMARY

This Security Audit Report was prepared by Verichains Lab on Sep 11, 2024. We would like to thank the Ancient8 for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the Ancient8 Native Bridge. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified no vulnerable issue in the smart contracts code.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About Ancient8 Native Bridge

The Ancient8 Native Bridge provides a seamless way for users to transfer ETH, A8, and a variety of other tokens between the Ethereum network and the Ancient8 Mainnet. By utilizing the OP Stack Standard, this bridge significantly enhances both the scalability and security of the transactions, ensuring efficient cross-chain transfers while maintaining a high level of protection for assets. This solution is designed to optimize the user experience by facilitating smooth, reliable token movements across different blockchain ecosystems.

1.2. Audit Scope

This audit focused on identifying security flaws in code and the design of the Ancient8 Native Bridge that was deployed on Ancient8 Network.

The latest version was made available in the course of the review:

FIELD	VALUE
Contract Name	L2StandardBridgeCustom
Address Deploy	0xE7877F027134DCD7eEB4e0217b5e699F980cc970
Tx Deploy	0x2bed1cde5cfd4fec0dc50208c8191cecfe5150d96ec90eba1373590e81b7948c
Deployer	0xf1cEa1A3301fE0c4B59A7B367d7214f0A782C238
Block Number	9491318

Table 1. The L2StandardBridgeCustom contract

FIELD	VALUE
Contract Name	OptimismMintableERC20 (Ancient8 Token)
Address Deploy	0xD812d616A7C54ee1C8e9c9CD20D72090bDf0d424
Tx Deploy	0xbfd86dd9dc089fb37b6809c4739715c670b6bacece9720bc830fc77b079ed461
Deployer	0xf1cEa1A3301fE0c4B59A7B367d7214f0A782C238
Block Number	9708205

Table 2. The Ancient8 Token contract

1.3. Audit Methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that were considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 3. Severity levels

1.4. Disclaimer

Ancient8 acknowledges that the security services provided by Verichains, are conducted to the best of their professional abilities but cannot guarantee 100% coverage of all security vulnerabilities. Ancient8 understands and accepts that despite rigorous auditing, certain vulnerabilities may remain undetected. Therefore, Ancient8 agrees that Verichains shall not be held responsible or liable, and shall not be charged for any hacking incidents that occur due to security vulnerabilities not identified during the audit process.

1.5. Acceptance Minute

This final report served by Verichains to the Ancient8 will be considered an Acceptance Minute. Within 7 days, if no any further responses or reports is received from the Ancient8, the final report will be considered fully accepted by the Ancient8 without the signature.

2. AUDIT RESULT

2.1. Overview

The Ancient8 Native Bridge was written in `Solidity` language, with the deployed compiler version is `0.8.26`.

The contract extends the `Ownable2Step` to manage access control and ownership.

2.1.1. L2StandardBridgeCustom

The `L2StandardBridgeCustom` contract facilitates the withdrawal of ETH and ERC-20 tokens from a Layer 2 (L2) network back to a Layer 1 (L1) network. It provides an owner-controlled mechanism for setting and managing transaction fees while ensuring smooth withdrawal processes via an external bridge. The contract is designed to handle both native ETH and ERC-20 token transfers, providing flexibility for users who operate on Layer 2 networks.

Actors in the contract:

- Owner: The owner (set at deployment and can be transferable) has the authority to set the transaction fees (`proveFee` and `finalizeFee`) through the `setFee` function. This ensures control over the cost of withdrawal operations.
- Users: Users can withdraw their ETH or ERC-20 tokens from L2 to L1 via the `withdraw` function. To perform a withdrawal, users are required to pay the total transaction fee (defined by the sum of `proveFee` and `finalizeFee`). If withdrawing ERC-20 tokens, the user must provide approval and transfer the tokens to the contract before the withdrawal process is initiated.

2.1.2. Ancient8 Token (A8)

The **Ancient8 Token** contract, **OptimismMintableERC20**, is an extended ERC-20 token implementation designed for use in L2 scaling solutions, specifically on the Ancient8 network. It allows the bridging and minting of tokens between the L1 and L2 networks.

The Standard Bridge can mint tokens without any limit, while other minters are subject to a capped amount, intended to accommodate future third-party bridges. Minters, including the Bridge, are only allowed to burn tokens up to the amount they have minted. Additionally, the contract allows for adding, removing, or modifying the permissions of minters by the owner.

The Token was deployed on Ancient8 network with following properties:

PROPERTY	VALUE
Name	Ancient8
Symbol	A8
Decimals	18
Mintable	True
Standard Bridge	0x4200000000000000000000000000000000000010
Remote Token (L1)	0x3E5A19c91266aD8cE2477B91585d1856B84062dF

Table 4. The Ancient8 Native Bridge properties

2.2. Findings

During the audit process, the audit team found no vulnerability in the given version of Ancient8 Native Bridge.

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Sep 11, 2024*	Public Report	Verichains Lab

Table 5. Report versions history