SECURITY AUDIT OF

ARK RIVALS TOKEN SMART CONTRACT

Public Report

Feb 09, 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 Feb 09, 2022. We would like to thank the Ark Rivals 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 Ark Rivals Token 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 Ark Rivals Token Smart Contract

Ark Rivals is a Sci-fi action strategy NFT game entirely based on User-Generated Content (UGC). As a result, everything in Ark Rivals is significant, and it is your duty to develop and protect it.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the Ark Rivals Token Smart Contract.

It was conducted on commit c2684c0f2d4ad6b364c37e018742e3744c4e574d from git repository *https://github.com/arkrivals/arkn-token*.

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

This table lists some properties of the audited Ark Rivals Token Smart Contract (as of the report writing time).

PROPERTY	VALUE
Name	ArkRivals
Symbol	ARKN
Decimals	18
Total Supply	1,000,000,000 (x10^{18}) Note: the number of decimals is 18, so the total representation token will be 1,000,000,000 or 1 billion.

Table 2. The Ark Rivals Token Smart Contract properties

The Ark Rivals Token Smart Contract was written in Solidity language, with the required version to be ^0.8.3. Almost all source codes in the Ark Rivals Token Smart Contract are imported from OpenZeppelin contracts.

Ark Rivals Token Smart Contract extends ERC20, BEP20Burnable, Pausable, ERC20Snapshot and Ownable contracts. by default, Token Owner is contract deployer, but he can transfer ownership to another address at any time. He can pause/unpause contract using Pausable contract, users can only transfer tokens when contract is not paused. ERC20Snapshot help Token Owner takes a snapshot of the balances and total supply at a time for later access. 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.

2.2. Findings

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

2.3. Additional notes and recommendations

2.3.1. Unnecessary usage of SafeMath library in Solidity 0.8.0+ INFORMATIVE

All safe math usages in the contract are for overflow checking, solidity 0.8.0+ already do that by default, the only usage of safemath now is to have a custom revert message which isn't the case in the auditing contracts. We suggest using normal operators for readability and gas saving.

RECOMMENDATION

We suggest changing all methods from SafeMath library to normal arithmetic operator.

APPENDIX

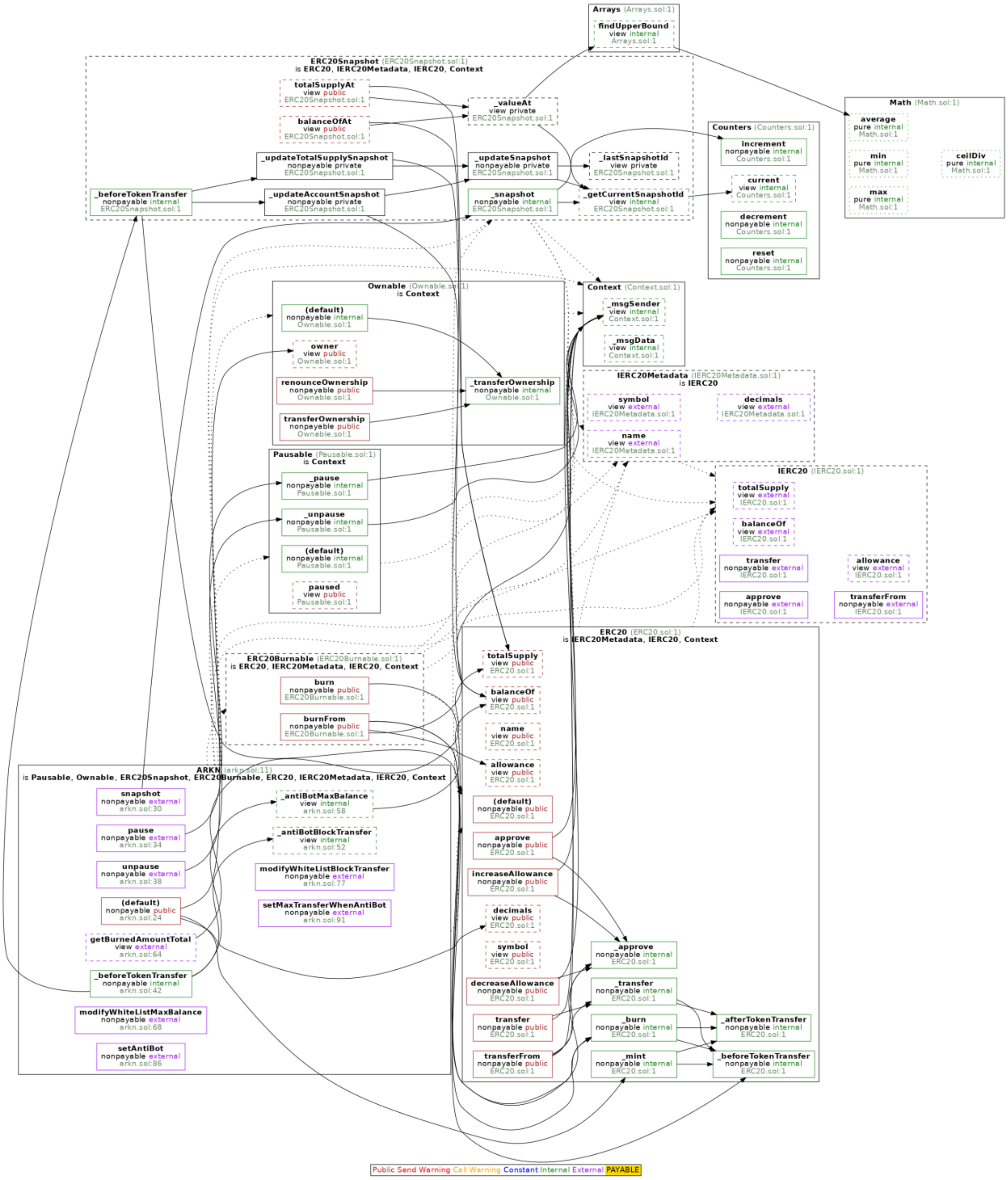
Arrays (Arrays.sol:1)
findUpperBound view internal Arrays.sol:1
ERC20Snapshot (ERC20Snapshot.sol:1) is ERC20, IERC20Metadata, IERC20, Context
totalSupplyAt view public ERC20Snapshot.sol:1
balanceOfAt view public ERC20Snapshot.sol:1
_valueAt view private ERC20Snapshot.sol:1
_updateTotalSupplySnapshot nonpayable private ERC20Snapshot.sol:1
_updateSnapshot nonpayable private ERC20Snapshot.sol:1
_lastSnapshotId view private ERC20Snapshot.sol:1
_beforeTokenTransfer nonpayable internal ERC20Snapshot.sol:1
_updateAccountSnapshot nonpayable private ERC20Snapshot.sol:1
_snapshot nonpayable internal ERC20Snapshot.sol:1
_getCurrentSnapshotId view internal ERC20Snapshot.sol:1
Counters (Counters.sol:1)
increment nonpayable internal Counters.sol:1
current view internal Counters.sol:1
decrement nonpayable internal Counters.sol:1
reset nonpayable internal Counters.sol:1
Math (Math.sol:1)
average pure internal Math.sol:1
min pure internal Math.sol:1
ceilDiv pure internal Math.sol:1
max pure internal Math.sol:1
Ownable (Ownable.sol:1) is Context
(default) nonpayable internal Ownable.sol:1
owner view public Ownable.sol:1
renounceOwnership nonpayable public Ownable.sol:1
transferOwnership nonpayable public Ownable.sol:1
_transferOwnership nonpayable internal Ownable.sol:1
Context (Context.sol:1)
_msgSender view internal Context.sol:1
_msgData view internal Context.sol:1
IERC20Metadata (IERC20Metadata.sol:1) is IERC20
symbol view external IERC20Metadata.sol:1
decimals view external IERC20Metadata.sol:1
name view external IERC20Metadata.sol:1
Pausable (Pausable.sol:1) is Context
_pause nonpayable internal Pausable.sol:1
_unpause nonpayable internal Pausable.sol:1
(default) nonpayable internal Pausable.sol:1
paused view public Pausable.sol:1
IERC20 (IERC20.sol:1)
totalSupply view external IERC20.sol:1
balanceOf view external IERC20.sol:1
transfer nonpayable external IERC20.sol:1
allowance view external IERC20.sol:1
approve nonpayable external IERC20.sol:1
transferFrom nonpayable external IERC20.sol:1
ERC20 (ERC20.sol:1) is IERC20Metadata, IERC20, Context
ERC20Burnable (ERC20Burnable.sol:1) is ERC20, IERC20Metadata, IERC20, Context
burn nonpayable public ERC20Burnable.sol:1
burnFrom nonpayable public ERC20Burnable.sol:1
totalSupply view public ERC20.sol:1
balanceOf view public ERC20.sol:1
name view public ERC20.sol:1
allowance view public ERC20.sol:1
(default) nonpayable public ERC20.sol:1
approve nonpayable public ERC20.sol:1
increaseAllowance nonpayable public ERC20.sol:1
decimals view public ERC20.sol:1
symbol view public ERC20.sol:1
decreaseAllowance nonpayable public ERC20.sol:1
transfer nonpayable public ERC20.sol:1
transferFrom nonpayable public ERC20.sol:1
_approve nonpayable internal ERC20.sol:1
_transfer nonpayable internal ERC20.sol:1
_burn nonpayable internal ERC20.sol:1
_mint nonpayable internal ERC20.sol:1
_afterTokenTransfer nonpayable internal ERC20.sol:1
_beforeTokenTransfer nonpayable internal ERC20.sol:1
ARKN (arkn.sol:11) is Pausable, Ownable, ERC20Snapshot, ERC20Burnable, ERC20, IERC20Metadata, IERC20, Context
snapshot nonpayable external arkn.sol:30
_antiBotMaxBalance view internal arkn.sol:58
pause nonpayable external arkn.sol:34
_antiBotBlockTransfer view internal arkn.sol:52
unpause nonpayable external arkn.sol:38
modifyWhiteListBlockTransfer nonpayable external arkn.sol:77
(default) nonpayable public arkn.sol:24
setMaxTransferWhenAntiBot nonpayable external arkn.sol:91
getBurnedAmountTotal view external arkn.sol:64
_beforeTokenTransfer nonpayable internal arkn.sol:42
modifyWhiteListMaxBalance nonpayable external arkn.sol:68
setAntiBot nonpayable external arkn.sol:86
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 1. Ark Rivals Token Smart Contract call graph

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Feb 09, 2022*	Public Report	Verichains Lab

Table 3. Report versions history