verichains

SECURITY AUDIT OF

SNAKECITY SMART CONTRACT

Public Report

Apr 07, 2022

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
Ether (ETH)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
ERC20	ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain.

EXECUTIVE SUMMARY

This Security Audit Report prepared by Verichains Lab on Apr 07, 2022. We would like to thank the SnakeCity for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the SnakeCity Smart Contract. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified no vulnerable issues in the smart contracts code.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About SnakeCity Smart Contract

SnakeCity is the next-generation decentralized blockchain game which is inspired by famous game Slither.io. With concept: Simplicity + Fun + Addiction is the core value, SnakeCity bring the unique experience to user who can "earn from playing" and really enjoy and have fun together.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the smart contracts of AuditProduct.

It was conducted on commit 69edb7ccf70a813073da948d80ea6c9c52dd5f8e from git repository *https://github.com/longhg/snakecity-contracts*.

The following file was made available in the course of the review:

SHA256 Sum	File
5154c0eac91481753f333a1e3096f728a073755471f92623b9062dfee048195f	SnctToken.sol

1.3. Audit methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function

- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 1. Severity levels

1.4. Disclaimer

Please note that security auditing cannot uncover all existing vulnerabilities, and even an audit in which no vulnerabilities are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

2. AUDIT RESULT

2.1. Overview

2.2. Contract codes

The SnakeCity Smart Contract was written in Solidity language, with the required version to be 0.8.2.

The contract imported some contracts that were implemented by OpenZeppelin.

SnakeCity Smart Contract extends Ownable, ERC20Burnable and ERC20Votes contracts. by default, Token Owner is contract deployer, but he can transfer ownership to another address at any time. ERC20Snapshot is an extension of ERC20 to support Compound's voting and delegation. ERC20Burnable allows token holders to destroy both their own tokens and those that they have an allowance for. The contract also implements a bunch of AntiBot functions which reduce the bot control the token price in the IDO time.

2.3. Findings

During the audit process, the audit team found no vulnerability in the given version of SnakeCity Smart Contract.

2.4. Additional notes and recommendations

2.4.1. BPContract function INFORMATIVE

Since we do not control the logic of the bpContract, there is no guarantee that bpContract will not contain any security related issues. With the current context, in case the bpContract is compromised, there is not yet a way to exploit the SnakeCity Smart Contract, but we still note that here as a warning for avoiding any related issue in the future.

By the way, if having any issue, the bpContract function can be easily disabled anytime by the contract owner using the setBPEnabled function. In addition, bpContract is only used in a short time in token public sale IDO then the contract owner will disable it forever by the setBPDisableForever function.

APPENDIX

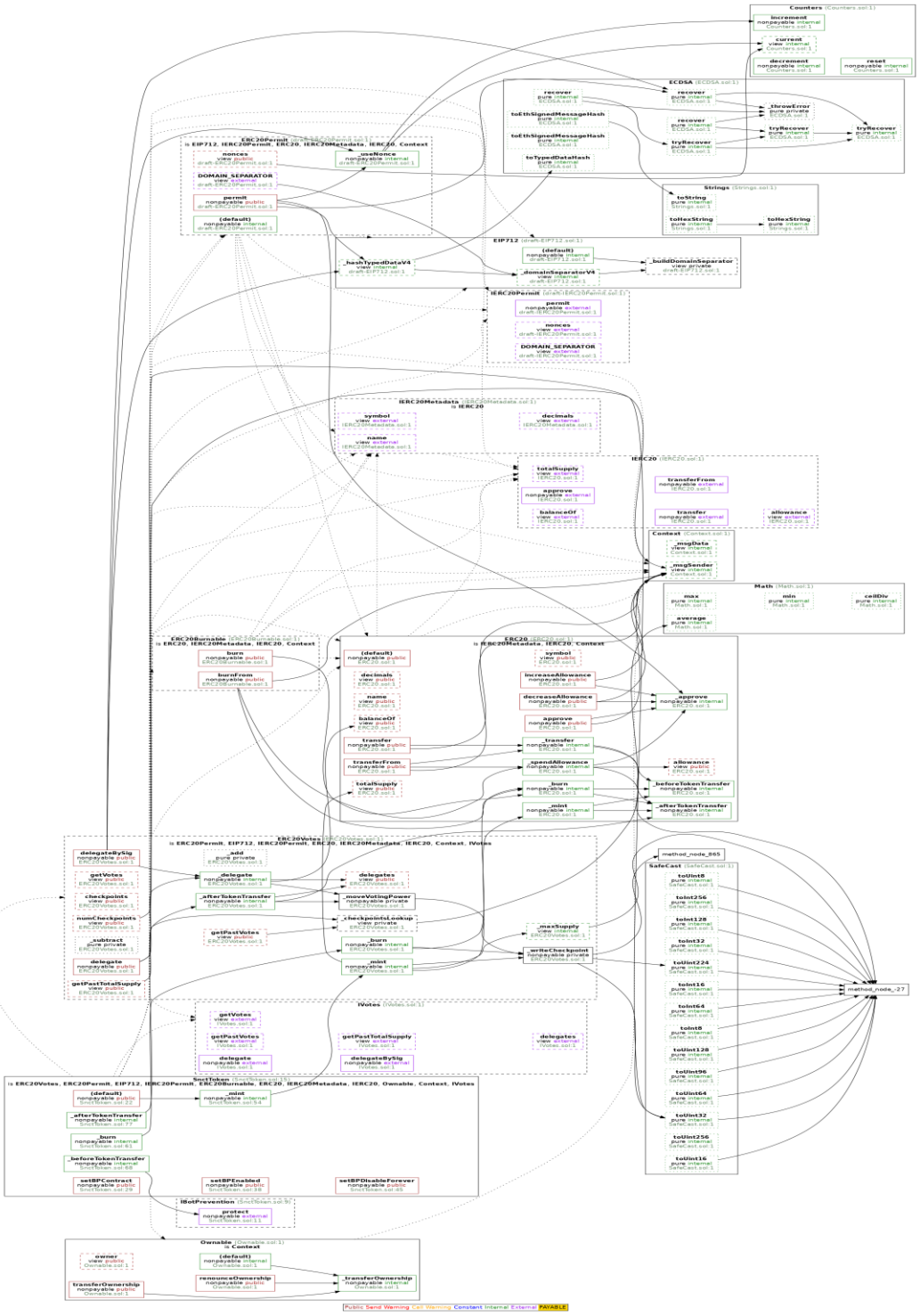

Image 1. SnakeCity Smart Contract call graph

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Apr 07, 2023*	Public Report	Verichains Lab

Table 2. Report versions history