智能合约安全审计报告

慢雾安全团队于 2019-01-25 日，收到 STON 团队对 STON Token 项目智能合约安全审计申请。如下为本次智能合约安全审计细节及结果：

Token 名称：

STON

合约地址：

0x1571c3815bd411c39daed0e61b8eecb37c441486

链接地址：

https://etherscan.io/address/0x1571c3815bd411c39daed0e61b8eecb37c441486

本次审计项及结果：

（其他未知安全漏洞不包含在本次审计责任范围）

序号	审计大类	审计子类	审计结果
1	溢出审计	-	通过
2	条件竞争审计	-	通过
3	权限控制审计	权限漏洞审计	通过
		权限过大审计	通过
4	安全设计审计	Zeppelin 模块使用安全	通过
		编译器版本安全	通过
		硬编码地址安全	通过
		Fallback 函数使用安全	通过
		显现编码安全	通过
		函数返回值安全	通过
		call 调用安全	通过
5	拒绝服务审计	-	通过
6	Gas 优化审计	-	通过
7	设计逻辑审计	-	通过
8	“假充值”漏洞审计	-	通过

9	恶意 Event 事件日志审计	-	通过
10	未初始化的存储指针	-	通过
11	算术精度误差	-	通过

备注：审计意见及建议见代码注释 **//SlowMist//.....**

审计结果：**通过**

审计编号：0X001901280001

审计日期：2019 年 01 月 28 日

审计团队：慢雾安全团队

总结：此为代币(token)合约，不包含锁仓(tokenVault)部分。合约不存在溢出、条件竞争问题，综合评估合约无风险。

合约源代码如下：

```
pragma solidity ^0.5.0;
```

//SlowMist// 合约不存在溢出、条件竞争问题

```
// ----------------------------------------------------------------------------
// Symbol      : STON
// Name        : STONetwork
// Total supply: 100,000,0000.000000000000000000
// Decimals    : 18
// Copyright (c) 2018 <STONetwork>. The MIT Licence.
// ----------------------------------------------------------------------------
```

//SlowMist// 使用了 OpenZeppelin 的 SafeMath 安全模块，值得称赞的做法

```
// ----------------------------------------------------------------------------
// Safe maths
// ----------------------------------------------------------------------------
library SafeMath {
    function add(uint a, uint b) internal pure returns (uint c) {
```

```
        c = a + b;
        require(c >= a);
    }
    function sub(uint a, uint b) internal pure returns (uint c) {
        require(b <= a);
        c = a - b;
    }
    function mul(uint a, uint b) internal pure returns (uint c) {
        c = a * b;
        require(a == 0 || c / a == b);
    }
    function div(uint a, uint b) internal pure returns (uint c) {
        require(b > 0);
        c = a / b;
    }
}


// ----------------------------------------------------------------------------
// ERC Token Standard #20 Interface
// https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md
// ----------------------------------------------------------------------------
contract ERC20Interface {
    function totalSupply() public view returns (uint);
    function balanceOf(address tokenOwner) public view returns (uint balance);
    function allowance(address tokenOwner, address spender) public view returns (uint remaining);
    function transfer(address to, uint _value) public returns (bool success);
    function approve(address spender, uint _value) public returns (bool success);
    function transferFrom(address _from, address _to, uint _value) public returns (bool success);

    event Transfer(address indexed _from, address indexed _to, uint _value);
    event Approval(address indexed tokenOwner, address indexed spender, uint _value);
}


// ----------------------------------------------------------------------------
// Contract function to receive approval and execute function in one call
//
// Borrowed from MiniMeToken
// ----------------------------------------------------------------------------
contract ApproveAndCallFallBack {
    function receiveApproval(address _from, uint256 _value, address token, bytes memory data) public;
```

```
}


// ----------------------------------------------------------------------------
// Owned contract
// ----------------------------------------------------------------------------
contract Owned {
    address public owner;
    address public newOwner;

    event OwnershipTransferred(address indexed _from, address indexed _to);

    constructor() public {
        owner = msg.sender;
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    function transferOwnership(address _newOwner) public onlyOwner {
        newOwner = _newOwner;
    }
    function acceptOwnership() public {
        require(msg.sender == newOwner);
        emit OwnershipTransferred(owner, newOwner);
        owner = newOwner;
        newOwner = address(0);
    }
}


// ----------------------------------------------------------------------------
// ERC20 Token, with the addition of symbol, name and decimals and a
// fixed supply
// ----------------------------------------------------------------------------
contract STONetwork is ERC20Interface, Owned {
    using SafeMath for uint;

    string public symbol;
    string public  name;
```

```
uint8 public decimals;
uint _initialTokenNumber;
uint _totalSupply;

mapping(address => uint) balances;
mapping(address => mapping(address => uint)) allowed;

uint exchangerToken;
uint reservedToken;
uint developedToken;

address public constant developed1Address     = 0xcFCb491953Da1d10D037165dFa1298D00773fcA7;
address public constant developed2Address     = 0xA123BceDB9d2E4b09c8962C62924f091380E1Ad7;
address public constant developed3Address     = 0x51aeD4EDC28aad15C353D958c5A813aa21F351b6;
address public constant exchangedAddress     = 0x2630e8620d53C7f64f82DAEA50257E83297eE009;

// ------------------------------------------------------------------------
// Constructor
// ------------------------------------------------------------------------
constructor() public {
    symbol = "STON";
    name = "STONetwork";
    decimals = 18;
    _initialTokenNumber = 1000000000;
    _totalSupply = _initialTokenNumber * 10 ** uint(decimals);

    reservedToken = _totalSupply * 40 / 100;  // 40%

    developedToken = _totalSupply * 10 / 100; //30% 3 Address

    exchangerToken = _totalSupply * 30 / 100; // 30%

    balances[owner] = reservedToken;
    emit Transfer(address(0), owner, reservedToken);

    balances[exchangedAddress] = exchangerToken;
    emit Transfer(address(0), exchangedAddress, exchangerToken);

    balances[developed1Address] = developedToken;
    emit Transfer(address(0), developed1Address, developedToken);
    balances[developed2Address] = developedToken;
    emit Transfer(address(0), developed2Address, developedToken);
```

```
        balances[developed3Address] = developedToken;
        emit Transfer(address(0), developed3Address, developedToken);

    }


    // ------------------------------------------------------------------------
    // Total supply
    // ------------------------------------------------------------------------
    function totalSupply() public view returns (uint) {
        return _totalSupply;
    }


    // ------------------------------------------------------------------------
    // Get the token balance for account `tokenOwner`
    // ------------------------------------------------------------------------
    function balanceOf(address _owner) public view returns (uint balance) {
        return balances[_owner];
    }


    // ------------------------------------------------------------------------
    // Transfer the balance from token owner's account to `to` account
    // - Owner's account must have sufficient balance to transfer
    // - 0 value transfers are allowed
    // ------------------------------------------------------------------------
    function transfer(address _to, uint _value) public returns (bool success) {
        require(balances[msg.sender] >= _value);
        require(_to != address(0)); //SlowMist// 这类检查很好，避免用户失误导致 Token 转丢
        balances[msg.sender] = balances[msg.sender].sub(_value);
        balances[_to] = balances[_to].add(_value);
        emit Transfer(msg.sender, _to, _value);
        return true;  //SlowMist// 返回值符合 EIP20 规范
    }


    // ------------------------------------------------------------------------
    // Token owner can approve for `spender` to transferFrom(...) `tokens`
    // from the token owner's account
```

```
//
// https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md
// recommends that there are no checks for the approval double-spend attack
// as this should be implemented in user interfaces
// ------------------------------------------------------------------------
function approve(address _spender, uint _value) public returns (bool success) {
    require(_spender != address(0)); //SlowMist// 这类检查很好，避免用户授权错误白白消耗 Gas
    allowed[msg.sender][_spender] = _value;
    emit Approval(msg.sender, _spender, _value);
    return true;  //SlowMist// 返回值符合 EIP20 规范
}


// ------------------------------------------------------------------------
// Transfer `tokens` from the `from` account to the `to` account
//
// The calling account must already have sufficient tokens approve(...)-d
// for spending from the `from` account and
// - From account must have sufficient balance to transfer
// - Spender must have sufficient allowance to transfer
// - 0 value transfers are allowed
// ------------------------------------------------------------------------
function transferFrom(address _from, address _to, uint _value) public returns (bool success) {
    require(_to != address(0)); //SlowMist// 这类检查很好，避免用户失误导致 Token 转丢
    require(balances[_from] >= _value);
    require(allowed[_from][msg.sender] >= _value);

    balances[_from] = balances[_from].sub(_value);
    allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
    balances[_to] = balances[_to].add(_value);
    emit Transfer(_from, _to, _value);
    return true; //SlowMist// 返回值符合 EIP20 规范
}


// ------------------------------------------------------------------------
// Returns the amount of tokens approved by the owner that can be
// transferred to the spender's account
```

```
    // ------------------------------------------------------------------------
    function allowance(address _owner, address _spender) public view returns (uint remaining) {
        return allowed[_owner][_spender];
    }


    // ------------------------------------------------------------------------
    // Token owner can approve for `spender` to transferFrom(...) `tokens`
    // from the token owner's account. The `spender` contract function
    // `receiveApproval(...)` is then executed
    // ------------------------------------------------------------------------
    function approveAndCall(address _spender, uint _value, bytes memory data) public returns (bool success)
{
        allowed[msg.sender][_spender] = _value;
        emit Approval(msg.sender, _spender, _value);
        ApproveAndCallFallBack(_spender).receiveApproval(msg.sender, _value, address(this), data);
        return true;
    }


    // ------------------------------------------------------------------------
    // Don't accept ETH
    // ------------------------------------------------------------------------
    function () external payable {
        revert();
    }


    // ------------------------------------------------------------------------
    // Owner can transfer out any accidentally sent ERC20 tokens
    // ------------------------------------------------------------------------
    function transferAnyERC20Token(address _tokenAddress, uint _value) public onlyOwner returns (bool
success) {
        return ERC20Interface(_tokenAddress).transfer(owner, _value);
    }
}
```

慢雾科技
slow mist

官方网址

www.slowmist.com

电子邮箱

team@slowmist.com

微信公众号