OpenZeppelin | security

Scroll USDC Gateway Audit

September 13, 2023

Table of Contents

Summary

Type	zkEVM-based ZK-rollup, Bridge & Rollup
Timeline	From 2023-08-25 To 2023-08-30
Languages	Solidity

Total Issues	5 (4 resolved, 1 partially resolved)
Critical Severity Issues	0 (0 resolved)
High Severity Issues	0 (0 resolved)
Medium Severity Issues	1 (1 resolved)
Low Severity Issues	3 (3 resolved)
Notes & Additional Information	1 (0 resolved, 1 partially resolved)
Client Reported Issues	0 (0 resolved)

Scope

We audited the USDC Gateway changes from the scroll-tech/scroll repository at commit f6894bb.

```
contracts
└── src
    ├── L1
    │   ├── gateways
    │   │   ├── L1ERC20Gateway.sol
    │   │   └── usdc
    │   │       └── L1USDCGateway.sol
    ├── L2
    │   ├── gateways
    │   │   └── usdc
    │   │       └── L2USDCGateway.sol
    ├── interfaces
    │   ├── L2USDCGateway.sol
    │   ├── ITokenMessenger.sol
    │   ├── IMessangerTransmitter.sol
    │   ├── IUSDCBurnableSourceBrdge.sol
    │   ├── IUSDCDestinationBridge.sol
    └── libraries
        └── gateway
            └── CCTPGatewayBase.sol
```

System Overview

Scroll is an EVM-equivalent ZK-rollup designed to be a scaling solution for Ethereum. It achieves this by interpreting EVM bytecode directly at the bytecode level, following a similar path to projects like Polygon's zkEVM and Consensys' Linea.

This audit reviewed the extension of the special USDC Gateway from the scroll protocol.

This report presents our findings and recommendations for the new additions to the Scroll ZK-rollup protocol. We urge the Scroll team to consider these findings in their ongoing efforts to provide a secure and efficient Layer 2 solution for Ethereum.

Trust Assumptions

It is assumed that the `USDC` contract to be deployed in the Scroll Layer 2 Network will be identical to the one deployed in the Ethereum Mainnet.

Privileged Roles

Certain privileged roles within the Scroll protocol were identified during the audit. These roles possess special permissions that could potentially impact the system's operation:

- **Access Control:** The access control manager is a contract in which there is an address with default admin privileges that can perform the critical administrative action of giving and revoking roles for different addresses. This mechanism is used in the following contracts:
  - `ScrollOwner`: The default admin role can grant roles for addresses to execute functions through the contract. Every role will be associated with a designated set of functions tied to specific addresses permissible to execute within that role. Moreover, existing roles will come with execution delays ranging from 0 days for instant execution to 1 day, 7 days, and 14 days. This provides a dynamic control

mechanism over the timing of function execution based on their respective impact levels.
    - `TokenRateLimiter`: The default admin role can update the total token amount limit. The admin can also grant a token spender role for the Scroll gateways and messengers to ensure a rate limit when depositing or withdrawing funds.

- **Implementation Owners:** Most contracts are also ownable. The following actions describe what the owner can do in each contract.
    - `L1ScrollMessenger`: Pause the relay of L2 to L1 messages and L1 to L2 message requests.
    - `EnforcedTxGateway`: Pause L1 to L2 transaction requests and change the fee vault.
    - `L1{CustomERC20|ERC721|ERC1155}Gateway`: Change the token mapping containing which L1 token is bridged to which L2 token.
    - `L1GatewayRouter`: Set the respective gateway for ETH, custom ERC-20s and default ERC-20s.
    - `ScrollMessengerBase`: Change the fee vault address which collects fees for message relaying.
    - `ScrollStandardERC20Factory`: Use the factory to deploy another instance of a standard ERC-20 token on L2.
    - `L2ScrollMessenger`: Pause the relay of L1 to L2 messages and L2 to L1 message requests.
    - `L2{CustomERC20|ERC721|ERC1155}Gateway`: Change the token mapping containing which L2 token is bridged to which L1 token.
    - `L2GatewayRouter`: Set the respective gateway for ETH, custom ERC-20s and default ERC-20s.

- **USDC**: The following roles are present in the `USDC` contract:
    - `owner`: This role can transfer ownership of the contract and grant or remove the `masterMinter`, `pauser` and `blacklister` roles.
    - `masterMinter`: This role can create new minters and assign allowances to existing minters.
    - `pauser`: This role has the ability to pause and unpause the contract.
    - `blacklister`: This role can add or remove addresses from a blacklist, which if added would prevent that address from transferring or receiving `USDC`.

- `minter`: This role allows the minting of tokens up to each minter's allowance.

Each of these roles presents a unique set of permissions within the Scroll protocol. The potential implications of these permissions warrant further consideration and mitigation to ensure the system's security and robustness.

Medium Severity

M-01 `L2USDCGateway` Is Missing Rate Limiter Functionality

The `L1USDCGateway` contract inherits from `L1ERC20Gateway`. When a user initiates a deposit, the `_transferERC20In` function is called, which in turn invokes the rate limiter function `_addUsedAmount`. However, the `L2USDCGateway` contract inherits from `L2ERC20Gateway` which does not call the rate limiter `_addUsedAmount` function. This means that USDC withdrawals will not be subject to rate limiting.

Consider ensuring that `_addUsedAmount` is called when users make a withdrawal in USDC.

Update: *Resolved in pull request #927 at commit be6d404.*

Low Severity

L-01 Misleading Comment

The comment in line 172 of the `L2USDCGateway` contract should say `L2ScrollMessenger` instead of `L1ScrollMessenger`.

Consider resolving this instance of incorrect documentation to improve the clarity and readability of the codebase.

Update: *Resolved in pull request #928 at commit 733d2a6.*

L-02 Lack of `gap` Variable

The `CCTPGatewayBase` contract does not contain a gap variable although it is upgradeable.

Consider adding a gap variable following OpenZeppelin's upgradeable contracts guide to avoid future storage collisions.

Update: *Resolved in pull request #929 at commit 5e61a05.*

L-03 Missing Docstrings

Throughout the codebase there are several parts that do not have docstrings. For instance:

- Line 17 in `L1ERC20Gateway.sol`
- Line 64 in `L2USDCGateway.sol`
- Line 5 in `IMessageTransmitter.sol`
- Line 6 in `IMessageTransmitter.sol`
- Line 5 in `ITokenMessenger.sol`
- Line 6 in `IUSDCBurnableSourceBridge.sol`
- Line 6 in `IUSDCDestinationBridge.sol`
- Line 9 in `CCTPGatewayBase.sol`

Consider thoroughly documenting all functions (and their parameters) that are part of any contract's public API. Functions implementing sensitive functionality, even if not public, should be clearly documented. When writing docstrings, consider following the Ethereum Natural Specification Format (NatSpec).

Update: *Resolved in pull request #940 at commit 30fa5e6.*

Notes & Additional Information

N-01 Unused Imports

Throughout the codebase there are imports that are unused and could be removed. For instance:

- Import `IScrollMessenger` of `L1ERC20Gateway.sol`
- Import `ScrollConstants` of `L1ERC20Gateway.sol`
- Import `OwnableUpgradeable` of `L1USDCGateway.sol`
- Import `IERC20Upgradeable` of `L1USDCGateway.sol`
- Import `IL1ERC20Gateway` of `L1USDCGateway.sol`
- Import `IL2ERC20Gateway` of `L2USDCGateway.sol`

- Import `IScrollGateway` of `L2USDCGateway.sol`

Consider removing unused imports to improve the overall clarity and readability of the codebase.

Update: *Partially resolved in pull request #930 at commit 23bf84a. `L1USDCGateway.sol` still imports `IL1ERC20Gateway` and `L2USDCGateway.sol` still imports `IL2ERC20Gateway`.*

Conclusion

Throughout this 4-day audit, we reviewed both L1 and L2 USDC gateways. We identified a single medium-severity issue, as well as a few low-severity issues and additional notes. Overall, we commend the quality and thoughtful integration of the USDC gateway. The auditing process was seamless, and we appreciate the Scroll team's prompt responses to our inquiries throughout the process.