



verichains

*SECURITY AUDIT OF*  
**KINGDOM RAIDS IDO SMART**  
**CONTRACT**



**Public Report**

*Nov 29, 2021*

**Verichains Lab**

[info@verichains.io](mailto:info@verichains.io)

<https://www.verichains.io>

*Driving Technology > Forward*

## ABBREVIATIONS

Name	Description
<b>Ethereum</b>	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
<b>Ether (ETH)</b>	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
<b>Smart contract</b>	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
<b>Solidity</b>	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
<b>Solc</b>	A compiler for Solidity.
<b>ERC20</b>	ERC20 (BEP20 in Binance Smart Chain or xRP20 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 Nov 29, 2021. We would like to thank the Kingdom Raids 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 Kingdom Raids IDO 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

<b>1. MANAGEMENT SUMMARY.....</b>	<b>5</b>
<b>1.1. About Kingdom Raids IDO Smart Contract.....</b>	<b>5</b>
<b>1.2. Audit scope .....</b>	<b>5</b>
<b>1.3. Audit methodology.....</b>	<b>5</b>
<b>1.4. Disclaimer .....</b>	<b>6</b>
<b>2. AUDIT RESULT .....</b>	<b>7</b>
<b>2.1. Overview .....</b>	<b>7</b>
<b>2.2. Contract codes.....</b>	<b>7</b>
<b>2.3. Findings .....</b>	<b>7</b>
<b>2.4. Additional notes and recommendations.....</b>	<b>7</b>
2.4.1. Using hardcode in getCurrentPrice function INFORMATIVE.....	7
2.4.2. Outdated version of Solidity INFORMATIVE.....	8
2.4.3. An incorrect comment causes misunderstand INFORMATIVE.....	8
<b>3. VERSION HISTORY .....</b>	<b>11</b>

# 1. MANAGEMENT SUMMARY

## 1.1. About Kingdom Raids IDO Smart Contract

Kingdom Raids is an RPG developed by a leading gaming studio, Alley Labs which has attracted more than 100 million downloads for their games. The game takes place in a fictional kingdom, “Dood Kingdom”, a land of mystery and adventure.

The Kingdom Raids IDO Smart Contract is a contract used to support to release of tokens in the Initial DEX Offering.

## 1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the smart contracts of Kingdom Raids Token. It was conducted on commit [ae8e713dfa8a919b4d4eb1491837dff187323adf](https://github.com/kingdomraids/kr-ido-contract/commit/ae8e713dfa8a919b4d4eb1491837dff187323adf) from git repository <https://github.com/kingdomraids/kr-ido-contract/>.

## 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
<b>CRITICAL</b>	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
<b>HIGH</b>	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
<b>MEDIUM</b>	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
<b>LOW</b>	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

The initial review was conducted in Nov 2021 and a total effort of 5 working days was dedicated to identifying and documenting security issues in the code base of Kingdom Raids IDO contract.

### 2.2. Contract codes

The Kingdom Raids IDO Smart Contract was written in **Solidity** language, with the required version to be **0.6.12**.

The contract is used to support to release tokens in the Initial DEX Offering.

There are three phases during the contract works. The contract will handle 3 logical ido phases according to time as below:

- **SalePhase**: initial phase, users can **deposit** the input token to join the IDO. If users want to stop joining the IDO, they can **withdraw** the input token.
- **GracePhase**: this will follow after SalePhase, it's the second chance for users to **withdraw** the input token if they want to leave the IDO. After this phase, only the **owner** of the contract can **withdraw** the input token through the **finalize** function.
- **RedeemPhase**: users can **redeem** the tokens corresponding to the input token that they **deposited**. If users don't **redeem** the tokens in this phase, the tokens will belong to the **owner** of the contract.

### 2.3. Findings

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

### 2.4. Additional notes and recommendations

#### 2.4.1. Using **hardcode** in **getCurrentPrice** function **INFORMATIVE**

In the **getCurrentPrice** functions, the function calculates by **mul** with the power of **decimals** input token but it is using a **hardcode** value. It may be an issue if the **decimals** value is changed.

```
101 function getCurrentPrice() public view returns (uint256) {  
102     return totalDeposit.mul(1e18).div(totalSupply);  
103 }
```

#### RECOMMENDATION

We suggest importing the [IERC20Metadata](#) interface and using the [decimals](#) value instead of hardcode in the [getCurrentPrice](#) function like the below code:

```
import "@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol...";

function getCurrentPrice() public view returns (uint256) {
    uint8 decimals=token.decimals()
    return totalDeposit.mul(10**decimals).div(totalSupply);
}
```

#### UPDATES

- 2021-11-29: This issue has been acknowledged and fixed by the Kingdom Raids team in commit [ec6b13a6e72f846afd56baf42c9d701617d77bdd](#).

#### 2.4.2. Outdated version of Solidity **INFORMATIVE**

The required version of Solidity in the Kingdom Raids IDO Smart Contract source code is [0.6.12](#), which is outdated.

Updating the Solidity version from [0.6.12](#) to [0.8.10](#) will introduce many features and fix some bugs like:

- Automatic integer overflow/underflow checking for arithmetic operations.
- Allow overrides to have stricter state mutability: view can override [nonpayable](#) and [pure](#) can override [view](#).
- Disallow public state variables overwriting [pure](#) functions.

#### UPDATES

- 2021-11-29: This issue has been acknowledged and fixed by the Kingdom Raids team in commit [1e3352de9485c648bd07e870377f8b9af4c49e72](#).

#### 2.4.3. An incorrect comment causes misunderstand **INFORMATIVE**

The comment says that the below variable is the end sale time but the below variable is the end grace time.

```
24 // Start grace time
25 uint256 public startGraceAt;
26 // End sale time
27 uint256 public endGraceAt;
28 // Start redeem time
29 uint256 public startRedeemAt;
```



## Report for Kingdom Raids

### Security Audit – Kingdom Raids IDO Smart Contract

Version: 1.1 - Public Report

Date: Nov 29, 2021



#### RECOMMENDATION

We suggest changing the comment to `// End grace time` for readability.

#### UPDATES

- 2021-11-29: This issue has been acknowledged and fixed by the Kingdom Raids team in commit [1e3352de9485c648bd07e870377f8b9af4c49e72](#).

## APPENDIX

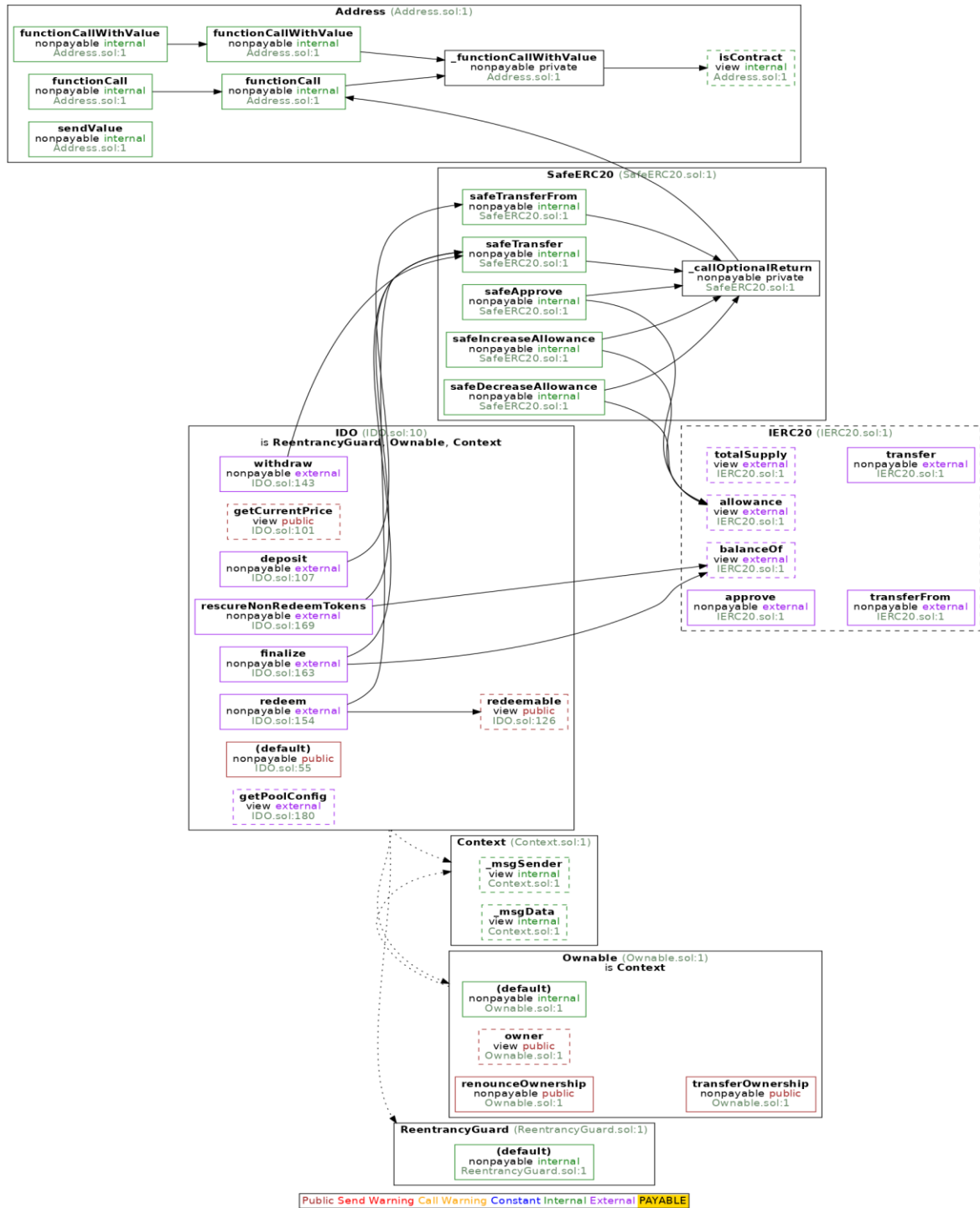


Image 1. Kingdom Raids IDO Smart Contract call graph

## Report for Kingdom Raids

### Security Audit – Kingdom Raids IDO Smart Contract

Version: 1.1 – Public Report

Date: Nov 29, 2021



## 3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	2021-11-23	Public Report	Verichains Lab
1.1	2021-11-29	Public Report	Verichains Lab

*Table 2. Report versions history*