Zulu Network Smart Contract

Audit Report

Tue Jul 16 2024

contact@bitslab.xyz https://twitter.com/scalebit_

ScaleBit

Zulu Network Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	A cross-chain bridge project.
Type	DeFi
Auditors	ScaleBit
Timeline	Thu Jul 04 2024 - Thu Jul 11 2024
Languages	Solidity
Platform	Ethereum
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/zulu-network/l2-bridge-contracts
Commits	9d90a2cc2a5485924f1ff09d0398f0dad44a996d 2aa56944523cc78fd63fa5c63a1643e4fe070d5f

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
ZULUL2BERC2	contracts/ZULUL2BridgeERC20.sol	76d1eb592abfb3e11ed6c06f7df3c06c91f0d128
ZULUL2BERC7	contracts/ZULUL2BridgeERC721.sol	8d0af28af7216fdf2c0fbc46116056ce2278dc57
ZULUL2B	contracts/ZULUL2Bridge.sol	89a42fbd3a020ac841d1a13b2eed6572e5c23396
IZULUL2BERC2	contracts/interfaces/IZULUL2BridgeERC20.sol	1164b7305d6f119ed4ce7a9635d4faebbb3b348b
IZULUL2BERC7	contracts/interfaces/IZULUL2BridgeERC721.sol	e3e7119c9cc52fdaf65cc2d8cc0477d3f21375d1
ERC2TW	contracts/ERC20TokenWrapped.sol	743797d05250f3f00d8a82deb3fc042974beb436
ERC7TW	contracts/ERC721TokenWrapped.sol	5ed020cb03bf44a29429559f6f0aef6a49ac7d11

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	4	3	1
Informational	2	2	0
Minor	0	0	0
Medium	1	1	0
Major	1	0	1
Critical	0	0	0

1.4 ScaleBit Audit Breakdown

ScaleBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Unchecked External Call
- Unchecked CALL Return Values
- Functionality Checks
- Reentrancy
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic issues
- Gas usage
- Fallback function usage
- tx.origin authentication
- Replay attacks
- Coding style issues

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by Zulu Network to identify any potential issues and vulnerabilities in the source code of the Zulu Network smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 4 issues of varying severity, listed below.

ID	Title	Severity	Status
ERC-1	Code Optimization	Informational	Fixed
IZU-1	Gas Optimization	Informational	Fixed
ZUL-1	Centralization Risk	Major	Acknowledged
ZUL-2	Confirm that The Burn Method in The Token that Supports Works as Expected	Medium	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the Zulu Network Smart Contract :

SuperAdmin

- The `SuperAdmin` can set the `SuperAdmin` address through `setSuperAdminAddress()` .
- The `SuperAdmin` can set the `BurnAdmin` address through `setBurnAdminAddress()` .
- The `SuperAdmin` can set the `MintAdmin` address through `setMintAdminAddress()` .
- The `SuperAdmin` can set the `SettingAdmin` address through `setSettingAdminAddress()` .
- The `SuperAdmin` can pause and unpause the contract through `pause()` and `unpause()` .

MintAdmin

- The `MintAdmin` can mint `ERC20` tokens through `mintERC20Token()` .
- The `MintAdmin` can mint `ERC721` tokens through `batchMintERC721Token()` .
- The `MintAdmin` can transfer native tokens to any address from the contract through `unlockNativeToken()` .

BurnAdmin

- The `BurnAdmin` can set the `burnGlobalEnabled` status of the contract through `enableBurnGlobal()` and `disableBurnGlobal()` .
- The `BurnAdmin` can set the `burnTokenEnabled` status of the token through `enableBurnToken()` and `disableBurnToken()` .

SettingAdmin

- The `SettingAdmin` can set the `BaseURI` of the `ERC721` token through `setBaseURI` .
- The `SettingAdmin` can set `bridgeFee` through `setBridgeSettingsFee()` .

User

- The `User` can burn their `ERC20` tokens through `burnERC20Token()` .
- The `User` can burn their `ERC721` tokens through `batchBurnERC721Token` .
- The `User` can lock their native tokens in the contract through `lockNativeToken()` .

4 Findings

ERC-1 Code Optimization

Severity: Informational

Status: Fixed

Code Location:

contracts/ERC721TokenWrapped.sol#43

Descriptions:

In the `mint` function, when checking the length of `bytes(mpTokenId2InscriptionId[tokenId])`, since its length will not be less than 0, you can check if its length is 0.

```
require(
    bytes(mpTokenId2InscriptionId[tokenId]).length <= 0,
    "tokenId is repeat"
);
```

Suggestion:

It is recommended to modify the code as :

```
require(
    bytes(mpTokenId2InscriptionId[tokenId]).length == 0,
    "tokenId is repeat"
);
```

Resolution:

This issue has been fixed. The client has adopted our suggestions.

IZU-1 Gas Optimization

Severity: Informational

Status: Fixed

Code Location:

contracts/interfaces/IZULUL2BridgeERC20.sol#63

Descriptions:

In the `mintERC20Token` function, using `erc20TxHashUnlocked[txHash] == false` consumes more gas, whereas using `!erc20TxHashUnlocked[txHash]` generates less bytecode, reducing gas consumption and making the code more concise and efficient.

```
require(
        erc20TxHashUnlocked[txHash] == false,
        "Transaction has been executed"
    );
```

Suggestion:

It is recommended to change the conditional logic to `!erc20TxHashUnlocked[txHash]`.

```
require(
        !erc20TxHashUnlocked[txHash],
        "Transaction has been executed"
    );
```

Resolution:

This issue has been fixed. The client has adopted our suggestions.

ZUL-1 Centralization Risk

Severity: Major

Status: Acknowledged

Code Location:

contracts/ZULUL2Bridge.sol#143,187,225,271

Descriptions:

Centralization risk was identified in the smart contract.

1. The Admin can mint or burn tokens and nfts to the cap.
2. The Admin can transfer any amount of `NativeToken` from the contract to any address.

Suggestion:

It's recommended that measures be taken to reduce the centralization issue like using multi-sig.

ZUL-2 Confirm that The Burn Method in The Token that Supports Works as Expected

Severity: Medium

Status: Fixed

Code Location:

contracts/ZULUL2Bridge.sol

Descriptions:

We noticed that the `burn()` function of the on-chain token is called during cross-chaining and then releases the cross-chain event, please make sure that the burn method in the token contract that supports cross-chaining works as you expect it to. For example, if a token's burn method is not implemented or requires administrator privileges, this may prevent the cross-chain from working properly.

Suggestion:

It is recommended to check that a token contract that supports cross-chain correctly implements the burn method.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided 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 own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.