SALUS SECURITY

APR 2023

CODE SECURITY ASSESSMENT

REIGN OF TERROR

Overview

Project Summary

- Name: Reign of Terror
- Version: commit 166bd35
- Platform: BNB Smart Chain
- Language: Solidity
- Repository: https://github.com/reddoordigital/deploy-audit-rot-contracts-evm
- Audit Scope: See Appendix - 1

Project Dashboard

Application Summary

Name	Reign of Terror
Version	v2
Type	Solidity
Dates	Apr 22 2023
Logs	Apr 19 2023; Apr 22 2023

Vulnerability Summary

Total High-Severity issues	0
Total Medium-Severity issues	1
Total Low-Severity issues	0
Total informational issues	2
Total	3

Contact

E-mail: support@salusec.io

Risk Level Description

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for clients' reputations or serious financial implications for clients and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to a moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or defense in depth.

Content

Introduction

1.1 About SALUS

At Salus Security, we are in the business of trust.

We are dedicated to tackling the toughest security challenges facing the industry today. By building foundational trust in technology and infrastructure through security, we help clients to lead their respective industries and unlock their full Web3 potential.

Our team of security experts employ industry-leading proof-of-concept (PoC) methodology for demonstrating smart contract vulnerabilities, coupled with advanced red teaming capabilities and a stereoscopic vulnerability detection service, to deliver comprehensive security assessments that allow clients to stay ahead of the curve.

In addition to smart contract audits and red teaming, our Rapid Detection Service for smart contracts aims to make security accessible to all. This high calibre, yet cost-efficient, security tool has been designed to support a wide range of business needs including investment due diligence, security and code quality assessments, and code optimisation.

We are reachable on Telegram (https://t.me/salusec), Twitter (https://twitter.com/salus_sec), or Email (support@salusec.io).

1.2 Audit Breakdown

The objective was to evaluate the repository for security-related issues, code quality, and adherence to specifications and best practices. Possible issues we looked for included (but are not limited to):

- Risky external calls
- Integer overflow/underflow
- Transaction-ordering dependence
- Timestamp dependence
- Access control
- Call stack limits and mishandled exceptions
- Number rounding errors
- Centralization of power
- Logical oversights and denial of service
- Business logic specification
- Code clones, functionality duplication

1.3 Disclaimer

Note that this security audit is not designed to replace functional tests required before any software release and does not give any warranties on finding all possible security issues with the given smart contract(s) or blockchain software, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues.

Findings

2.1 Summary of Findings

ID	Title	Severity	Category	Status
1	Centralization risk	Medium	Centralization	Acknowledged
2	Use of floating compiler version	Informational	Configuration	Acknowledged
3	Use of deprecated contract	Informational	Configuration	Acknowledged

2.2 Notable Findings

Significant flaws that impact system confidentiality, integrity, or availability are listed below.

1. Centralization risk

Severity: Medium	Category: Centralization
Target: - contracts/reign_token.sol	

Description

The ReignOfTerror contract has several privileged roles:

- The SNAPSHOT_ROLE can create snapshots using the snapshot() function;
- The PAUSER_ROLE can pause or unpause the contract;
- The MINTER_ROLE can mint tokens to an address using mint();
- The UPGRADER_ROLE can upgrade the contract to a new implementation;
- The DEFAULT_ADMIN_ROLE can add or remove an address from any of the roles.

contracts/reign_token.sol:L28-L45

```
function initialize() initializer public {
    __ERC20_init("Reign of Terror", "REIGN");
    __ERC20Burnable_init();
    __ERC20Snapshot_init();
    __AccessControl_init();
    __Pausable_init();
    __ERC20Permit_init("Reign of Terror");
    __ERC20Capped_init(2000000000 * 10 ** decimals());  // Set maximum supply cap to 2
billion tokens
    __ERC20Votes_init();
    __UUPSUpgradeable_init();

    _grantRole(DEFAULT_ADMIN_ROLE, msg.sender);
    _grantRole(SNAPSHOT_ROLE, msg.sender);
    _grantRole(PAUSER_ROLE, msg.sender);
    _mint(msg.sender, 300000000 * 10 ** decimals());
    _grantRole(MINTER_ROLE, msg.sender);
    _grantRole(UPGRADER_ROLE, msg.sender);
}
```

The ReignOfTerror contract uses the UUPS proxy pattern. After the contract is deployed and initialized, the deployer is granted all the roles mentioned earlier. If attackers were to gain access to the deployer's private key, they could cause significant damage to the project.

Recommendation

We recommend transferring the privileged roles to multi-sig accounts.

Status

This issue has been acknowledged by the team.

2.3 Informational Findings

2. Use of floating compiler version

Severity: Informational	Category: Configuration
Target: - contracts/reign_token.sol	

Description

contracts/reign_token.sol:L2

```
pragma solidity ^0.8.0;
```

The ReignOfTerror contract uses a floating compiler version ^0.8.0.

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Recommendation

Consider locking the pragma version.

Status

This issue has been acknowledged by the team.

3. Use of deprecated contract

Severity: Informational	Category: Configuration
Target: - contracts/reign_token.sol	

Description

contracts/reign_token.sol:L10

```
import
"@openzeppelin/contracts-upgradeable/token/ERC20/extensions/draft-ERC20PermitUpgradeable
.sol";
```

As indicated in OpenZeppelin's changelog, the draft-ERC20PermitUpgradeable contract is deprecated. It is recommended to use the ERC20PermitUpgradeable contract instead.

Recommendation

Consider importing and using the ERC20PermitUpgradeable contract, instead of the draft-ERC20PermitUpgradeable contract.

Status

This issue has been acknowledged by the team.

Appendix

Appendix 1 - Files in Scope

This audit covered the following file in commit 166bd35:

File	SHA-1 hash
contracts/reign_token.sol	c0c3db45c818af644d735ebfdf6c953c2c330929