Smart Contract Security Audit Report

[2021]

The SlowMist Security Team received the team's application for smart contract security audit of the BitTorrent on 2021.12.05. The following are the details and results of this smart contract security audit:

Token Name :

BitTorrent

The contract address :

https://tronscan.org/#/contract/TAFjULxiVgT4qWk6UZwjqwZXTSaGaqnVp4

The audit items and results :

(Other unknown security vulnerabilities are not included in the audit responsibility scope)

NO.	Audit Items	Result
1	Replay Vulnerability	Passed
2	Denial of Service Vulnerability	Passed
3	Race Conditions Vulnerability	Passed
4	Authority Control Vulnerability	Passed
5	Integer Overflow and Underflow Vulnerability	Passed
6	Gas Optimization Audit	Passed
7	Design Logic Audit	Passed
8	Uninitialized Storage Pointers Vulnerability	Passed
9	Arithmetic Accuracy Deviation Vulnerability	Passed
10	"False top-up" Vulnerability	Passed
11	Malicious Event Log Audit	Passed
12	Scoping and Declarations Audit	Passed

NO.	Audit Items	Result
13	Safety Design Audit	Passed

Audit Result : Passed

Audit Number : 0X002112060001

Audit Date : 2021.12.05 - 2021.12.06

Audit Team : SlowMist Security Team

Summary conclusion : This is a token contract that does not contain the tokenVault section. The total amount of contract tokens remains unchangeable. SafeMath security module is used, which is a recommended approach. The contract does not have the Overflow and the Race Conditions issue.

The source code:

```
//SlowMist// The contract does not have the Overflow and the Race Conditions issue
pragma solidity ^0.5.8;

interface ITRC20 {
    function transfer(address to, uint256 value) external returns (bool);
    function approve(address spender, uint256 value) external returns (bool);
    function transferFrom(address from, address to, uint256 value) external returns
(bool);
    function totalSupply() external view returns (uint256);
    function balanceOf(address who) external view returns (uint256);
    function allowance(address owner, address spender) external view returns
(uint256);
    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);
}
//SlowMist// SafeMath security module is used, which is a recommended approach
library SafeMath {

    function add(uint256 a, uint256 b, string memory errorMessage) internal pure
returns (uint256) {
        uint256 c = a + b;
        require(c >= a, errorMessage);
        return c;
    }

    function sub(uint256 a, uint256 b, string memory errorMessage) internal pure
returns (uint256) {
        require(b <= a, errorMessage);
        uint256 c = a - b;
        return c;
    }
}

contract BTT is ITRC20 {
    using SafeMath for uint256;
    string constant public name = "BitTorrent";
    string constant public symbol = "BTT";
    uint8 constant  public decimals = 18;

    uint256 private totalSupply_;
    mapping(address => uint256) private  balanceOf_;
    mapping(address => mapping(address => uint256)) private  allowance_;

    constructor(address fund) public {
        totalSupply_ = 9900 * 1e8 * 1e18 * 1e3;
        balanceOf_[fund] = totalSupply_;
        emit Transfer(address(0x00), fund, totalSupply_);
    }

    function totalSupply() public view returns (uint256) {
        return totalSupply_;
    }

    function balanceOf(address guy) public view returns (uint256){
        return balanceOf_[guy];
    }

    function allowance(address src, address guy) public view returns (uint256){
        return allowance_[src][guy];
    }

    function approve(address guy, uint256 sad) public returns (bool) {
        allowance_[msg.sender][guy] = sad;
        emit Approval(msg.sender, guy, sad);
        return true;
    }
    function transfer(address dst, uint256 sad) public returns (bool) {
        return transferFrom(msg.sender, dst, sad);
    }

    function transferFrom(address src, address dst, uint256 sad)
    public returns (bool)
    {
        require(balanceOf_[src] >= sad, "src balance not enough");

        if (src != msg.sender && allowance_[src][msg.sender] != uint256(-1)) {
            require(allowance_[src][msg.sender] >= sad, "src allowance is not
enough");
            allowance_[src][msg.sender] = allowance_[src][msg.sender].sub(sad,
"allowance subtraction overflow") ;
        }
        balanceOf_[src] = balanceOf_[src].sub(sad, "from balance subtraction
overflow");
        balanceOf_[dst] = balanceOf_[dst].add(sad, "to balance addition overflow") ;

        emit Transfer(src, dst, sad);
        return true;
    }

    function increaseAllowance(address guy, uint256 addedValue) public returns (bool)
{
        require(guy != address(0));

        allowance_[msg.sender][guy] = allowance_[msg.sender][guy].add(addedValue,
"allowance addition overflow") ;
        emit Approval(msg.sender, guy, allowance_[msg.sender][guy]);
        return true;
    }

    function decreaseAllowance(address guy, uint256 subtractedValue) public returns
(bool) {
        require(guy != address(0));

        allowance_[msg.sender][guy] = allowance_[msg.sender]
[guy].sub(subtractedValue, "allowance subtraction overflow") ;
        emit Approval(msg.sender, guy, allowance_[msg.sender][guy]);
        return true;
    }
}
```

Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.

For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

SLOWMIST

Official Website

www.slowmist.com

✉

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist