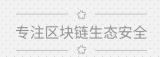


# 智能合约安全审计报告





慢雾安全团队于 2018-07-04 日,收到 X-power 团队对 XPO 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:

XPO

合约地址:

0x2259133B1Aa6B7f0B101559BF76091f383141B69

链接地址:

https://etherscan.io/address/0x2259133B1Aa6B7f0B101559BF76091f383141B69#code

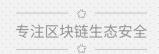
### 本次审计项及结果:

(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(良) |
|    |          | 编译器版本安全         | 通过(良) |
|    |          | 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(良) |
|    |          | 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    |

备注:审计意见及建议见代码注释 //SlowMist//.....





审计结果:通过(良)

审计编号: 0X001807050002

审计日期: 2018年07月05日

审计团队:慢雾安全团队

(声明:慢雾仅就本报告出具前已经发生或存在的事实出具本报告,并就此承担相应责任。对于出具以后发生或存在的事实,慢雾无法判断其智能合约安全状况,亦不对此承担责任。本报告所作的安全审计分析及其他内容,仅基于信息提供者截至本报告出具时向慢雾提供的文件和资料(简称"已提供资料")。慢雾假设:已提供资料不存在缺失、被篡改、删减或隐瞒的情形。如已提供资料信息缺失、被篡改、删减、隐瞒或反映的情况与实际情况不符的,慢雾对由此而导致的损失和不利影响不承担任何责任。)

总结:此为代币(token)合约,不包含锁仓(tokenVault)部分。综合评估合约在对接去中心化 DApp 时可能存在兼容性问题(取决于 DApp 部署时的编译器版本),其他方面没有安全风险。

合约源代码如下:

```
pragma solidity ^0.4.16; //SlowMist// 编译器版本过低,存在 ZeroFunctionSelector 缺陷(低危)
```

### //SlowMist// 合约不存在溢出、条件竞争问题

#### //SlowMist// 建议引入 OpenZeppelin 的 SafeMath 安全模块

```
interface tokenRecipient { function receiveApproval(address _from, uint256 _value, address _token, bytes
_extraData) public; }

contract TokenERC20 {
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid changing it
    uint256 public totalSupply;

// This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public allowance;

// This generates a public event on the blockchain that will notify clients
    event Transfer(address indexed from, address indexed to, uint256 value);
```



```
// This notifies clients about the amount burnt
event Burn(address indexed from, uint256 value);
 * Constrctor function
 * Initializes contract with initial supply tokens to the creator of the contract
function TokenERC20(
   uint256 initialSupply,
   string tokenName,
   string tokenSymbol
) public {
   totalSupply = initialSupply * 10 ** uint256(decimals); // Update total supply with the decimal amount
   balanceOf[msg.sender] = totalSupply;
                                                     // Give the creator all initial tokens
   name = tokenName;
                                                     // Set the name for display purposes
   symbol = tokenSymbol;
                                                     // Set the symbol for display purposes
}
 * Internal transfer, only can be called by this contract
function _transfer(address _from, address _to, uint _value) internal {
   // Prevent transfer to 0x0 address. Use burn() instead
   require(_to != 0x0); //SlowMist// 这类检查很好,避免用户失误导致 Token 转丢
   // Check if the sender has enough
   require(balanceOf[_from] >= _value);
   // Check for overflows
   require(balanceOf[_to] + _value > balanceOf[_to]); //SlowMist// 溢出检查
   // Save this for an assertion in the future
   uint previousBalances = balanceOf[_from] + balanceOf[_to];
   // Subtract from the sender
   balanceOf[_from] -= _value;
   // Add the same to the recipient
   balanceOf[_to] += _value;
   Transfer(_from, _to, _value);
   // Asserts are used to use static analysis to find bugs in your code. They should never fail
   assert(balanceOf[_from] + balanceOf[_to] == previousBalances); //SlowMist// 前面 require 里
```



# 已经校验了溢出,这边的检查可移除,以节省 Gas } /\*\* \* Transfer tokens \* Send `\_value` tokens to `\_to` from your account \* @param \_to The address of the recipient \* @param \_value the amount to send \*/ function transfer(address \_to, uint256 \_value) public { \_transfer(msg.sender, \_to, \_value); //SlowMist// 没有返回值,不符合 EIP20 规范,对接去中心化交易所时存在兼容性问题 } \* Transfer tokens from other address \* Send `\_value` tokens to `\_to` on behalf of `\_from` \* @param \_from The address of the sender \* @param \_to The address of the recipient \* @param \_value the amount to send \*/ function transferFrom(address \_from, address \_to, uint256 \_value) public returns (bool success) { require(\_value <= allowance[\_from][msg.sender]); // Check allowance</pre> allowance[\_from][msg.sender] -= \_value; \_transfer(\_from, \_to, \_value); return true; //SlowMist// 返回值符合 EIP20 规范 } \* Set allowance for other address \* Allows `\_spender` to spend no more than `\_value` tokens on your behalf \* @param \_spender The address authorized to spend \* @param \_value the max amount they can spend



```
function approve(address _spender, uint256 _value) public
       returns (bool success) {
       allowance[msg.sender][_spender] = _value;
       return true; //SlowMist// 返回值符合 EIP20 规范
   }
    * Set allowance for other address and notify
    * Allows `_spender` to spend no more than `_value` tokens on your behalf, and then ping the contract
about it
    * @param _spender The address authorized to spend
    * @param _value the max amount they can spend
    * @param _extraData some extra information to send to the approved contract
   function approveAndCall(address _spender, uint256 _value, bytes _extraData)
       returns (bool success) {
       tokenRecipient spender = tokenRecipient(_spender);
       if (approve(_spender, _value)) {
           spender.receiveApproval(msg.sender, _value, this, _extraData);
           return true;
       }
   }
     * Destroy tokens
     * Remove `_value` tokens from the system irreversibly
     * @param _value the amount of money to burn
    */
   function burn(uint256 _value) public returns (bool success) {
       require(balanceOf[msg.sender] >= _value); // Check if the sender has enough
       balanceOf[msg.sender] -= _value;
                                                 // Subtract from the sender
       totalSupply -= _value;
                                                 // Updates totalSupply
       Burn(msg.sender, _value);
       return true;
   }
```





### //SlowMist// 因为 burnFrom()和 transferFrom()共享 approve()的 allowance 额度,假如代理商

### 作恶,存在被恶意燃烧的可能

```
* Destroy tokens from other account
     * Remove `_value` tokens from the system irreversibly on behalf of `_from`.
    * @param _from the address of the sender
    * @param _value the amount of money to burn
   function burnFrom(address _from, uint256 _value) public returns (bool success) {
       require(balanceOf[_from] >= _value);
                                                         // Check if the targeted balance is enough
       require(_value <= allowance[_from][msg.sender]);  // Check allowance</pre>
       balanceOf[_from] -= _value;
                                                         // Subtract from the targeted balance
       allowance[_from][msg.sender] -= _value;
                                                        // Subtract from the sender's allowance
       totalSupply -= _value;
                                                         // Update totalSupply
       Burn(_from, _value);
       return true;
   }
}
```



## 官方网址

www.slowmist.com

## 电子邮箱

team@slowmist.com

微信公众号

