# LORC

## Smart Contract Audit (Final Report)

17 August 2021

**For:** LandOrc Limited
**Prepared By:** Entersoft Pte Ltd of 1B Trengganu Street, Singapore 058455

## CONTENTS

## Revision History and Version Control

| Release Number | Date | Author | Comments |
|---|---|---|---|
| 1.0 | 18th August 2021 | Abhishek | Final Report |
| **Reviewed By** | Jake Lemke | | |
| **Released By** | Paul Kang | | |

Entersoft was commissioned by LORC to perform source code review on their solidity smart contract. The review was conducted between 10 August 2021 to 15 August 2021. The report is organized into the following sections.

- Executive Summary: A high-level overview of the Smart Contract Security Audit findings.
- Technical analysis: Our detailed analysis of the Smart Contract Code

The information in this report should be used to understand overall code quality, security, correctness, and meaning that code will work as LORC described in the smart contract.

## 1.0 Disclaimer

This is a limited audit report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to: (i) smart contract best coding practices and issues in the framework and algorithms based on white paper, code, the details of which are set out in this report, (Smart Contract audit). To get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us based on what it says or does not say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full. DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Entersoft Australia and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Entersoft) owe no duty of care towards you or any other person, nor does Entersoft make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Entersoft hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Entersoft hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Entersoft, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the Smart contract is purely based on the smart contract code shared with us alone.

# 2.0 Overview

## 2.1 Scope of Audit

The scope of this audit was to analyse LORC smart contract codebase for quality, security, and correctness. Following is the list of smart contracts included in the scope of this audit:

- ✓ LandOrc.sol
- ✓ LORCPresale.sol

**OUT-OF-SCOPE:** External contracts, External Oracles, other smart contracts in the repository or imported smart contracts.

## 2.2 Project Summary

| | |
|---|---|
| **Project Name** | LORC |
| **Platform** | ETHEREUM |
| **Codebase** | https://www.dropbox.com/sh/72zqgait2zyo0fr/AABUvOYxcHXwqqwko_pOQqDBa?dl=0 |
| **Token Name** | LORC |
| **Contract Name** | LORCPresale.sol, LandOrc.sol |
| **Contract Address** | Not Deployed |
| **Verified** | Yes |
| **Audited** | Yes |
| **Errors or Vulnerabilities** | All identified vulnerabilities have been remediated. See Section 3.2 Findings. |

## 2.3 Audit Summary

| | |
|---|---|
| **Delivery Date** | 17 August 2021 |
| **Method of Audit** | Static Analysis, Manual Review, Automated Review, Unit Testing |
| **Consultants Engaged** | 2 |

## 2.4 Security Level references

Every vulnerability in this report was assigned a severity level from the following:

High severity vulnerabilities:
The vulnerability puts the vast majority of, or large numbers of, users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.

Medium severity vulnerabilities:
The vulnerability puts a subset of individual users' sensitive information is at risk, exploitation would be detrimental for the client's reputation, or is reasonably likely to lead to moderate financial impact.

Low severity vulnerabilities:
The risk is relatively small and could not be exploited on a recurring basis or is a risk the client has indicated is not important or impactful in view of the client's business circumstances.

Informational:
The issue does not pose an immediate threat to continued operation or usage, but is relevant for security best practices, software engineering best practices, or defensive redundancy.

## 2.5 Vulnerability Summary

The vulnerability summary table showcases the number of vulnerabilities in the smart contracts upon the retest of the smart contract after initial vulnerability findings have been reviewed and resolved by the LORC team.

| **Total Vulnerabilities** | **0** |
|---|---|
| **Critical** | 0 |
| **High** | 0 |
| **Medium** | 0 |
| **Low** | 0 |
| **Informational** | 0 |

## 2.6 Audit Results Overview

| Audit Item | Audit Subclass | Audit Result |
|---|---|---|
| Overflow | - | Passed |
| Race Conditions | - | Passed |
| Permissions | Permission Vulnerability Audit<br>Excessive Auditing Authority | Passed |
| Safety Design | Zeppelin Safe Math | Passed |
| DDOS Attack | Call Function Security | Passed |
| Gas Optimization | - | Passed |
| Design Logic | - | Passed |
| Know Attacks | - | Passed |
| Overall Audit Result | - | Passed |

# 3.0 Executive Summary

## 3.1 Files in Scope

- LandOrc.sol
- LORCPresale.sol

## 3.2 Findings

| Vulnerability ID | Title | Severity | Resolved |
|---|---|---|---|
| **LORC-SCA-001** | Unused Return value | Medium | ✓ |
| **LORC-SCA-002** | Function that could be declared externally | Low | ✓ |
| **LORC-SCA-003** | Pragma solidity version is not locked | Low | ✓ |

## 3.3 Comments

The use case of the smart contract is very well designed and Implemented. Overall, the code is well written and demonstrates effective use of abstraction, separation of concerns, and modularity.

The LORC development team demonstrated high technical capabilities, both in the design of the architecture and in the implementation. The identified Medium and Low Level vulnerabilities have been fixed and re-tested.

## 4.0 Vulnerabilities

### 4.1 Unused Return Value

| Type | Severity | Location | Confidence | Status |
|---|---|---|---|---|
| Unused return | Medium | LORCPresale.sol | High | Fixed |

**Description:**
LORCPreSale.buyWithETH (LorcPreSale.sol#) does not use the value returned by external calls:
-lorcToken.transfer(msg.sender,lorcValue) (LorcPreSale.sol#)

LorcPreSale.buyWithUSDT (LorcPreSale.sol#) does not use the value returned by external calls:
-lorcToken.transfer(msg.sender,lorcValue) (LorcPreSale.sol#)

Line Number: 47 (LorcPreSale.sol)

**Remediation**
Use Require statement

```
// Transfer tokens to the sender
require(lorcToken.transfer(msg.sender, lorcValue));
```

### 4.2 Function that could be declared external

| Type | Severity | Location | Confidence | Status |
|---|---|---|---|---|
| External Function | Low (optimisation) | LorcPresales.sol | High | Fixed |

**The following public functions that are never called by the contract should be declared external to save gas:**

- Ownable.owner (LorcPreSale.sol#142-144) should be declared external
- Ownable.renounceOwnership (LorcPreSale.sol#161-164) should be declared external
- Ownable.transferOwnership (LorcPreSale.sol#170-174) should be declared external
- LorcPreSale.buyWithETH (LorcPreSale.sol#508-521) should be declared external
- LorcPreSale.buyWithUSDT (LorcPreSale.sol#524-539) should be declared external
- LorcPreSale.withdrawETH (LorcPreSale.sol#542-546) should be declared external
- LorcPreSale.withdrawErc20 (LorcPreSale.sol#549-554) should be declared external

**Remediation**
Use the external attribute for functions never called from the contract.

## 4.3 Pragma solidity version is not locked

| Type | Severity | Location | Confidence | Status |
|---|---|---|---|---|
| Version | Low | N/A | High | Fixed |

**Description**
pragma solidity ^0.8.2 -Solidity version should be locked

**Remediation**
Use pragma solidity 0.8.2.

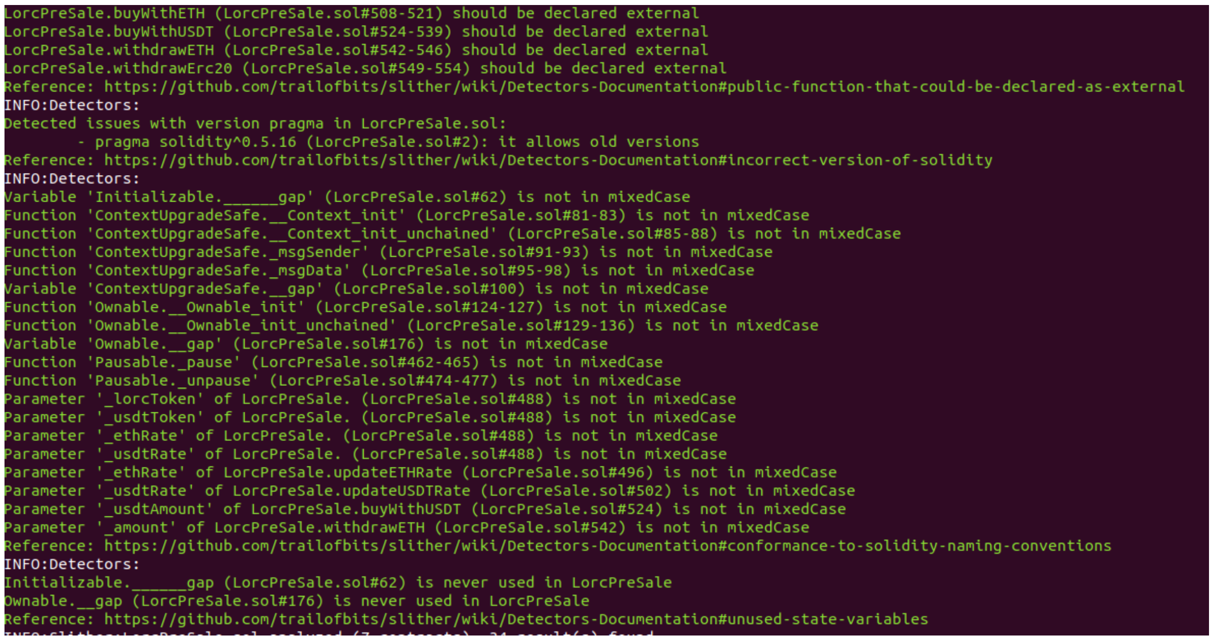
## 5.0 Automated Testing

Automated testing was carried out with the following tools:

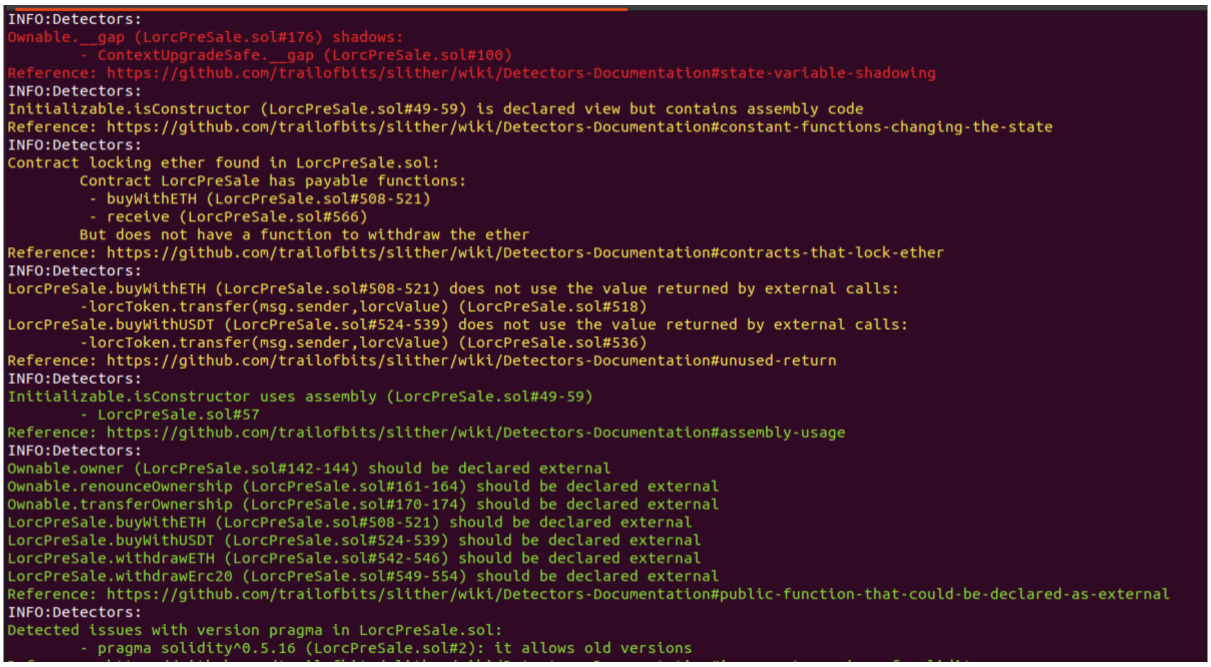
- Slither
- Mythril
- Echidna
- Manticore

### 5.1 Slither

Slither is an open-source Solidity static analysis framework. This tool provides rich information about Ethereum smart contracts and has the critical properties. It runs a suite of vulnerability detectors, prints visual information about contract details, and provides an API to easily write custom analyses.



```
LorcPreSale.buyWithETH (LorcPreSale.sol#508-521) should be declared external
LorcPreSale.buyWithUSDT (LorcPreSale.sol#524-539) should be declared external
LorcPreSale.withdrawETH (LorcPreSale.sol#542-546) should be declared external
LorcPreSale.withdrawErc20 (LorcPreSale.sol#549-554) should be declared external
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#public-function-that-could-be-declared-as-external
INFO:Detectors:
Detected issues with version pragma in LorcPreSale.sol:
        - pragma solidity^0.5.16 (LorcPreSale.sol#2): it allows old versions
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#incorrect-version-of-solidity
INFO:Detectors:
Variable 'Initializable.______gap' (LorcPreSale.sol#62) is not in mixedCase
Function 'ContextUpgradeSafe.__Context_init' (LorcPreSale.sol#81-83) is not in mixedCase
Function 'ContextUpgradeSafe.__Context_init_unchained' (LorcPreSale.sol#85-88) is not in mixedCase
Function 'ContextUpgradeSafe._msgSender' (LorcPreSale.sol#91-93) is not in mixedCase
Function 'ContextUpgradeSafe._msgData' (LorcPreSale.sol#95-98) is not in mixedCase
Variable 'ContextUpgradeSafe.__gap' (LorcPreSale.sol#100) is not in mixedCase
Function 'Ownable.__Ownable_init' (LorcPreSale.sol#124-127) is not in mixedCase
Function 'Ownable.__Ownable_init_unchained' (LorcPreSale.sol#129-136) is not in mixedCase
Variable 'Ownable.__gap' (LorcPreSale.sol#176) is not in mixedCase
Function 'Pausable._pause' (LorcPreSale.sol#462-465) is not in mixedCase
Function 'Pausable._unpause' (LorcPreSale.sol#474-477) is not in mixedCase
Parameter '_lorcToken' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_usdtToken' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_ethRate' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_usdtRate' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_ethRate' of LorcPreSale.updateETHRate (LorcPreSale.sol#496) is not in mixedCase
Parameter '_usdtRate' of LorcPreSale.updateUSDTRate (LorcPreSale.sol#502) is not in mixedCase
Parameter '_usdtAmount' of LorcPreSale.buyWithUSDT (LorcPreSale.sol#524) is not in mixedCase
Parameter '_amount' of LorcPreSale.withdrawETH (LorcPreSale.sol#542) is not in mixedCase
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Initializable.______gap (LorcPreSale.sol#62) is never used in LorcPreSale
Ownable.__gap (LorcPreSale.sol#176) is never used in LorcPreSale
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#unused-state-variables
```



```
INFO:Detectors:
Ownable.__gap (LorcPreSale.sol#176) shadows:
        - ContextUpgradeSafe.__gap (LorcPreSale.sol#100)
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#state-variable-shadowing
INFO:Detectors:
Initializable.isConstructor (LorcPreSale.sol#49-59) is declared view but contains assembly code
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#constant-functions-changing-the-state
INFO:Detectors:
Contract locking ether found in LorcPreSale.sol:
        Contract LorcPreSale has payable functions:
         - buyWithETH (LorcPreSale.sol#508-521)
         - receive (LorcPreSale.sol#566)
        But does not have a function to withdraw the ether
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#contracts-that-lock-ether
INFO:Detectors:
LorcPreSale.buyWithETH (LorcPreSale.sol#508-521) does not use the value returned by external calls:
        -lorcToken.transfer(msg.sender,lorcValue) (LorcPreSale.sol#518)
LorcPreSale.buyWithUSDT (LorcPreSale.sol#524-539) does not use the value returned by external calls:
        -lorcToken.transfer(msg.sender,lorcValue) (LorcPreSale.sol#536)
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#unused-return
INFO:Detectors:
Initializable.isConstructor uses assembly (LorcPreSale.sol#49-59)
        - LorcPreSale.sol#57
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#assembly-usage
INFO:Detectors:
Ownable.owner (LorcPreSale.sol#142-144) should be declared external
Ownable.renounceOwnership (LorcPreSale.sol#161-164) should be declared external
Ownable.transferOwnership (LorcPreSale.sol#170-174) should be declared external
LorcPreSale.buyWithETH (LorcPreSale.sol#508-521) should be declared external
LorcPreSale.buyWithUSDT (LorcPreSale.sol#524-539) should be declared external
LorcPreSale.withdrawETH (LorcPreSale.sol#542-546) should be declared external
LorcPreSale.withdrawErc20 (LorcPreSale.sol#549-554) should be declared external
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#public-function-that-could-be-declared-as-external
INFO:Detectors:
Detected issues with version pragma in LorcPreSale.sol:
        - pragma solidity^0.5.16 (LorcPreSale.sol#2): it allows old versions
```

## 5.2 Mythril

Mythril is a security analysis tool for EVM bytecode. It detects security vulnerabilities in smart contracts built for Ethereum. It uses symbolic execution, SMT solving and taint analysis to detect a variety of security vulnerabilities.

## 5.3 Manticore

Manticore is a symbolic execution tool for analysis of smart contracts and binaries. During Symbolic Execution / EVM bytecode security assessment did not detect any high severity issue. All the considerable issues are already covered in the **Findings and Tech Details** of this report.

## 5.4 Results

Some false positive errors have been reported by the tool, all other errors have been covered in issues explained above, under low level severity issues.

## 6.0 Implementation Recommendations

Variable 'Initializable.______gap' (LorcPreSale.sol#62) is not in mixedCase
Function 'ContextUpgradeSafe.__Context_init' (LorcPreSale.sol#81-83) is not in mixedCase
Function 'ContextUpgradeSafe.__Context_init_unchained' (LorcPreSale.sol#85-88) is not in mixedCase
Function 'ContextUpgradeSafe._msgSender' (LorcPreSale.sol#91-93) is not in mixedCase
Function 'ContextUpgradeSafe._msgData' (LorcPreSale.sol#95-98) is not in mixedCase
Variable 'ContextUpgradeSafe.__gap' (LorcPreSale.sol#100) is not in mixedCase
Function 'Ownable.__Ownable_init' (LorcPreSale.sol#124-127) is not in mixedCase
Function 'Ownable.__Ownable_init_unchained' (LorcPreSale.sol#129-136) is not in mixedCase
Variable 'Ownable.__gap' (LorcPreSale.sol#176) is not in mixedCase
Function 'Pausable._pause' (LorcPreSale.sol#462-465) is not in mixedCase
Function 'Pausable._unpause' (LorcPreSale.sol#474-477) is not in mixedCase
Parameter '_lorcToken' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_usdtToken' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_ethRate' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_usdtRate' of LorcPreSale. (LorcPreSale.sol#488) is not in mixedCase
Parameter '_ethRate' of LorcPreSale.updateETHRate (LorcPreSale.sol#496) is not in mixedCase
Parameter '_usdtRate' of LorcPreSale.updateUSDTRate (LorcPreSale.sol#502) is not in mixedCase
Parameter '_usdtAmount' of LorcPreSale.buyWithUSDT (LorcPreSale.sol#524) is not in mixedCase
Parameter '_amount' of LorcPreSale.withdrawETH (LorcPreSale.sol#542) is not in mixedCase

## 7.0 Auditing Approach and Methodologies applied

Throughout the audit of the **Lorc** smart contract care was taken to ensure:

- Overall quality of code
- Utilisation of best Smart Contract Coding and Auditing practices
- Code documentation and comments match logic and expected behaviour
- Token distribution and calculations are as per intended behaviour mentioned in the whitepaper
- Implementation of token standards
- Gas Usage Efficiency.
- Code is safe from re-entrancy and other smart contract related vulnerabilities

The Smart Contract Audit included a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy regarding the scope of the smart contract. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of smart contracts and can quickly identify items that do not follow security best practices.

The following phases and associated tools were used throughout the smart contract security and quality audit:

### Structural Analysis

In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure Smart contract is structured in a way that will not result in future problems.

### Static Analysis

Static Analysis of smart contracts was done to identify contract vulnerabilities. In this step series of automated tools are used to test security of smart contracts.

### Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed; their logic was checked and compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

### Gas Consumption

In this step, we have checked the behaviour of smart contract in production. Checks were done to know how much gas gets consumed and possibilities of optimization of code to reduce gas consumption.

### Tools and Platforms used for Audit

VSCode, Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Manticore, Slither.

## 7.1 Checked Vulnerabilities

We have scanned the smart contracts for commonly known and more specific vulnerabilities.

Here are some of the commonly known vulnerabilities that have been considered as part of the smart contract audit:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with (Unexpected) Throw
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level
- Address hardcoded
- Using delete for arrays
- Integer overflow/underflow
- Locked money
- Private modifier
- Revert/require functions
- Using var
- Visibility
- Using blockhash
- Using SHA3
- Using suicide
- Using throw
- Using inline assembly

## 8.0 Limitations on Disclosure and Use of this Report.

This report contains information concerning potential details of Lorc and methods for exploiting them. Entersoft recommends that special precautions be taken to protect the confidentiality of both this document and the information contained herein. Security Assessment is an uncertain process, based on past experiences, currently available information, and known threats. All information security systems, which by their nature are dependent on human beings, are vulnerable to some degree. Therefore, while Entersoft considers the major security vulnerabilities of the analyzed systems to have been identified, there can be no assurance that any exercise of this nature will identify all possible vulnerabilities or propose exhaustive and operationally viable recommendations to mitigate those exposures. In addition, the analysis set forth herein is based on the technologies and known threats as of the date of this report. As technologies and risks change over time, the vulnerabilities associated with the operation of the Lorc Smart Contract described in this report, as well as the actions necessary to reduce the exposure to such vulnerabilities will also change. Entersoft makes no undertaking to supplement or update this report based on changed circumstances or facts of which Entersoft becomes aware after the date hereof, absent a specific written agreement to perform the supplemental or updated analysis. This report may recommend that Entersoft use certain software or hardware products manufactured or maintained by other vendors. Entersoft bases these recommendations upon its prior experience with the capabilities of those products. Nonetheless, Entersoft does not and cannot warrant that a particular product will work as advertised by the vendor, nor that it will operate in the manner intended. This report was prepared by Entersoft for the exclusive benefit of Lorc and is proprietary information. The Non-Disclosure Agreement (NDA) in effect between Entersoft and Lorc govern the disclosure of this report to all other parties including product vendors and suppliers.