SECURITY AUDIT OF

CRYPTOWARLORDS TOKEN

Public Report

Mar 04, 2024

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 was prepared by Verichains Lab on Mar 04, 2024. We would like to thank the CryptoWarlords 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 CryptoWarlords Token. 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 some vulnerable issues in the smart contracts code.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About CryptoWarlords Token

CryptoWarlords is an epic Play-to-Earn Game based on Blockchain Technology in Binance Smart Chain Network. CryptoWarlords welcomes you to an engaging Battle where Warriors come together for exciting adventures and earn rewards in $LORDS, our native Cryptocurrency on BSC.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the CryptoWarlords Token that was deployed on BSC.

The latest version was made available in the course of the review:

FIELD	VALUE
Address Deploy	0x7db84Fd9130169505170535fe29Db1Aa76F55659
Tx Deploy	0x8405dd5b0b7d54d73584f522f5600b258b2fc573dbe076cc9e7e68fb6e742cf9
Deployer	0x784b0A4E9CCe1c449fDc8662D2BF5a3782AbF860
Block Number (at audit time)	36664994

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 were 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

CryptoWarlords acknowledges that the security services provided by Verichains, are conducted to the best of their professional abilities but cannot guarantee 100% coverage of all security vulnerabilities. CryptoWarlords understands and accepts that despite rigorous auditing, certain vulnerabilities may remain undetected. Therefore, CryptoWarlords agrees that Verichains shall not be held responsible or liable, and shall not be charged for any hacking incidents that occur due to security vulnerabilities not identified during the audit process.

1.5. Acceptance Minute

This final report served by Verichains to the CryptoWarlords will be considered an Acceptance Minute. Within 7 days, if no any further responses or reports is received from the CryptoWarlords, the final report will be considered fully accepted by the CryptoWarlords without the signature.

2. AUDIT RESULT

2.1. Overview

The CryptoWarlords Token was written in `Solidity` language, with the required version to be `0.8.24`.

The contract extends `ERC20` and `Ownable`. Below table describes some properties of the audited CryptoWarlords Token (at the report writing time).

PROPERTY	VALUE
Name	CryptoWarlords
Symbol	LORDS
Decimals	18
Total Supply	100,000,000x10^{18} Note: the number of decimals is 18, so the total representation token will be 100 million.
Sell/Buy Tax	No

Table 2. The CryptoWarlords Token properties

Gas Price Control:

- The contract introduces a gas price control mechanism to mitigate potential front-running attacks.
- The `maxGas` state (modifiable) represents the maximum allowed gas price for transactions and is set to 25 gwei by default.

Trading Time Control:

- The contract introduces a trading time control mechanism to mitigate potential sandwich attacks.
- The `lastTrade` mapping store the last trade time of the user and a user can trade in a same block.

2.2. Findings

During the audit process, the audit team found some vulnerabilities in the given version of CryptoWarlords Token.

2.2.1. LOW - Sandwich Attack Mechanism Can be Bypassed.

Description

If the user is not whitelisted, the contract will check their previous trade. Unfortunately, `tx.origin` is used to verify. The attacker can easily bypass this mechanism by executing transactions using multiple senders.

```
function _transfer(
    address from,
    address to,
    uint256 amount
) internal override {
    if (!isWhitelisted(from)) {
        //...
        if (antiSandwich) {
            uint lastNumber = lastTrade[tx.origin];
            require(lastNumber != block.number, "You cant trade yet.");
        }
    }
    //...
    if (antiSandwich) {
        if (!isWhitelisted(from)) {
            lastTrade[from] = block.number;
        }
        if (!isWhitelisted(to)) {
            lastTrade[to] = block.number;
        }
    }
}
```

RECOMMENDATION

The audit team recommends using `from` instead of `tx.origin` to check the last trade of the user. In addition, the `router` and `pair` addresses should be excluded from the last trade check.

Note: In the blockchain, It is not possible to prevent sandwich attacks completely. Therefore, this issue is not a critical issue and above recommendation is optional.

UPDATES

- ***Mar 04, 2024***: The issue is mitigated by CryptoWarlords Token team.

2.2.2. INFORMATIVE - The User May be Unable to Execute Calls Due to `maxgas` is Insufficient.

Description

Users may encounter issues executing calls in smart contract due to a `maxgas` set too low. This can lead to transaction failures and unsuccessful attempts to execute calls.

```
uint public maxGas = 25 gwei;

function _transfer(
    address from,
    address to,
    uint256 amount
) internal override {
    if (!isWhitelisted(from)) {
        if (antiFR) {
            require(tx.gasprice <= maxGas, "Cant pay that much gas price." );
        }
        //...
    }
    //...
}
```

RECOMMENDATION

Setting the `maxGas` value is best practice when simply exchanging tokens. When the token is used for purposes other than trading, the audit team suggests increasing the `maxGas` value or disabling the gas limit.up

UPDATES

- ***Mar 04, 2024***: The issue is acknowledged by CryptoWarlords Token team.

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Feb 29, 2024*	Public Report	Verichains Lab
1.1	*Mar 04, 2024*	Public Report	Verichains Lab

Table 3. Report versions history