Security Assessment

Epic League

Sep 6th, 2022

Secure3

secure3.io

Table of Contents

Summary

Epic League is a Game platform with its ERC20 game coin EPL. It is a hub of the finest blockchain games with its NFT marketplace, NFT rent portal, treasury, etc to embrace more game players to enjoy blockchain games with its innovative tokenomics.

This report has been prepared for the project to identify issues and vulnerabilities in the smart contract source code. A comprehensive examination with Static Analysis and Manual Review techniques has been performed.

The examination and auditing scope includes:

- Cross checking contract implementation against functionalities described in the documents and white paper disclosed by the project owner.
- Contract Privilege Role Review to provide more clarity on smart contract roles and privilege.
- Using static scanner to analyze smart contracts against common known vulnerabilities patterns.
- Verify the code base is compliant with the most up-to-date industry standards and best practices.
- Comprehensive line-by-line manual code review of the entire codebase by industry experts.

The security assessment resulted in findings that are categorized in four severity levels: Informational, Medium, Critical. For each of the findings we have provided recommendation of a fix or mitigation for security and best practices.

Overview

Project Detail

Project Name	Epic League
Platform & Language	Ethereum, Solidity
Codebase	https://github.com/EpicLeague/contract-audit audit commit - 87e1547ec5309086446ffd14fa72fcccc41df155 final commit - 671669224ec458af6ce1ee54605cdd95f2fb0369
Audit Methodology	• Business Logic Understanding and Review • Static Analysis • Code Review

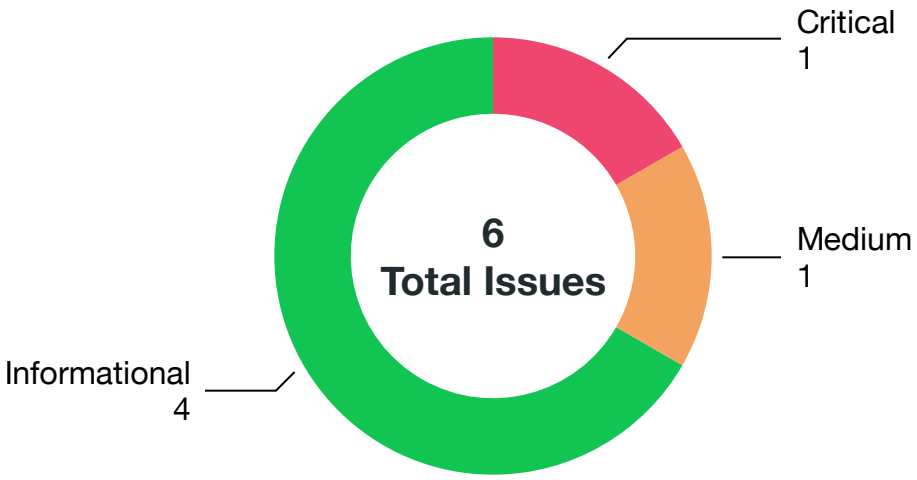
Code Vulnerability Review Summary

Vulnerability Level	Total	Reported	Acknowleged	Fixed	Mitigated
Critical	1	0	0	1	0
Medium	1	0	1	0	0
Informational	4	0	4	0	0

Audit Scope

File	Commit Hash
token/EpicLeague.sol	87e1547ec5309086446ffd14fa72fcccc41df155
token/ozys/token/erc20/BridgeERC20.sol	87e1547ec5309086446ffd14fa72fcccc41df155
token/ozys/token/standard/SafeMath.ERC.sol	87e1547ec5309086446ffd14fa72fcccc41df155

Code Assessment Findings



ID	Name	Category	Severity	Status
EPL-1	BridgeERC20::constructor() lacks validation	Logical	Informational	Acknowledged
EPL-2	BridgeERC20::setTokenInfo() gas optimization	Gas Optimization	Informational	Acknowledged
EPL-3	BridgeERC20.isInitialized logic issue	Logical	Critical	Fixed
EPL-4	BridgeERC20 super privilege of _minter address	Privilege Related	Informational	Acknowledged
EPL-5	Solidity compiler version is not fixed	Language Specific	Informational	Acknowledged
EPL-6	SafeMathERC library is not necessary	Language Specific	Medium	Acknowledged

EPL-1: `BridgeERC20::constructor()` lacks validation

Category	Severity	Code Reference	Status
Logical	Informational	token/ozys/token/erc20/BridgeERC20.sol:39	Acknowledged

Code

```
39:     constructor(address owner_, address minter_, string memory name_, string memory
symbol_, uint8 decimals_, bool init) {
40:         _owner = owner_;
41:         _minter = minter_;
42:
43:         _name = name_;
44:         _symbol = symbol_;
45:         _decimals = decimals_;
46:
47:         isInitialized = init;
48:     }
```

Description

The `BridgeERC20::constructor()` accepts `owner_` and `_minter` address during deployment without any validation before assign them to state variables, while the values can be `address(0)`.

Recommendation

Add validation logic in the constructor to require the two parameters are not `address(0)`.

Client Response

Acknowledged. The Epic League's ERC20 token has already been deployed with the correct constructor parameters.

EPL-2: `BridgeERC20::setTokenInfo()` gas optimization

Category	Severity	Code Reference	Status
Gas Optimization	Informational	token/ozys/token/erc20/BridgeERC20.sol:98	Acknowledged

Code

```
098:     function setTokenInfo(string memory tokenName, string memory tokenSymbol) public
onlyOwnerOrBeforeInit {
099:         _name = tokenName;
100:         _symbol = tokenSymbol;
101:
102:         if(isInitialized == false) isInitialized = true;
103:     }
```

Description

The `BridgeERC20::setTokenInfo()` function is only called externally. Hence it is gas saving to use `calldata` instead of `memory` for `tokenName` and `tokenSymbol`.

Recommendation

Use `calldata` instead of `memory` in the function signature.

Client Response

Acknowledged. The `BridgeERC20::setTokenInfo()` function is only called once by the team internal operators, slightly higher gas consumption is not a concern.

EPL-3: `BridgeERC20.isInitialized` logic issue

Category	Severity	Code Reference	Status
Logical	Critical	token/ozys/token/erc20/BridgeERC20.sol:47	Fixed

Code

```
39:     constructor(address owner_, address minter_, string memory name_, string memory
symbol_, uint8 decimals_, bool init) {
40:         _owner = owner_;
41:         _minter = minter_;
42:
43:         _name = name_;
44:         _symbol = symbol_;
45:         _decimals = decimals_;
46:
47:         isInitialized = init;
48:     }
```

Description

The `BridgeERC20::constructor()` accepts `init` from caller. When `init` is `false`, **any** address can call `setTokenInfo()` to set the token name and token symbol. This is because the `onlyOwnerOrBeforeInit` modifier checks `require(_msgSender() == owner() || !isInitialized)` and `!isInitialized` is `true`.

Recommendation

Consider change `onlyOwnerOrBeforeInit` modifier to `onlyOwner`**`And`**`BeforeInit`, and ensure only the contract owner can set the critical token information.

Client Response

Fixed. The new s contract has been added with init to be true to avoid the problematic state mentioned above.
The `BridgeERC20` is used for the bridge purpose only, and the design is intended for the partner to change their own token information for one-time during the bridge system deployment. Even if some attackers changed the token information maliciously, the owner role can still change it again with `setTokenInfo()` function.

EPL-4: `BridgeERC20` super privilege of `_minter` address

Category	Severity	Code Reference	Status
Privilege Related	Informational	token/ozys/token/erc20/BridgeERC20.sol:261,281	Acknowledged

Code

```
260:     function mint(address account, uint256 amount) public onlyMinter {
261:         _mint(account, amount);
262:     }

281:     function burn(address account, uint256 amount) public onlyMinter {
282:         _burn(account, amount);
283:     }
```

Description

While we understand the `_minter` address needs to perform `mint()` and the `burn()` function to bridge the tokens, the the `_minter` address has the privilege to burn any amount of token from any arbitrary address. In the event of `_minter` address private key compromise, all the token holders' assets are at risk.

Recommendation

Please confirm the `_minter` address is owned or trusted by the Epic League project.

Client Response

Acknowledged. The owner has confirmed the `_minter` address is trusted by the Epic League project.

EPL-5: Solidity compiler version is not fixed

Category	Severity	Code Reference	Status
Language Specific	Informational	All contracts	Acknowledged

Code

```
2: pragma solidity ^0.8.15;
```

Description

The solidity version `^0.8.15` version is floating. Having non fixed compiler version is not the best practice.

Recommendation

Fix the compiler version to `0.8.15` or a version preferred.

Client Response

Acknowledged. The contract has been deployed already and the complier version was `0.8.15`, and the project team has no plan to deploy again, hence it is not a problem.

EPL-6: `SafeMathERC` library is not necessary

Category	Severity	Code Reference	Status
Language Specific	Medium	token/standard/SafeMath.ERC.sol	Acknowledged

Code

```
17: library SafeMathERC {
```

Description

Since version `0.8.0` the solidity compiler handles the arithmetic operations underflow and overflow internally, the `SafeMathERC` functionality is redundant. For more information please refer to https://docs.soliditylang.org/en/v0.8.13/080-breaking-changes.html official document.

Recommendation

Remove the `SafeMathERC` library and use arithmetic operations directly.

Client Response

Acknowledged. We agree on redundancy. This is because the code provided by the bridge partner was written based on compiler version 0.5. Moreover, there will be no major problems with functionality.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Invoices, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Invoice. This report provided in connection with the services set forth in the Invoices shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Invoice. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without Secure3's prior written consent in each instance.

This report is not an "endorsement" or "disapproval" of any particular project or team. This report is not an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Secure3 to perform a security assessment. This report does not provide any warranty or guarantee of free of bug of codes analyzed, nor do they provide any indication of the technologies, business model or legal compliancy.

This report should not be used in any way to make decisions around investment or involvement with any particular project. Instead, it represents an extensive assessing process intending to help our customers increase the quality of their code and high-level consistency of implementation and business model, while reducing the risk presented by cryptographic tokens and blockchain technology.

Secure3's position on the final decisions over blockchain technologies and corresponding associated transactions is that each company and individual are responsible for their own due diligence and continuous security.

The assessment services provided by Secure3 is subject to dependencies and under continuing development. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.