BAILSEC.IO

OFFICE@BAILSEC.IO

X: @BAILSECURITY

TG: @HELLOATBAILSEC

FINAL REPORT

Euler

ERC20BurnableMintable

December 2024

Disclaimer:

Security assessment projects are time-boxed and often reliant on information that may be provided by a client, its affiliates, or its partners. As a result, the findings documented in this report should not be considered a comprehensive list of security issues, flaws, or defects in the target system or codebase.

The content of this assessment is not an investment. The information provided in this report is for general informational purposes only and is not intended as investment, legal, financial, regulatory, or tax advice. The report is based on a limited review of the materials and documentation provided at the time of the audit, and the audit results may not be complete or identify all possible vulnerabilities or issues. The audit is provided on an "as-is," "where-is," and "as-available" basis, and the use of blockchain technology is subject to unknown risks and flaws.

The audit does not constitute an endorsement of any particular project or team, and we make no warranties, expressed or implied, regarding the accuracy, reliability, completeness, or availability of the report, its content, or any associated services or products. We disclaim all warranties, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We assume no responsibility for any product or service advertised or offered by a third party through the report, 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. We will not be liable for any loss or damages incurred as a result of the use or reliance on the audit report or the smart contract.

The contract owner is responsible for making their own decisions based on the audit report and should seek additional professional advice if needed. The audit firm or individual assumes no liability for any loss or damages incurred as a result of the use or reliance on the audit report or the smart contract. The contract owner agrees to indemnify and hold harmless the audit firm or individual from any and all claims, damages, expenses, or liabilities arising from the use or reliance on the audit report or the smart contract.

By engaging in a smart contract audit, the contract owner acknowledges and agrees to the terms of this disclaimer.

1. Project Details

Important:
Please ensure that the deployed contract matches the source-code of the last commit hash.

Project	Euler - ERC20BurnableMintable
Website	app.euler.finance
Language	Solidity
Methods	Manual Analysis
Github repository	https://github.com/euler-xyz/evk-periphery/blob/b3ac5f126c6c0a0374374a6b7fbf18b4461f5496/src/ERC20/deployed/ERC20BurnableMintable.sol

2. Detection Overview

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High				
Medium				
Low				
Informational				
Governance	1			1
Total	1			1

2.1 Detection Definitions

Severity	Description
High	The problem poses a significant threat to the confidentiality of a considerable number of users' sensitive data. It also has the potential to cause severe damage to the client's reputation or result in substantial financial losses for both the client and the affected users.
Medium	While medium level vulnerabilities may not be easy to exploit, they can still have a major impact on the execution of a smart contract. For instance, they may allow public access to critical functions, which could lead to serious consequences.
Low	Poses a very low-level risk to the project or users. Nevertheless the issue should be fixed immediately
Informational	Effects are small and do not post an immediate danger to the project or users
Governance	Governance privileges which can directly result in a loss of funds or other potential undesired behavior

3. Detection

ERC20BurnableMintable

The ERC20BurnableMintable contract is a simple ERC20 token which inherits ERC20Permit and ERC20Burnable.

This allows users to use off-chain signatures for approval settings and burn tokens from their own address or from an address which has granted approvals.

Furthermore, the contract inherits OpenZeppelin's AccessControlEnumerable contract which handles role-based-access-control by determining one address with the DEFAULT_ADMIN_ROLE during the contract deployment.

The address with the DEFAULT_ADMIN_ROLE can then grant the MINTER_ROLE and REVOKE_MINTER_ROLE to other addresses.

The MINTER_ROLE has the privilege of minting tokens which means this address can alter the token supply at will, while the REVOKE_MINTER_ROLE can remove the MINTER_ROLE from an address again. Both roles are overseen by the address with the DEFAULT_ADMIN_ROLE, as this address can immediately revoke roles again via the revokeRole function.

The contract is meant to be used as the EUL multichain token by leveraging Wormhole's Native-Token-Transfers architecture (NTT), which serves as a bridge between different blockchains.

However, it can be considered as a general purpose ERC20 token which is compatible with Wormhole's NTT architecture.

More information about the Wormhole NTT architecture can be found here:

https://wormhole.com/docs/learn/messaging/native-token-transfers/overview/#

Deployed Blockchain: Multiple

Trust Assumptions: The following trust assumptions are made:

a) Any address with the DEFAULT_ADMIN_ROLE is a highly trusted and secured address. If any address with this role is ever compromised, that means all funds are at risk.
b) The address with the MINTER_ROLE must be the corresponding Wormhole contract which is responsible for minting tokens on a chain. If any address with this role is ever compromised, that means all funds are at risk.
c) The address with the REVOKE_MINTER_ROLE must serve as a fallback security address if under any scenario, the MINTER_ROLE must be revoked. This could be the case if Wormhole is suddenly compromised. If any address with this role is compromised, this can result in a temporary DoS for the bridge module.

Decimals: 18

Initial mint: -

Minting occasions: Minting must only occur if tokens are bridged from Chain A to Chain B which means tokens must be burned on Chain A and minted on Chain B.

Appendix: Core Invariants:

INV 1: The DEFAULT_ADMIN_ROLE must only be assigned to highly trusted addresses.

INV 2: The DEFAULT_ADMIN_ROLE must be the only role which can assign MINTER and REVOKE_MINTER roles.

INV 3: The MINTER_ROLE must be the Wormhole smart contract.

INV 4: Minting must only be allowed by the MINTER_ROLE.

INV 5: Minting must only occur if the token on the source chain has been burned or locked.

INV 6: revokeMinterRole must only be callable by addresses with the REVOKE_MINTER_ROLE.

Privileged Functions

- grantRole
- revokeRole
- mint
- revokeMinterRole

Issue_01	Governance: Centralization Risk
Severity	**Governance**
Description	The contract uses OpenZeppelin's AccessControlEnumerable contract for RBAC which exposes the following roles: DEFAULT_ADMIN_ROLE MINTER_ROLE REVOKE_MINTER_ROLE Since the mint function is only callable by the MINTER_ROLE, this exposes a governance risk if any address with the MINTER_ROLE is ever compromised.
Recommendations	Consider implementing a strong internal security system for the RBAC.
Comments / Resolution	Acknowledged. Client agrees with the recommendation and will implement a strong internal security system for the RBAC.