CONTRACT CHECKER
Blockchain Solutions

https://t.me/contractchecker
contact@contractchecker.app
contractchecker.app
Anywhere on the Blockchain

Date: 01.11.2022

POM Lock Smart Contract Security Audit For Proof Of Memes

PROOF OF MEMES

Harry K

Harry Kedelman
General Manager

CONTRACT CHECKER

Table of Contents

CONTRACT CHECKER

Summary

This report has been prepared for Proof of Memes and focuses on overall system architecture and codebase against issues, vulnerabilities, exploitations, hacks, and back-doors in the source code of PomLock future as well as any contract dependencies that were not part of an officially recognized library. An advanced examination has been performed, utilizing Static Analysis and Manual Review techniques.

The audit result classified with categories as “Critical, Major, Medium, Minor and Informational”. Each finding evaluated by our experts and corrective/preventive recommendations provided to catch up a high level of security standard.

Auditing Approach and Applied Methodologies

The auditing process pays special attention to the following considerations:

- Code design patterns analysis in which smart contract architecture is reviewed to ensure it is structured according to industry standards and safe use of third-party smart contracts and libraries.
- Line-by-line inspection of the Smart Contract to find any potential vulnerability like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.
- Unit testing Phase, we coded/conducted custom unit tests written for each function in the contract to verify that each function works as expected.
- Automated Test performed with our in-house developed tools to identify vulnerabilities and security flaws of the Smart Contract.

Security

Identifying security related issues within each contract and the system of contract.

Sound Architecture

Evaluation of the architecture of this system through the lens of established smart contract best practices and general software best practices.

Code Correctness and Quality

A full review of the contract source code. The primary areas of focus include:

- Accuracy
- Readability
- Sections of code with high complexity
- Quantity and quality of test coverage

Overview

Project Summary

Project Name	Proof of Memes
Audited Future	PomLock
Platform	Multichain
Language	Solidity
Delivery Date	November 01, 2022
Audit Methodology	Static Analysis, Manual Review

Audited Code Package

PomLock.sol

CONTRACT CHECKER

Vulnerability Summary

Vulnerability Level	Total	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0
Major	0	0	0	0
Medium	0	0	0	0
Minor	0	0	0	0
Informational	1	1	0	0

Findings

Title	Severity	Status
Possible wasting of funds due to lack of msg.value check	Informational	Acknowledged

Possible wasting of funds due to lack of msg.value check

Severity	Location	Status
Informational	PomLock.sol:1497-1503	Acknowledged

Description

Being able send native currency value to the contract in normal token locks should be blocked.

```
1496
1497    if(address(token) != address(0)){
1498      uint256 oldRecipientBalance = IERC20(token).balanceOf(recipient);
1499      IERC20(token).safeTransferFrom(sender, recipient, amount);
1500      uint256 newRecipientBalance = IERC20(token).balanceOf(recipient);
1501      require(
1502        newRecipientBalance - oldRecipientBalance == amount,
1503        "Not enough token was transfered"
1504      );
1505    } else {
1506      require(msg.value == (amount), "Amount not match");
1507    }
1508    }
1509
```

Recommendation

We recommend adding a require as below to be able to prevent native currency value lock.

```
function safeTransferFromEnsureExactAmount(    infinite gas
  address token,
  address sender,
  address recipient,
  uint256 amount
) internal {

if(address(token) != address(0)){
  require(msg.value == 0, "Cant lock bnb in token lock");
  uint256 oldRecipientBalance = IERC20(token).balanceOf(recipient);
  IERC20(token).safeTransferFrom(sender, recipient, amount);
  uint256 newRecipientBalance = IERC20(token).balanceOf(recipient);
  require(
    newRecipientBalance - oldRecipientBalance == amount,
    "Not enough token was transfered"
  );
} else {
  require(msg.value == (amount), "Amount not match");
}
}
```

CONTRACT CHECKER

SWC Attack Test

SWC ID	Describtion	Test Result
SWC-100	Function Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed
SWC-107	Re-entrancy	LOW
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegate Call to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed
SWC-116	A control flow decision is made based on The block.timestamp environment variable	LOW
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	LOW
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions with Multiple Variable Length Arguments	Passed
SWC-134	Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects (Irrelevant/Dead Code)	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed

SWC Details

PomLock SWC.pdf

CONTRACT CHECKER

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. 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 doesn't 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 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 ContractChecker and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (ContractChecker) owe no duty of care towards you or any other person, nor does ContractChecker 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 ContractChecker 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, ContractChecker hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against ContractChecker, 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 security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed. If you have any doubt about the Genuity for this document, please check QR code: