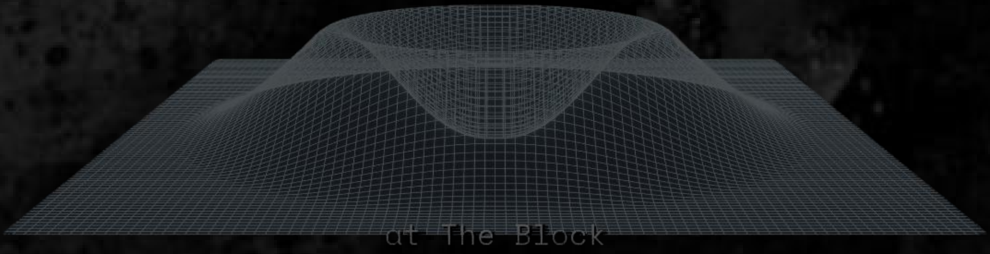
CHAINLION

MOY

Smart Contract Audit Report

OCT 12th，2022

NO.0C002210120001



CATALOGUE

1. PROJECT SUMMARY

Entry type	Specific description
Entry name	MOY
Project type	BEP-20
Application platform	BSC
DawnToken	0xa0cC4414019471bef0fe0b07a76dA1F7CDc4cCf7

2. AUDIT SUMMARY

Entry type	Specific description
Project cycle	OCT/08/2022-OCT/12/2022
Audit method	Black box test、White box test、Grey box test
Auditors	THREE

3. VULNERABILITY SUMMARY

Audit results are as follows:

Entry type	Specific description
Serious vulnerability	0
High risk vulnerability	0
Moderate risk	2
Low risk vulnerability	6

Security vulnerability rating description：

1) **Serious vulnerability：** Security vulnerabilities that can directly cause token contracts or user capital losses， For example: shaping overflow vulnerability、

Fake recharge vulnerability、Reentry attacks, vulnerabilities, etc.

2) **High risk vulnerability：** Security vulnerabilities that can directly cause the contract to fail to work normally, such as reconstructed smart contract caused by constructor design error, denial of service vulnerability caused by unreasonable design of require / assert detection conditions, etc.

3) **Moderate risk：** Security problems caused by unreasonable business logic design, such as accuracy problems caused by unreasonable numerical operation sequence design, variable ambiguous naming, variable coverage, call injection, conditional competition, etc.

4) **Low risk vulnerability：** Security vulnerabilities that can only be triggered by users with special permissions, such as contract backdoor vulnerability, duplicate name pool addition vulnerability, non-standard contract coding, contract detection bypass, lack of necessary events for key state variable change, and security vulnerabilities that are harmful in theory but have harsh utilization conditions.

4. EXECUTIVE SUMMARY

This report is prepared for **MOY** smart contract， The purpose is to find the security vulnerabilities and non-standard coding problems in the smart contract through the security audit of the source code of the smart contract. This audit mainly involves the following test methods:

White box test

Conduct security audit on the source code of smart contract and check the security issues such as coding specification, DASP top 10 and business logic design

Grey box test

Deploy smart contracts locally and conduct fuzzy testing to check function robustness, function call permission and business logic security

Black box test

Conduct security test attacks on smart contracts from the perspective of attackers, combined with black-and-white and testing techniques, to check whether there are exploitable vulnerabilities.

This audit report is subject to the latest contract code provided by the current project party, does not include the newly added business logic function module after the contract upgrade, does not include new attack methods in the future, and does not include web front-end security and server-side security.

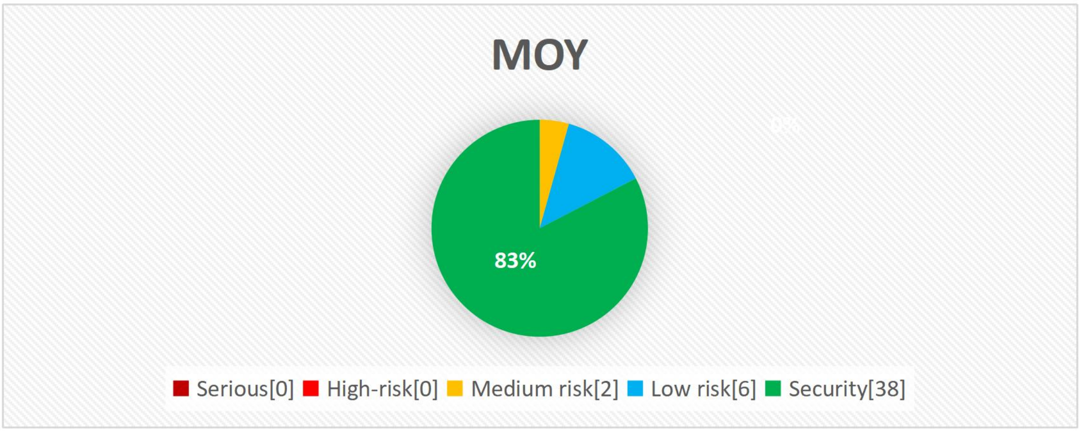
5. Directory structure

```
├─contract
|       Address.sol
|       BitMaps.sol
|       Context.sol
|       IERC20.sol
|       IERC20Metadata.sol
|       IPancakeFactory.sol
|       IPancakePair.sol
|       IPancakeRouter.sol
|       Math.sol
|       Moy.sol
|       Ownable.sol
|       PancakeLibrary.sol
|       SafeMath.sol
```

6. File hashes

Contract	SHA1 Checksum
Address.sol	EC295C93CB5EE1BC1DD7E1DB327EACB5704F1694
BitMaps.sol	835D19020B6453A278D2710F28EAAB93DBC64195
Context.sol	37CCEF725837F7FA2F3DD5446C62A9776B435B6F
IERC20.sol	3A4C9C7E4D7BDC778A551C2601C77FC0A8043614
IERC20Metadata.sol	F0BBEA0ED52A83D99E52D7A30FE75633176E82BA
IPancakeFactory.sol	93BFFF9F8928BA4A369023568081D5E82F301BB3
IPancakePair.sol	675BD8F857807F3D3BFBDC13B33EFBC5F492D2B2
IPancakeRouter.sol	94177930CC53BAA6A0AC20C1376AFE47D4CB77ED
Math.sol	40B428DDB47C653F7B3400D3B05F1A14A985B99D
Moy.sol	D7F7660515EC6CC0B69FC8822C277EB7EA488D19
Ownable.sol	FCAB37D90398E8463C3FBA93801109A46BB3846B
PancakeLibrary.sol	670E8A0FA6053CD0F2132584531FD94ECCE92FDF
SafeMath.sol	6CF75DBD13453DD8F7D5348BA5BC2BAB9215B2A5

7. Vulnerability distribution



8. Audit content

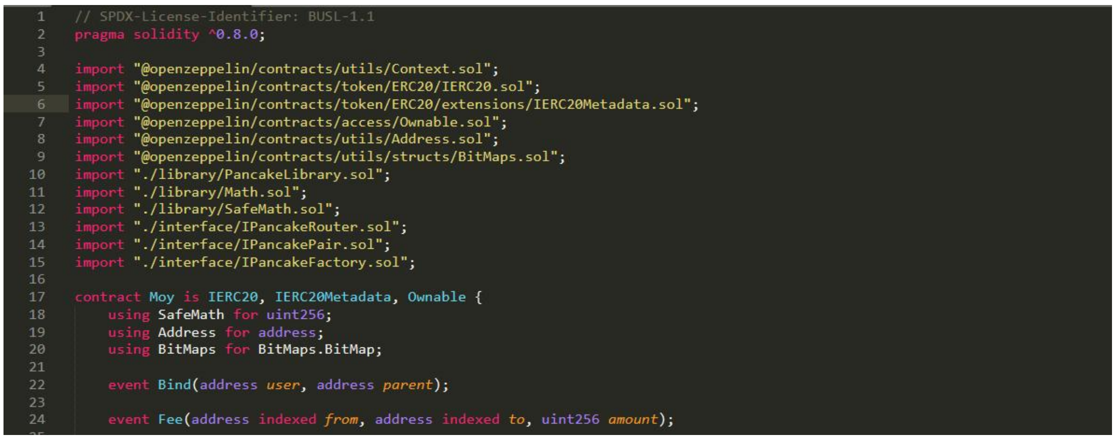
8.1. Coding specification

Smart contract supports contract development in programming languages such as solid, Vyper, C + +, Python and rust. Each programming language has its own coding specification. In the development process, the coding specification of the development language should be strictly followed to avoid security problems such as business function design defects.

8.1.1. Compiler Version【security】

Audit description： The compiler version should be specified in the smart contract code. At the same time, it is recommended to use the latest compiler version. The old version of the compiler may cause various known security problems. At present, the latest version is v 0.8 x. And this version has been protected against shaping overflow.

Audit results： According to the audit, the compiler version used in the smart contract code is 0.8.0, so there is no such security problem.



```
// SPDX-License-Identifier: BUSL-1.1
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/utils/Context.sol";
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol";
import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/utils/Address.sol";
import "@openzeppelin/contracts/utils/structs/BitMaps.sol";
import "./library/PancakeLibrary.sol";
import "./library/Math.sol";
import "./library/SafeMath.sol";
import "./interface/IPancakeRouter.sol";
import "./interface/IPancakePair.sol";
import "./interface/IPancakeFactory.sol";

contract Moy is IERC20, IERC20Metadata, Ownable {
    using SafeMath for uint256;
    using Address for address;
    using BitMaps for BitMaps.BitMap;

    event Bind(address user, address parent);

    event Fee(address indexed from, address indexed to, uint256 amount);
```

Safety advice： NONE.

8.1.2. Return value verification【security】

Audit description： Smart contract requires contract developers to strictly follow EIP / tip and other standards and specifications during contract development. For transfer, transferfrom and approve functions, Boolean values should be returned to feed back the final execution results. In the smart contract, the relevant business logic code often calls the transfer or transferfrom function to transfer. In this case, the return value involved in the transfer operation should be strictly checked to determine whether the transfer is successful or not, so as to avoid security vulnerabilities such as false recharge caused by the lack of return value verification.

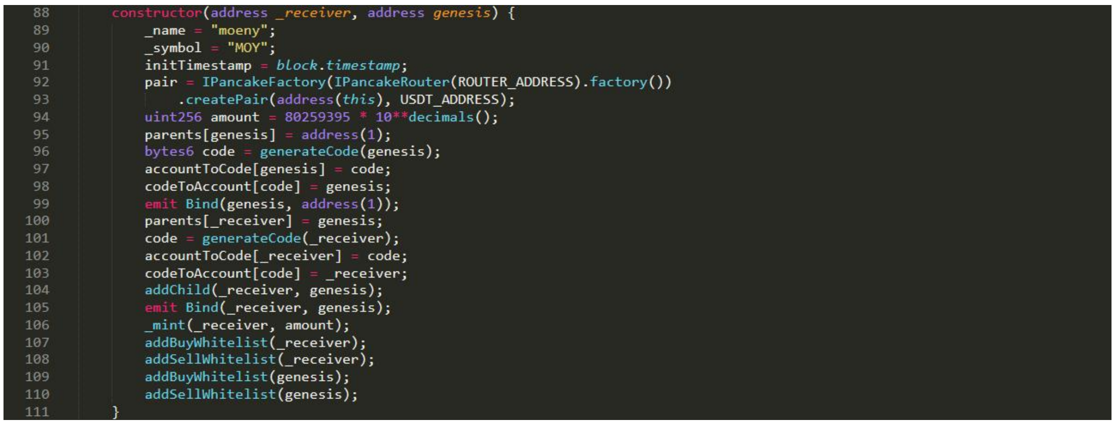
Audit results： According to the audit, there is no embedded function calling the official standards transfer and transferfrom in the smart contract, so there is no such security problem.

Safety advice： NONE.

8.1.3. Constructor writing【security】

Audit description： In solid v0 The smart contract written by solidity before version 4.22 requires that the constructor must be consistent with the contract name. When the constructor name is inconsistent with the contract name, the constructor will become an ordinary public function. Any user can call the constructor to initialize the contract. After version V 0.4.22, The constructor name can be replaced by constructor, so as to avoid the coding problems caused by constructor writing.

Audit results： After audit, the constructor in the smart contract is written correctly, and there is no such security problem.



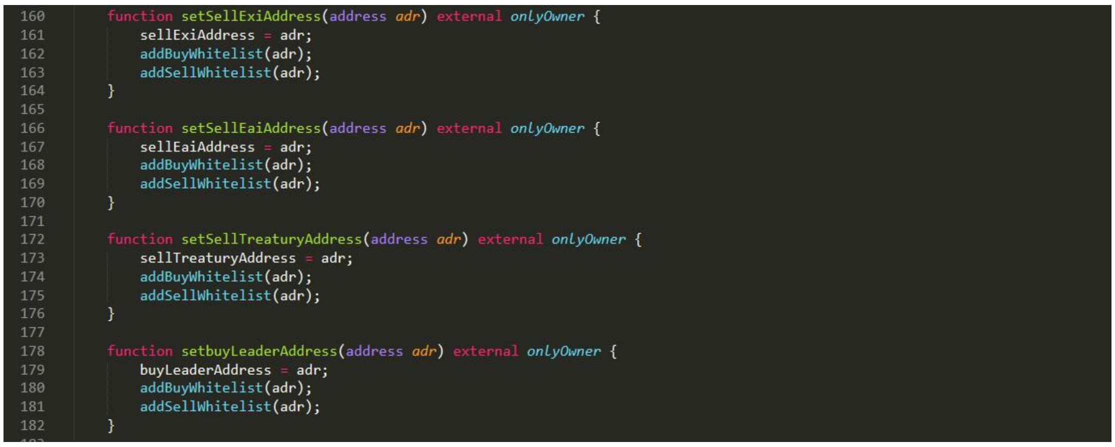
```
    constructor(address _receiver, address genesis) {
        _name = "moeny";
        _symbol = "MOY";
        initTimestamp = block.timestamp;
        pair = IPancakeFactory(IPancakeRouter(ROUTER_ADDRESS).factory())
            .createPair(address(this), USDT_ADDRESS);
        uint256 amount = 80259395 * 10**decimals();
        parents[genesis] = address(1);
        bytes6 code = generateCode(genesis);
        accountToCode[genesis] = code;
        codeToAccount[code] = genesis;
        emit Bind(genesis, address(1));
        parents[_receiver] = genesis;
        code = generateCode(_receiver);
        accountToCode[_receiver] = code;
        codeToAccount[code] = _receiver;
        addChild(_receiver, genesis);
        emit Bind(_receiver, genesis);
        _mint(_receiver, amount);
        addBuyWhitelist(_receiver);
        addSellWhitelist(_receiver);
        addBuyWhitelist(genesis);
        addSellWhitelist(genesis);
    }
```

Safety advice： NONE.

8.1.4. Key event trigger【Low risk】

Audit description： Most of the key global variable initialization or update operations similar to setXXX exist in the smart contract. It is recommended to trigger the corresponding event through emit when operating on similar key events.

Audit results： After audit, the necessary event design and event trigger are lacking.



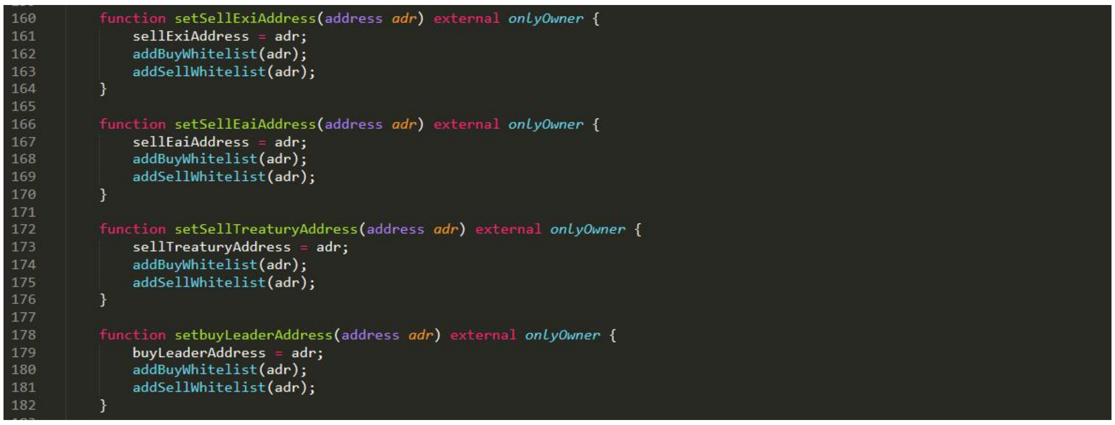
```
    function setSellExiAddress(address adr) external onlyOwner {
        sellExiAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellEaiAddress(address adr) external onlyOwner {
        sellEaiAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellTreaturyAddress(address adr) external onlyOwner {
        sellTreaturyAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setbuyLeaderAddress(address adr) external onlyOwner {
        buyLeaderAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }
```

Safety advice： When operating on similar key events, trigger the corresponding event through emit.

8.1.5. Address non-zero check【Low risk】

Audit description： The smart contract initializes the key information of the contract through the constructor. When it comes to address initialization, the address should be non-zero checked to avoid irreparable economic losses.

Audit results： It is audited that some functions lack non-zero checking of addresses.



```
    function setSellExiAddress(address adr) external onlyOwner {
        sellExiAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellEaiAddress(address adr) external onlyOwner {
        sellEaiAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellTreaturyAddress(address adr) external onlyOwner {
        sellTreaturyAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setbuyLeaderAddress(address adr) external onlyOwner {
        buyLeaderAddress = adr;
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }
```

Safety advice： Non zero check of address.

8.1.6. Code redundancy check【security】

Audit description： The deployment and execution of smart contracts need to consume certain gas costs. The business logic design should be optimized as much as possible, while avoiding unnecessary redundant code to improve efficiency and save costs.

Audit results： After audit, there is no such security problem.

Safety advice： **NONE.**

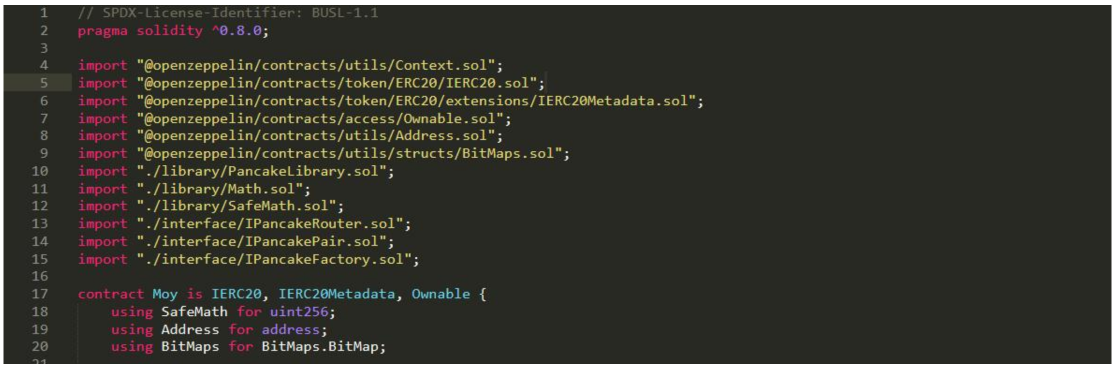
8.2. Coding design

DASP top 10 summarizes the common security vulnerabilities of smart contracts. Smart contract developers can study smart contract security vulnerabilities before developing contracts to avoid security vulnerabilities during contract development. Contract auditors can quickly audit and check the existing security vulnerabilities of smart contracts according to DASP top 10.

8.2.1. Shaping overflow detection【security】

Audit description： Solid can handle 256 digits at most. When the number is unsigned, the maximum value will overflow by 1 to get 0, and 0 minus 1 will overflow to get the maximum value. The problem of shaping overflow often appears in the relevant logic code design function modules such as transaction transfer, reward calculation and expense calculation. The security problems caused by shaping overflow are also very serious, such as excessive coinage, high sales and low income, excessive distribution, etc. the problem of shaping overflow can be solved by using solid V 0.8 X version or by using the safemath library officially provided by openzenppelin.

Audit results： According to the audit, the smart contract is applicable to the compiler of version 0.8.0, and the safemath library is used for numerical operation, which better prevents the problem of shaping overflow.



```
// SPDX-License-Identifier: BUSL-1.1
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/utils/Context.sol";
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol";
import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/utils/Address.sol";
import "@openzeppelin/contracts/utils/structs/BitMaps.sol";
import "./library/PancakeLibrary.sol";
import "./library/Math.sol";
import "./library/SafeMath.sol";
import "./interface/IPancakeRouter.sol";
import "./interface/IPancakePair.sol";
import "./interface/IPancakeFactory.sol";

contract Moy is IERC20, IERC20Metadata, Ownable {
    using SafeMath for uint256;
    using Address for address;
    using BitMaps for BitMaps.BitMap;
```

Safety advice： **NONE.**

8.2.2. Reentry detection【security】

Audit description： The in solidity provides call Value(), send(), transfer() and other functions are used for transfer operation. When call When value() sends ether, it will send all gas for transfer operation by default. If the transfer function can be called recursively again through call transfer, it can cause reentry attack.

Audit results： After audit, there is no such security problem.

Safety advice： **NONE.**

8.2.3. Rearrangement attack detection【security】

Audit description： Rearrangement attack means that miners or other parties try to compete with smart contract participants by inserting their information into the list or mapping, so that attackers have the opportunity to store their information in the contract.

Audit results： After audit, there is no such security problem.

Safety advice： NONE.

8.2.4. Replay Attack Detection【security】

Audit description： When the contract involves the business logic of delegated management, attention should be paid to the non reusability of verification to avoid replay attacks. In common asset management systems, there are often delegated management businesses. The principal gives the assets to the trustee for management, and the principal pays a certain fee to the trustee. In similar delegated management scenarios, it is necessary to ensure that the verification information will become invalid once used.

Audit results： After audit, there is no such security problem.

Safety advice： NONE.

8.2.5. False recharge detection【security】

Audit description：When a smart contract uses the transfer function for transfer, it should use require / assert to strictly check the transfer conditions. It is not recommended to use if Use mild judgment methods such as else to check, otherwise it will misjudge the success of the transaction, resulting in the security problem of false recharge.

Audit results： After audit, there is no such security problem.

Safety advice： NONE.

8.2.6. Access control detection 【security】

Audit description：Solid provides four function access domain Keywords: public, private, external and internal to limit the scope of function. In the smart contract, the scope of function should be reasonably designed to avoid the security risk of improper access control. The main differences of the above four keywords are as follows:

1．public： The marked function or variable can be called or obtained by any account, which can be a function in the contract, an external user or inherit the function in the contract

2．external： The marked functions can only be accessed from the outside and cannot be called directly by the functions in the contract, but this can be used Func() calls this function as an external call

3．private： Marked functions or variables can only be used in this contract (Note: the limitation here is only at the code level. Ethereum is a public chain, and anyone can directly obtain the contract status information from the chain)

4．internal: It is generally used in contract inheritance. The parent contract is marked as an internal state variable or function, which can be directly accessed and called by the child contract (it cannot be directly obtained and called externally)

Audit results： After audit, there is no such security problem.

Safety advice： NONE.

8.2.7. Denial of service detection 【security】

Audit description：Denial of service attack is a DoS attack on Ethereum contract, which makes ether or gas consume a lot. In more serious cases, it can make the contract code logic unable to operate normally. The common causes of DoS attack are: unreasonable design of require check condition, uncontrollable number of for cycles, defects in business logic design, etc.

Audit results： After audit, there is no such security problem.

Safety advice： NONE.

8.2.8. Conditional competition detection 【security】

Audit description： The Ethereum node gathers transactions and forms them into blocks. Once the miners solve the consensus problem, these transactions are considered effective. The miners who solve the block will also choose which transactions from the mine pool will be included in the block. This is usually determined by gasprice transactions. Attackers can observe whether there are transactions in the transaction pool that may contain problem solutions, After that, the attacker can obtain data from this transaction, create a higher-level transaction gasprice, and include its transaction in a block before the original, so as to seize the original solution.

Audit results： After audit, there is no such security problem.

Safety advice：NONE.

8.2.9. Consistency detection【security】

Audit description： The update logic in smart contract (such as token quantity update, authorized transfer quota update, etc.) is often accompanied by the check logic of the operation object (such as anti overflow check, authorized transfer quota check, etc.), and when the update object is inconsistent with the check object, the check operation may be invalid, Thus, the conditional check logic is ignored and unexpected logic is executed. For example, the authorized transfer function function function transfer from (address _from, address _to, uint256 _value) returns (bool success) is used to authorize others to transfer on behalf of others. During transfer, the permission [_from] [MSG. Sender] authorized transfer limit will be checked, After passing the check, the authorized transfer limit will be updated at the same time of transfer. When the update object in the update logic is inconsistent with the check object in the check logic, the authorized transfer limit of the authorized transfer user will not change, resulting in that the authorized transfer user can transfer all the assets of the authorized account.

Audit results： After audit, there is no such security problem.

Safety advice：NONE.

8.2.10. Variable coverage detection【security】

Audit description： Smart contracts allow inheritance relationships, in which the child contract inherits all the methods and variables of the parent contract. If a global variable with the same name as the parent contract is defined in the child contract, it may lead to variable coverage and corresponding asset losses.

Audit results： After audit, there is no such security problem.

Safety advice：NONE.

8.2.11. Random number detection【security】

Audit description： Random numbers are often used in smart contracts. When designing the random number generation function, the generation and selection of random seeds should avoid the data information that can be queried on the blockchain, such as block Number and block Timestamp et al. These data are vulnerable to the influence of miners, resulting in the predictability of random numbers to a certain extent.

Audit results： After audit, there is no such security problem.

Safety advice：NONE.

8.2.12. Numerical operation detection【security】

Audit description： Solidity supports addition, subtraction, multiplication, division and other conventional numerical operations, but solidty does not support floating-point types. When multiplication and division operations exist at the same time, the numerical operation order should be adjusted reasonably to reduce the error as much as possible.

Audit results： After audit, there is no such security problem.

Safety advice： **NONE.**

8.2.13. Call injection detection 【security】

Audit description： In the solid language, you can call a contract or a method of a local contract through the call method. There are roughly two ways to call: < address > Call (method selector, arg1, arg2,...) or < address > Call (bytes). When using call call, we can pass method selectors and parameters by passing parameters, or directly pass in a byte array. Based on this function, it is recommended that strict permission check or hard code the function called by call when using call function call.

Audit results： After audit, there is no such security problem.

Safety advice： **NONE.**

8.3. Business logic

Business logic design is the core of smart contract. When using programming language to develop contract business logic functions, developers should fully consider all aspects of the corresponding business, such as parameter legitimacy check, business permission design, business execution conditions, interaction design between businesses, etc.

8.3.1. Constructor initialization logic 【security】

Audit description： Conduct security audit on the constructor initialization and business logic design in the contract, and check whether the initialization value is consistent with the requirements document.

Audit results: The constructor initialization business logic design in the contract is correct, and no relevant security risks are found.

Code file: Moy.sol 88~111

Code information:

```
contract Moy is IERC20, IERC20Metadata, Ownable {
    using SafeMath for uint256;
    using Address for address;
    using BitMaps for BitMaps.BitMap;

    event Bind(address user, address parent);

    event Fee(address indexed from, address indexed to, uint256 amount);

    address private constant ROUTER_ADDRESS =
        0x10ED43C718714eb63d5aA57B78B54704E256024E;    //PancakeRouter contract address

    address private constant USDT_ADDRESS =
        0x55d398326f99059fF775485246999027B3197955;   //USDT contract address

    uint256 public constant PERIOD = 12 hours;   //12 hour cycle

    uint256 public constant RATE1 = 10;   //rate
    uint256 public constant RATE2 = 11;
    uint256 public constant RATE3 = 12;
    uint256 public constant RATE4 = 14;
    uint256 public constant RATE5 = 17;
    mapping(address => uint256) private _balances;    //Retrieve the number of assets held by the
corresponding address

    mapping(address => mapping(address => uint256)) private _allowances;     //Retrieve
Authorization Limit

    uint256 private _totalSupply; //Total number of tokens issued

    string private _name;           //Token name
    string private _symbol;         //Token symbol
```

```
mapping(address => address) public parents;    //Hierarchy, retrieving parent

mapping(address => bytes6) public accountToCode;    //accountToCode

mapping(bytes6 => address) public codeToAccount;    //codeToAccount

mapping(bytes32 => address[]) public children;    //Sublevel

address public sellExiAddress = address(2);    //EXI Repo Address

address public sellEaiAddress = address(3);    //EAI Repo Address

address public sellTreaturyAddress = address(4); //Address for sale

address public buyLeaderAddress = address(5); //Purchase address

uint256 public initTimestamp; //Initialization timestamp

address public pair;    //Deal to contract address

mapping(uint24 => uint256) public lpPerEpoch;

mapping(uint24 => uint256) public moyPerEpoch;

mapping(address => uint256) public liquidityPerAccount;

mapping(address => uint24) public lastEpochPerAccount;

mapping(address => uint256) public buymoyPerAccount;

mapping(address => uint256) public feePerAccount;

address[] public lpAccounts;

BitMaps.BitMap private hasLp;

BitMaps.BitMap private sellWhitelist; //Seller white list address

BitMaps.BitMap private buyWhitelist;    //Buyer white list address

constructor(address _receiver, address genesis) { //Constructor Initializers
        _name = "moeny";    //Initial contract name
        _symbol = "MOY";    //Initialize Tokens symbol
        initTimestamp = block.timestamp;    //Update initialization timestamp
        pair = IPancakeFactory(IPancakeRouter(ROUTER_ADDRESS).factory())
            .createPair(address(this), USDT_ADDRESS);     //Create a transaction pair between
MOY and USDT and return the contract address of the transaction pair
        uint256 amount = 80259395 * 10**decimals();    //Calculate total issuance
        parents[genesis] = address(1); //Initialize the parent address of genesis
        bytes6 code = generateCode(genesis); //Generate Genesis Code
        accountToCode[genesis] = code; //Set the code corresponding to the genesis account
        codeToAccount[code] = genesis; //Set the account address corresponding to the code
        emit Bind(genesis, address(1));
        parents[_receiver] = genesis;    //Settings_ The parent address of the receiver is genesis
        code = generateCode(_receiver); //Generate_ Code corresponding to receiver
        accountToCode[_receiver] = code; //Settings_ Code of receiver
        codeToAccount[code] = _receiver; //Set the account address corresponding to the code
        addChild(_receiver, genesis);     //Build parent-child level, genesis is_ Parent of receiver
        emit Bind(_receiver, genesis);
        _mint(_receiver, amount);    //Number of initial tokens (additional tokens)
        addBuyWhitelist(_receiver);    //Add to white list address
        addSellWhitelist(_receiver);    //Add to white list address
        addBuyWhitelist(genesis);    //Add to white list address
        addSellWhitelist(genesis);    //Add to white list address

        }
```

Safety advice：NONE.

8.3.2. Logical design of contract address check【security】

Audit description： Audit the detection business logic design of whether the function caller is the contract address in the contract, and check whether there are design defects.

Audit results： In the contract, the official Openzeppelin library is used to detect whether the function caller is the contract address. The detection of the function caller not only depends on the value of extcodesize, but also detects address.code.length. There is no security risk.

Code file： Address.sol 36~42

Code information：

```
    /**
     * @dev Returns true if `account` is a contract.
     *
     * [IMPORTANT]
     * ====
     * It is unsafe to assume that an address for which this function returns
     * false is an externally-owned account (EOA) and not a contract.
     *
     * Among others, `isContract` will return false for the following
     * types of addresses:
     *
     *    - an externally-owned account
     *    - a contract in construction
     *    - an address where a contract will be created
     *    - an address where a contract lived, but was destroyed
     * ====
     *
     * [IMPORTANT]
     * ====
     * You shouldn't rely on `isContract` to protect against flash loan attacks!
     *
     * Preventing calls from contracts is highly discouraged. It breaks composability, breaks support
for smart wallets
     * like Gnosis Safe, and does not provide security since it can be circumvented by calling from a
contract
     * constructor.
     * ====
     */
    function isContract(address account) internal view returns (bool) {
        // This method relies on extcodesize/address.code.length, which returns 0
        // for contracts in construction, since the code is only stored at the end
        // of the constructor execution.

        return account.code.length > 0;
    }
```

Safety advice： NONE.

8.3.3. Contract Owner Related Logic【security】

Audit results： Conduct security audit on business logic design such as owner update, owner permission waiver and owner permission check in the contract to check whether there are design defects.

Audit results： The business logic design of owner update, owner permission waiver and owner permission check in the contract is reasonable, and no related security risks are found.

Code file： Ownable.sol 20~83

Code information：

```
      pragma solidity ^0.8.0;

import "../utils/Context.sol";

/**
 * @dev Contract module which provides a basic access control mechanism, where
 * there is an account (an owner) that can be granted exclusive access to
 * specific functions.
 *
 * By default, the owner account will be the one that deploys the contract. This
 * can later be changed with {transferOwnership}.
 *
 * This module is used through inheritance. It will make available the modifier
 * `onlyOwner`, which can be applied to your functions to restrict their use to
 * the owner.
 */
abstract contract Ownable is Context {
    address private _owner;

    event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);

    /**
     * @dev Initializes the contract setting the deployer as the initial owner.
     */
    constructor() {
        _transferOwnership(_msgSender());    //Constructor to initialize the owner of the contract
    }

    /**
     * @dev Throws if called by any account other than the owner.
     */
    modifier onlyOwner() { //Modifier, which can only be called by the owner of the contract
        _checkOwner(); //Check whether the caller is the owner of the contract
        _;
    }

    /**
     * @dev Returns the address of the current owner.
     */
    function owner() public view virtual returns (address) {    //Retrieve the owner address of the contract
        return _owner;
    }

    /**
     * @dev Throws if the sender is not the owner.
     */
    function _checkOwner() internal view virtual { //Only internal calls are allowed
        require(owner() == _msgSender(), "Ownable: caller is not the owner"); //Check whether the address is the owner address
    }

    /**
     * @dev Leaves the contract without owner. It will not be possible to call
     * `onlyOwner` functions anymore. Can only be called by the current owner.
     *
     * NOTE: Renouncing ownership will leave the contract without an owner,
     * thereby removing any functionality that is only available to the owner.
     */
    function renounceOwnership() public virtual onlyOwner { //Only the owner of the contract is allowed to call, which is used to give up the owner's permission
        _transferOwnership(address(0)); //The owner address of the contract is assigned to zero, that is, the owner authority is abandoned
    }

    /**
     * @dev Transfers ownership of the contract to a new account (`newOwner`).
     * Can only be called by the current owner.
     */
    function transferOwnership(address newOwner) public virtual onlyOwner { //Only contract owner
```

```
calls are allowed
            require(newOwner != address(0), "Ownable: new owner is the zero address"); //Address
non-zero check
            _transferOwnership(newOwner);    //Update the owner of the contract
      }

      /**
       * @dev Transfers ownership of the contract to a new account (`newOwner`).
       * Internal function without access restriction.
       */
      function _transferOwnership(address newOwner) internal virtual {    //Only contract internal calls
are allowed
            address oldOwner = _owner;    //Staging the old owner
            _owner = newOwner;    //Update owner
            emit OwnershipTransferred(oldOwner, newOwner);
      }

}
```

Safety advice： NONE.

8.3.4. AirdropNoParent logic design【Low risk】

Audit description： Conduct security audit on the airdropNoParent parentless airdrop business logic design in the contract to check whether zero address is checked and whether the business logic design is reasonable.

Audit results： The airdropNoParent in the contract checks the zero address, but during the loop traversal, it only checks whether the length of the two incoming arrays is consistent, but does not limit the length. Due to the upper limit of gas, there is a risk of Self DOS. Since the modified function can only be called by the owner of the contract, it is evaluated as a low risk.

Code file： Moy.sol 113~134

Code information：

```
    function airdropNoParent(address[] calldata addr, uint256[] calldata amount)
```

```
        external
        onlyOwner
    {   //Only contract owner calls are allowed
        require(addr.length == amount.length, "addrLen!=amountLen");    //The length of two arrays
is checked, but there is no limit. There is a Self DOS risk
        uint24 curPeriod = getCurrentEpoch(); //Get the current Epoch
        for (uint256 i = 0; i < addr.length; ++i) { //Loop through the address for airdrop operation
            address adr = addr[i];
            uint256 a = amount[i];
            require(adr != address(0), "ERC20: mint to the zero address"); //Address non-zero check
            _totalSupply += a; //Total Update Releases
            uint256 curB = balanceOf(adr);    //Obtain the current assets held by the address
            updateAccount(
                curPeriod,
                lastEpochPerAccount[adr],
                adr,
                curB - _balances[adr]
            ); //Update account information
            _balances[adr] = curB + a; //Update the number of assets held at the address
            emit Transfer(address(0), adr, a);
        }

    }
```

Safety advice： NONE.

8.3.5. Logic design of white list address management【security】

Audit description： Audit the logic design related to the white list management (adding addresses to the white list, removing addresses from the white list) in the contract to check whether there is permission control and whether the related logic design is reasonable.

Audit results： The white list management related functions in the contract can only be called by the owner of the contract, and the related business logic design is correct.

Code file： Moy.sol 136~158

Code information：

```
     function addSellWhitelist(address adr) public onlyOwner {    //Only contract owner calls are allowed
          sellWhitelist.set(uint256(uint160(adr))); //Add to sell white list
     }

     function removeSellWhitelist(address adr) public onlyOwner {//Only contract owner calls are allowed
          sellWhitelist.unset(uint256(uint160(adr)));//Remove the self whitelist address
     }

     function getSellWhitelist(address adr) public view returns (bool) {//Get the address of the sell white list
          return sellWhitelist.get(uint256(uint160(adr)));
     }

     function addBuyWhitelist(address adr) public onlyOwner { //Only contract owner calls are allowed
          buyWhitelist.set(uint256(uint160(adr)));    //Add to buy white list
     }

     function removeBuyWhitelist(address adr) public onlyOwner { //Only contract owner calls are allowed
          buyWhitelist.unset(uint256(uint160(adr)));    //Remove buy whitelist address
     }

     function getBuyWhitelist(address adr) public view returns (bool) { //Get buy white list address
          return buyWhitelist.get(uint256(uint160(adr)));
          }
```

Safety advice： NONE.

8.3.6. Logic Design of Business Address Updating【security】

Audit description： Carry out security audit on the design of the business address update logic in the contract to check whether the authority of the function caller is checked and whether there is a risk of shaping overflow.

Audit results: The business address update function in the contract can only be called by the owner of the contract, and the relevant business logic design is correct.

Code file: Moy.sol 160~182

Code information:

```
    function setSellExiAddress(address adr) external onlyOwner {   //Only contract owner calls are allowed
        sellExiAddress = adr; //Set EXI address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellEaiAddress(address adr) external onlyOwner {//Only contract owner calls are allowed
        sellEaiAddress = adr; //Set EAI address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellTreaturyAddress(address adr) external onlyOwner {//Only contract owner calls are allowed
        sellTreaturyAddress = adr; //Set the sell Feature address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setbuyLeaderAddress(address adr) external onlyOwner {//Only contract owner calls are allowed
        buyLeaderAddress = adr; //Set buy leader address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);

    }
```

Safety advice: NONE.

8.3.7. Contract basic information retrieval logic【security】

Audit description: Carry out security audit on the design of the business address update logic in the contract to check whether the authority of the function caller is checked and whether there is a risk of shaping overflow.

Audit results： The business address update function in the contract can only be called by the owner of the contract, and the relevant business logic design is correct.

Code file： Moy.sol 184~203

Code information：

```
  function name() public view virtual override returns (string memory) { //Retrieve contract name information
        return _name;
    }

    function symbol() public view virtual override returns (string memory) { //Retrieving contract symbol information
        return _symbol;
    }

    function decimals() public view virtual override returns (uint8) { //Accuracy of contract retrieval
        return 6;
    }

    function totalSupply() public view virtual override returns (uint256) { //Retrieve total contract issuance
        uint256 total = _totalSupply; //Get the current total number of issues
        for (uint256 i = 0; i < lpAccounts.length; ++i) {
            address account = lpAccounts[i];
            total += calAccountInterest(account, _balances[account], 0); //Calculate the account interest and add the total amount of tokens
        }
        return total;
    }
    function balanceOf(address account)
        public
        view
        virtual
        override
        returns (uint256)
    {
        uint256 balance = _balances[account]; //Calculate the number of assets held by the account
        return balance + calAccountInterest(account, balance, 0);    //Number of assets held+interest

    }
```

Safety advice： NONE.

8.3.8. CalAccountInterest logic【security】

Audit description： Carry out security audit on the logic design of the account interest calculation business in the contract to check whether the zero address is checked and whether the relevant business logic design is reasonable.

Audit results：The logic design of the account interest calculation business in the contract is reasonable and correct, and no relevant problems are found.

Code file： Moy.sol 243~283

Code information：

```
function calAccountInterest(
        address account,
        uint256 balance,
        uint256 removedLp
    ) private view returns (uint256) {//Divide the output interest rate income according to the amount
of liquidity added by users
        uint24 i = lastEpochPerAccount[account] + 1;
        uint256 curEpoch = getCurrentEpoch(); //Get the current Epoch
        if (!canInterest(account) || i > curEpoch) { //Check whether there is interest available
            return 0;
        }
        uint24 smallEpoch = i;
        if (lpPerEpoch[smallEpoch] == 0) {
            return 0;
        }
        uint256 lastEpochLp = lpPerEpoch[smallEpoch];
        uint256 lastEpochMoy = moyPerEpoch[smallEpoch];
        uint256 interest;
        uint256 liquidity = liquidityPerAccount[account]; //LP Search
        uint256 lpBalance = IPancakePair(pair).balanceOf(account) + removedLp; //Calculate the
number of LP assets
        uint256 effectLp = Math.min(liquidity, lpBalance); //Calculate valid LP
        uint256 moy;
        for (; i <= curEpoch; ++i) {
            uint256 iLp = lpPerEpoch[i];
            if (iLp == 0) { //LP 为 0
                moy = (effectLp * lastEpochMoy) / lastEpochLp;
            } else { //LP  不为 0
                lastEpochMoy = moyPerEpoch[i];
                lastEpochLp = lpPerEpoch[i];
                moy = (effectLp * lastEpochMoy) / lastEpochLp;
            }
            if (balance > moy) {
                return 0;
            }
            interest += calc(moy); //Calculate interest
            if (balance + interest > moy) {
                interest = moy - balance;
                break;
            }
        }
        return interest;
    }
    function canInterest(address adr) private view returns (bool) {
        if (adr.isContract() || adr == address(0)) { //If the address is a contract address or zero address,
false is returned directly
            return false;
        }
        return true;
    }
    function calc(uint256 amount) private pure returns (uint256) { //Calculation of interest rate
income
        uint256 rate;
        if (amount <= 1000 * 1e6) {                 // 1~1000 1.0%
            rate = RATE1;
        } else if (amount <= 3000 * 1e6) {      // 1001~3000 1.1%
            rate = RATE2;
        } else if (amount <= 10000 * 1e6) {    //3001~10000    1.2%
            rate = RATE3;
        } else if (amount <= 30000 * 1e6) {    // 10001~30000    1.4 %
            rate = RATE4;
        } else {
            rate = RATE5;                      // 1.7%
        }
        return (amount * rate) / 1000;

    }
```

Safety advice：：NONE.

8.3.9. Logic design of transfer business【security】

Audit description： Conduct security audit on the transfer business logic design in the contract to check whether zero address is checked and whether the relevant business logic design is reasonable.

Audit results： The transfer business logic design in the contract is reasonable and correct, and no relevant problems are found.

Code file： Moy.sol 294~303

Code information：

```
function transfer(address to, uint256 amount)
        public
        virtual
        override
        returns (bool)
    {
        address owner = _msgSender();    //Caller of retrieval function
        _transfer(owner, to, amount);    //Call_ Transfer
        return true;
    }
    function _transfer(
        address from,
        address to,
        uint256 amount
    ) internal virtual {
        require(from != address(0), "ERC20: transfer from the zero address");    //Address non-zero
check
        require(to != address(0), "ERC20: transfer to the zero address");    //Address non-zero check

        uint256 tranType = 0; //Transaction type, default to direct transfer between wallets
        uint256 changedLp;    //LP update
        uint256 feeToLp;
        if (to == pair) { //Token acceptance address is pair address
```

```
            (uint112 r0, uint112 r1, ) = IPancakePair(pair).getReserves(); //Obtain the value of each
asset of constant products in the transaction pair
            uint256 amountA;
            if (r0 > 0 && r1 > 0) {
                amountA = IPancakeRouter(ROUTER_ADDRESS).quote(amount, r1, r0);
//Calculate the value of one asset from another according to the proportion
            }
            uint256 balanceA = IERC20(USDT_ADDRESS).balanceOf(pair); //Obtain the quantity
of USDT
            if (balanceA < r0 + amountA) {
                tranType = 1;   //sell
            } else {
                tranType = 2;
                (changedLp, feeToLp) = calLiquidity(balanceA, amount, r0, r1); //Calculate
liquidity
            }
        }
        if (from == pair) {//Token source address is pair address
            (uint112 r0, uint112 r1, ) = IPancakePair(pair).getReserves();//Obtain the value of each
asset of the constant product in the transaction pair
            uint256 amountA;
            if (r0 > 0 && r1 > 0) { //R0 and r1 are both greater than zero
                amountA = IPancakeRouter(ROUTER_ADDRESS).getAmountIn(
                    amount,
                    r0,
                    r1
                ); //Calculate how many other assets need to be provided in exchange for a certain
amount of certain assets in the transaction pair
            }
            uint256 balanceA = IERC20(USDT_ADDRESS).balanceOf(pair); //Number of assets
acquired
            if (balanceA > r0 + amountA) {
                tranType = 3; //SELL
            } else {
                tranType = 4;
            }
        }

        uint24 currentEpoch = getCurrentEpoch(); //Get the current Epoch

        uint256 oldBalance = balanceOf(from); //Get the number of assets held by the from address
        require(oldBalance >= amount, "ERC20: transfer amount exceeds balance"); //Check
whether the transfer amount exceeds the balance
        updateAccount(
            currentEpoch,
            lastEpochPerAccount[from],
            from,
            oldBalance - _balances[from]
        ); //Update the number of assets held in the account
        unchecked {
            _balances[from] = oldBalance - amount;
        } //Update asset quantity

        uint256 subAmount;
        if (tranType == 0) {    //Transfer between wallets
            subAmount = (amount * 15) / 100;                        //15% of the transaction is destroyed
directly
            _balances[address(0)] += subAmount;
            emit Transfer(from, address(0), subAmount);
        } else if (tranType == 1) {    //卖
            if (!sellWhitelist.get(uint256(uint160(from)))) { //Check whether the from address is a
whitelist address
                uint256 amount50 = (amount * 50) / 1000;    //Transactional 5%
                uint256 curB = balanceOf(sellExiAddress); //Get the assets held by the current
sellExiAddress
                updateAccount(
                    currentEpoch,
                    lastEpochPerAccount[sellExiAddress],
                    sellExiAddress,
                    curB - _balances[sellExiAddress]
                ); //Update assets held in the account
                _balances[sellExiAddress] = curB + amount50; //buy-back EXI
                emit Fee(from, sellExiAddress, amount50);
                curB = balanceOf(sellEaiAddress);            //Get the assets held by the current
sellEaiAddress
                updateAccount(
                    currentEpoch,
                    lastEpochPerAccount[sellEaiAddress],
                    sellEaiAddress,
                    curB - _balances[sellEaiAddress]
                ); //Update assets held in the account
                _balances[sellEaiAddress] = curB + amount50; //buy-back EAI
                emit Fee(from, sellEaiAddress, amount50);
                uint256 amount25 = (amount * 25) / 1000;    //Transactional 2.5%
                _balances[address(0)] += amount25;              //2.5% is directly destroyed, and the
address of the black hole is penetrated
                emit Fee(from, address(0), amount25);
                curB = balanceOf(sellTreaturyAddress);        //Number of ecological treasury assets
acquired
                    updateAccount(
                          currentEpoch,
                          lastEpochPerAccount[sellTreaturyAddress],
                          sellTreaturyAddress,
                          curB - _balances[sellTreaturyAddress]
                    ); //Update asset quantity
                    _balances[sellTreaturyAddress] = curB + amount25; //2.5% Address of ecological
treasury
                    emit Fee(from, sellTreaturyAddress, amount25);
                    subAmount = amount50 + amount50 + amount25 + amount25; //Calculate the
number of assets consumed in the process
                  }
        } else if (tranType == 2) {
                  liquidityPerAccount[from] += changedLp;    //update LP
                  if (!hasLp.get(uint256(uint160(from)))) { //Check whether the from address holds LP
                    lpAccounts.push(from); //If there is no LP, join the LP holder queue
                    hasLp.set(uint256(uint160(from))); //Set from holding LP
                  }
        } else if (tranType == 3) { //buy
                  if (!buyWhitelist.get(uint256(uint160(to)))) { //Check whether the address to is in the
white list
                    uint256 amount50 = (amount * 50) / 1000;    //Of transaction amount 5%
                    uint256 curB = balanceOf(buyLeaderAddress); //Get current assets
                    updateAccount(
                          currentEpoch,
                          lastEpochPerAccount[buyLeaderAddress],
                          buyLeaderAddress,
                          curB - _balances[buyLeaderAddress]
                    ); //Update asset quantity
                    _balances[buyLeaderAddress] = curB + amount50;    //Put 5% into the leaders'
treasury
                    emit Fee(from, buyLeaderAddress, amount50);
                    marketReward(from, to, currentEpoch, amount, amount50); //Give 5% to reward
the recommendation of superiors and subordinates
                    _balances[address(0)] += amount50; //Take 5% and destroy it directly
                    emit Fee(from, address(0), amount50);
                    subAmount = amount50 + amount50 + amount50; //Calculate the number of assets
consumed in the process
                  }
                  buymoyPerAccount[to] += amount - subAmount; //Update the number of assets in the
post purchase address account
        } else if (tranType == 4) {
                  changedLp =
                ((amount + 1) * IPancakePair(pair).totalSupply()) /
                (balanceOf(pair) - 1);
            uint256 currB = balanceOf(to, changedLp); //Calculate the current asset quantity
            updateAccount(
                currentEpoch,
                lastEpochPerAccount[to],
                to,
                currB - _balances[to]
            ); //Update assets
            _balances[to] = currB;
            unchecked {
                liquidityPerAccount[to] -= changedLp;
            } //Update LP
        }

        uint256 toAmount = amount - subAmount; //Calculate the actual transfer quantity
        oldBalance = balanceOf(to);//Retrieve current asset quantity
        updateAccount(
            currentEpoch,
            lastEpochPerAccount[to],
            to,
            oldBalance - _balances[to]
        ); //Update asset quantity
        _balances[to] = oldBalance + toAmount; //Update the latest number of tokens held in the
current account

        uint256 lpTotal = IPancakePair(pair).totalSupply(); //Get total LP
        if (tranType == 2) {
            if (lpTotal == 0) {
                lpPerEpoch[currentEpoch + 1] = lpTotal + changedLp + 1000;
            } else {
                lpPerEpoch[currentEpoch + 1] = lpTotal + changedLp + feeToLp;
            }
        } else {
            lpPerEpoch[currentEpoch + 1] = lpTotal;
        }
        moyPerEpoch[currentEpoch + 1] = balanceOf(pair);
        emit Transfer(from, to, toAmount);
    }
    function calLiquidity(
        uint256 balanceA,
        uint256 amount,
        uint112 r0,
        uint112 r1
    ) private view returns (uint256 liquidity, uint256 feeToLiquidity) { //calculation Liquidity
        uint256 pairTotalSupply = IPancakePair(pair).totalSupply(); //Get Total
        address feeTo = IPancakeFactory(
            IPancakeRouter(ROUTER_ADDRESS).factory()
        ).feeTo(); //计算 fee
        bool feeOn = feeTo != address(0); //Check whether the fee switch is on. If the fee address is
not zero, it is considered to be on
        uint256 _kLast = IPancakePair(pair).kLast(); //Get the value of the product in the constant
product
        if (feeOn) { //Start charging
            if (_kLast != 0) { //Judge current_ Whether klast is 0 or not, if not, it means that the
corresponding transaction pair tokens have been initialized and injected into the pool, that is, there is
liquidity
                uint256 rootK = Math.sqrt(uint256(r0).mul(r1));    //Take the quadratic root of K
                uint256 rootKLast = Math.sqrt(_kLast); //Take the quadratic root of K
                if (rootK > rootKLast) { //If the current K is greater than the previous K
                    uint256 numerator = pairTotalSupply
                        .mul(rootK.sub(rootKLast))
                        .mul(8);    //Computational Molecule
                    uint256  denominator  =  rootK.mul(17).add(rootKLast.mul(8));  //Calculate
denominator
                    feeToLiquidity = numerator / denominator;
                    if (feeToLiquidity > 0) pairTotalSupply += feeToLiquidity;
                }
            }
        }
        uint256 amount0 = balanceA - r0;
        if (pairTotalSupply == 0) { //If the total amount is zero, the pool is not initialized
            liquidity = Math.sqrt(amount0 * amount) - 1000; //Calculate liquidity
        } else {
            liquidity = Math.min(
                (amount0 * pairTotalSupply) / r0,
                (amount * pairTotalSupply) / r1
            );
        }
    }

    function marketReward(
        address from,
        address to,
        uint24 currentEpoch,
        uint256 amount,
        uint256 restAmount
    ) private {    //5% will be used as reward for recommendation of superiors and subordinates, and
```

```
only ten generations of superior members will be rewarded
        address p = parents[to];    //Get the upper level address corresponding to the to address
        for (
            uint256 i = 1;
            i <= 10 && p != address(0) && p != address(1);
            ++i
        ) { //Loop up to retrieve the upper level of the 10th generation
            uint256 pAmount;
            if (i == 1) {
                pAmount = (amount * 9) / 1000;                    //0.95
            } else if (i == 2) {
                pAmount = (amount * 7) / 1000;                   //0.7%
            } else if (i == 3) {
                pAmount = (amount * 5) / 1000;                  //0.5
            } else if (i == 4) {
                pAmount = (amount * 3) / 1000;                 //0.3%
            } else if (i == 5) {
                pAmount = amount / 1000;                         //0.1%
            } else if (i == 6) {
                pAmount = amount / 1000;                        //0.1%
            } else if (i == 7) {
                pAmount = (amount * 3) / 1000;           //0.3%
            } else if (i == 8) {
                pAmount = (amount * 5) / 1000;           //0.5%
            } else if (i == 9) {
                pAmount = (amount * 7) / 1000;          //0.7%
            } else {
                pAmount = restAmount;
            }
            uint256 curB = balanceOf(p); //Get the asset quantity of the current superior member
            updateAccount(
                currentEpoch,
                lastEpochPerAccount[p],
                p,
                curB - _balances[p]
            ); //更新账户
            _balances[p] = curB + pAmount; //Update the number of member assets
            feePerAccount[p] += pAmount; //fee
            emit Fee(from, p, pAmount);
            restAmount -= pAmount; //Update remaining available reward assets
            p = parents[p]; //Update Parent
        }
        if (restAmount > 0) { //If there is a surplus, there is not enough series
            _balances[address(0)] += restAmount; //Without enough series, other wave ratios hit the
black hole address
            emit Fee(from, address(0), restAmount);
        }

}
```

Safety advice：：NONE.

8.3.10. Business logic related to authorized transfer【security】

Audit description：Conduct security audit on the design of business logic related to authorized transfer in the contract, check whether zero address is checked, and whether the design of relevant business logic is reasonable.

Audit results： The business logic design related to authorized transfer in the contract is reasonable and correct, and no relevant problems are found.

Code file： Moy.sol 305~324 337~364

Code information：

```
function allowance(address owner, address spender)
            public
            view
            virtual
            override
            returns (uint256)
        {
            return _allowances[owner][spender];    //Retrieve Authorization Limit
        }
        function approve(address spender, uint256 amount)
            public
            virtual
            override
            returns (bool)
        {
            address owner = _msgSender();    //Retrieve the caller of the current function
            _approve(owner, spender, amount);    //Update authorization limit
            return true;
        }
        function _approve(
            address owner,
            address spender,
            uint256 amount
        ) internal virtual {
            require(owner != address(0), "ERC20: approve from the zero address");    //Address non-zero check
            require(spender != address(0), "ERC20: approve to the zero address");    //Address non-zero check

            _allowances[owner][spender] = amount;    //Update authorization limit
            emit Approval(owner, spender, amount);
        }
        function increaseAllowance(address spender, uint256 addedValue)
            public
            virtual
            returns (bool)
        {
            address owner = _msgSender(); //Retrieve the caller of the current function
            _approve(owner, spender, allowance(owner, spender) + addedValue);    //Update authorization limit (increase)
            return true;
        }

        function decreaseAllowance(address spender, uint256 subtractedValue)
            public
            virtual
            returns (bool)
        {
            address owner = _msgSender();//Retrieve the caller of the current function
            uint256 currentAllowance = allowance(owner, spender); //Check the current authorization limit
            require(
                currentAllowance >= subtractedValue,
                "ERC20: decreased allowance below zero"
            ); //Check whether the current authorization limit is greater than the limit to be reduced
            unchecked {
                _approve(owner, spender, currentAllowance - subtractedValue);//Update authorization limit (decrease)
            }

            return true;
        }
```

Safety advice： NONE.

8.3.11. TransferFrom transfer logic design【security】

Audit description: Conduct security audit on the design of business logic related to authorized transfer in the contract, check whether zero address is checked, and whether the design of relevant business logic is reasonable.

Audit results: The business logic design related to authorized transfer in the contract is reasonable and correct, and no relevant problems are found.

Code file: Moy.sol 326~335

Code information:

```
function transferFrom(
        address from,
        address to,
        uint256 amount
    ) public virtual override returns (bool) {
        address spender = _msgSender();
        _spendAllowance(from, spender, amount); //Update authorization limit
        _transfer(from, to, amount); //Call_ Transfer
        return true;
    }
    function _spendAllowance(
        address owner,
        address spender,
        uint256 amount
    ) internal virtual {
        uint256 currentAllowance = allowance(owner, spender); //Get the current authorization limit
        if (currentAllowance != type(uint256).max) {
            require(
                currentAllowance >= amount,
                "ERC20: insufficient allowance"
            ); //Check whether the current authorization limit is greater than the quantity to be
transferred
            unchecked {
                _approve(owner, spender, currentAllowance - amount); //Update authorization
limit
            }
        }
    }
    function _transfer(
        address from,
        address to,
        uint256 amount
    ) internal virtual {
        require(from != address(0), "ERC20: transfer from the zero address");    //Address non-zero
check
        require(to != address(0), "ERC20: transfer to the zero address");    //Address non-zero check

        uint256 tranType = 0; //Transaction type, default to direct transfer between wallets
        uint256 changedLp;    //LP update
        uint256 feeToLp;
        if (to == pair) { //Token acceptance address is pair address
            (uint112 r0, uint112 r1, ) = IPancakePair(pair).getReserves(); //Obtain the value of each
asset of the constant product in the transaction pair
            uint256 amountA;
            if (r0 > 0 && r1 > 0) {
                amountA = IPancakeRouter(ROUTER_ADDRESS).quote(amount, r1, r0);
//Calculate the value of one asset from another according to the proportion
            }
            uint256 balanceA = IERC20(USDT_ADDRESS).balanceOf(pair); //Obtain the quantity
of USDT
            if (balanceA < r0 + amountA) {
                tranType = 1;    //sell
            } else {
                tranType = 2;
                (changedLp, feeToLp) = calLiquidity(balanceA, amount, r0, r1); //Calculate
liquidity
            }
        }
        if (from == pair) {//Token source address is pair address
            (uint112 r0, uint112 r1, ) = IPancakePair(pair).getReserves();//Obtain the value of each
asset of the constant product in the transaction pair
            uint256 amountA;
            if (r0 > 0 && r1 > 0) { //R0 and r1 are both greater than zero
                amountA = IPancakeRouter(ROUTER_ADDRESS).getAmountIn(
                    amount,
                    r0,
                    r1
                ); //Calculate how many other assets need to be provided in exchange for a certain
amount of certain assets in the transaction pair
            }
            uint256 balanceA = IERC20(USDT_ADDRESS).balanceOf(pair); //Number of assets
acquired
```

```
                if (balanceA > r0 + amountA) {
                    tranType = 3; //buy
                } else {
                    tranType = 4;
                }
            }

            uint24 currentEpoch = getCurrentEpoch(); //Get the current Epoch

            uint256 oldBalance = balanceOf(from); //Get the number of assets held by the from address
            require(oldBalance >= amount, "ERC20: transfer amount exceeds balance"); //Check
whether the transfer amount exceeds the balance
            updateAccount(
                currentEpoch,
                lastEpochPerAccount[from],
                from,
                oldBalance - _balances[from]
            ); //Update the number of assets held in the account
            unchecked {
                _balances[from] = oldBalance - amount;
            } //Update asset quantity

            uint256 subAmount;
            if (tranType == 0) {    //Transfer between wallets
                subAmount = (amount * 15) / 100;                //15% of the transaction is destroyed
directly
                _balances[address(0)] += subAmount;
                emit Transfer(from, address(0), subAmount);
            } else if (tranType == 1) {    //sell
                if (!sellWhitelist.get(uint256(uint160(from)))) { //Check whether the from address is a
whitelist address
                    uint256 amount50 = (amount * 50) / 1000;    //transaction 5%
                    uint256 curB = balanceOf(sellExiAddress); //Get the assets held by the current
sellExiAddress
                    updateAccount(
                        currentEpoch,
                        lastEpochPerAccount[sellExiAddress],
                        sellExiAddress,
                        curB - _balances[sellExiAddress]
                    ); //Update assets held in the account
                    _balances[sellExiAddress] = curB + amount50; //buy-back EXI
                    emit Fee(from, sellExiAddress, amount50);
                    curB = balanceOf(sellEaiAddress);          //Get the assets held by the current
sellEaiAddress
```

```
                    updateAccount(
                        currentEpoch,
                        lastEpochPerAccount[sellEaiAddress],
                        sellEaiAddress,
                        curB - _balances[sellEaiAddress]
                    ); //Update assets held in the account
                    _balances[sellEaiAddress] = curB + amount50; //Repo EAI
                    emit Fee(from, sellEaiAddress, amount50);
                    uint256 amount25 = (amount * 25) / 1000;    //transaction2.5%
                    _balances[address(0)] += amount25;               //2.5%Destroy it directly and enter
the address of the black hole
                    emit Fee(from, address(0), amount25);
                    curB = balanceOf(sellTreaturyAddress);       //Number of ecological treasury assets
acquired
                    updateAccount(
                        currentEpoch,
                        lastEpochPerAccount[sellTreaturyAddress],
                        sellTreaturyAddress,
                        curB - _balances[sellTreaturyAddress]
                    ); //Update asset quantity
                    _balances[sellTreaturyAddress] = curB + amount25; //2.5%Address of ecological
treasury
                    emit Fee(from, sellTreaturyAddress, amount25);
                    subAmount = amount50 + amount50 + amount25 + amount25; //Calculate the
number of assets consumed in the process
                }
            } else if (tranType == 2) {
                liquidityPerAccount[from] += changedLp;    //update LP
                if (!hasLp.get(uint256(uint160(from)))) { //Check whether the from address holds LP
                    lpAccounts.push(from); //If there is no LP, join the LP holder queue
                    hasLp.set(uint256(uint160(from))); //Set from holding LP
                }
            } else if (tranType == 3) { //buy
                if (!buyWhitelist.get(uint256(uint160(to)))) { //Check whether the address to is in the
white list
                    uint256 amount50 = (amount * 50) / 1000;    //Transaction amount5%
                    uint256 curB = balanceOf(buyLeaderAddress); //Get current assets
                    updateAccount(
                        currentEpoch,
                        lastEpochPerAccount[buyLeaderAddress],
                        buyLeaderAddress,
                        curB - _balances[buyLeaderAddress]
                    ); //Update asset quantity
                    _balances[buyLeaderAddress] = curB + amount50;    //Put 5% into the leaders'
```

```
treasury
                    emit Fee(from, buyLeaderAddress, amount50);
                    marketReward(from, to, currentEpoch, amount, amount50); //Give 5% to reward
the recommendation of superiors and subordinates
                    _balances[address(0)] += amount50; //Take 5% and destroy it directly
                    emit Fee(from, address(0), amount50);
                    subAmount = amount50 + amount50 + amount50; //Calculate the number of assets
consumed in the process
                }
                buymoyPerAccount[to] += amount - subAmount; //Update the number of assets in the
post purchase address account
            } else if (tranType == 4) {
                changedLp =
                    ((amount + 1) * IPancakePair(pair).totalSupply()) /
                    (balanceOf(pair) - 1);
                uint256 currB = balanceOf(to, changedLp); //Calculate the current asset quantity
                updateAccount(
                    currentEpoch,
                    lastEpochPerAccount[to],
                    to,
                    currB - _balances[to]
                ); //Update assets
                _balances[to] = currB;
                unchecked {
                    liquidityPerAccount[to] -= changedLp;
                } //Update LP
            }

            uint256 toAmount = amount - subAmount; //Calculate the actual transfer quantity
            oldBalance = balanceOf(to);//Retrieve current asset quantity
            updateAccount(
                currentEpoch,
                lastEpochPerAccount[to],
                to,
                oldBalance - _balances[to]
            ); //Update asset quantity
            _balances[to] = oldBalance + toAmount; //Update the latest number of tokens held in the
current account

            uint256 lpTotal = IPancakePair(pair).totalSupply(); //Get total LP
            if (tranType == 2) {
                if (lpTotal == 0) {
                    lpPerEpoch[currentEpoch + 1] = lpTotal + changedLp + 1000;
                } else {
```

```
                lpPerEpoch[currentEpoch + 1] = lpTotal + changedLp + feeToLp;
            }
        } else {
            lpPerEpoch[currentEpoch + 1] = lpTotal;
        }
        moyPerEpoch[currentEpoch + 1] = balanceOf(pair);
        emit Transfer(from, to, toAmount);
    }
    function calLiquidity(
        uint256 balanceA,
        uint256 amount,
        uint112 r0,
        uint112 r1
    ) private view returns (uint256 liquidity, uint256 feeToLiquidity) { //calculation Liquidity
        uint256 pairTotalSupply = IPancakePair(pair).totalSupply(); //Get Total
        address feeTo = IPancakeFactory(
            IPancakeRouter(ROUTER_ADDRESS).factory()
        ).feeTo(); //计算 fee
        bool feeOn = feeTo != address(0); //Check whether the fee switch is on. If the fee address is not zero, it is considered to be on
        uint256 _kLast = IPancakePair(pair).kLast(); //Get the value of the product in the constant product
        if (feeOn) { //Start charging
            if (_kLast != 0) { //Judge current_ Whether klast is 0 or not, if not, it means that the corresponding transaction pair tokens have been initialized and injected into the pool, that is, there is liquidity
                uint256 rootK = Math.sqrt(uint256(r0).mul(r1));    //Take the quadratic root of K
                uint256 rootKLast = Math.sqrt(_kLast); //Take the quadratic root of K
                if (rootK > rootKLast) { //If the current K is greater than the previous K
                    uint256 numerator = pairTotalSupply
                        .mul(rootK.sub(rootKLast))
                        .mul(8);    //Computational Molecule
                    uint256 denominator = rootK.mul(17).add(rootKLast.mul(8));   //Calculate denominator
                    feeToLiquidity = numerator / denominator;
                    if (feeToLiquidity > 0) pairTotalSupply += feeToLiquidity;
                }
            }
        }
        uint256 amount0 = balanceA - r0;
        if (pairTotalSupply == 0) { //If the total amount is zero, the pool is not initialized
            liquidity = Math.sqrt(amount0 * amount) - 1000; //Calculate liquidity
        } else {
            liquidity = Math.min(
                    (amount0 * pairTotalSupply) / r0,
                    (amount * pairTotalSupply) / r1
                );
            }
        }

        function marketReward(
            address from,
            address to,
            uint24 currentEpoch,
            uint256 amount,
            uint256 restAmount
        ) private {    //5% will be used as reward for recommendation of superiors and subordinates, and
only ten generations of superior members will be rewarded
            address p = parents[to];    //Get the upper level address corresponding to the to address
            for (
                uint256 i = 1;
                i <= 10 && p != address(0) && p != address(1);
                ++i
            ) { //Loop up to retrieve the upper level of the 10th generation
                uint256 pAmount;
                if (i == 1) {
                    pAmount = (amount * 9) / 1000;                    //0.95
                } else if (i == 2) {
                    pAmount = (amount * 7) / 1000;                   //0.7%
                } else if (i == 3) {
                    pAmount = (amount * 5) / 1000;                  //0.5
                } else if (i == 4) {
                    pAmount = (amount * 3) / 1000;                 //0.3%
                } else if (i == 5) {
                    pAmount = amount / 1000;                        //0.1%
                } else if (i == 6) {
                    pAmount = amount / 1000;                       //0.1%
                } else if (i == 7) {
                    pAmount = (amount * 3) / 1000;            //0.3%
                } else if (i == 8) {
                    pAmount = (amount * 5) / 1000;            //0.5%
                } else if (i == 9) {
                    pAmount = (amount * 7) / 1000;           //0.7%
                } else {
                    pAmount = restAmount;
                }
                uint256 curB = balanceOf(p); //Get the asset quantity of the current superior member
                updateAccount(
                    currentEpoch,
                    lastEpochPerAccount[p],
                    p,
                    curB - _balances[p]
                ); //Update account
                _balances[p] = curB + pAmount; //Update the number of member assets
                feePerAccount[p] += pAmount; //fee
                emit Fee(from, p, pAmount);
                restAmount -= pAmount; //Update remaining available reward assets
                p = parents[p]; //Update Parent
            }
            if (restAmount > 0) { //If there is a surplus, there is not enough series
                _balances[address(0)] += restAmount; //Without enough series, other wave ratios hit the
black hole address
                emit Fee(from, address(0), restAmount);
            }

        }
```

Safety advice：：NONE.

8.3.12. Bind hierarchical relationship business logic design【security】

Audit description: Conduct security audit on the binding logic design of the Bind hierarchy relationship in the contract to check whether zero addresses are checked and whether the design of relevant business logic is reasonable.

Audit results: The binding business logic design of Bind hierarchy relationship in the contract is reasonable.

Code file: Moy.sol 676~686

Code information:

```
 function bind(bytes6 pCode) external {
            require(parents[msg.sender] == address(0), "already bind"); //Check whether the address is
bound
            address parent = codeToAccount[pCode]; //Search the address account corresponding to the
code
            require(parents[parent] != address(0), "parent invalid"); //Check whether the parent address is
zero
            parents[msg.sender] = parent;    //Set the parent of msg.sender as parent (address account
corresponding to Code)
            addChild(msg.sender, parent);    //Add Child
            bytes6 code = generateCode(msg.sender); //Generate code for msg.sender
            accountToCode[msg.sender] = code; //msg.sender Bind with the corresponding code
            codeToAccount[code] = msg.sender; //Code Bind with msg.sender
            emit Bind(msg.sender, parent);
        }
        function addChild(address user, address p) private {
            for (
                uint256 i = 1;
                i <= 10 && p != address(0) && p != address(1);
                ++i
            ) { //Update sub level information one by one
                children[keccak256(abi.encode(p, i))].push(user); //Set Child
                p = parents[p]; //update P
            }

}
```

Safety advice： NONE.

8.3.13. Logical Design of Hierarchical Relation Retrieval【security】

Audit description： Conduct security audit on the design of the retrieval logic of the hierarchical relationship in the contract, and check whether the design of the relevant business logic is reasonable.

Audit results： The business logic design of contract middle level relationship retrieval is reasonable and correct.

Code file： Moy.sol 699~741

Code information：

```
    function getChildren(address user, uint256 level)
        external
        view
        returns (address[] memory)
    {
        return children[keccak256(abi.encode(user, level))];    //Retrieves the children of the
specified number of levels
```

```
    }

    function getChildrenLength(address user, uint256 level)
        external
        view
        returns (uint256)
    {
        return children[keccak256(abi.encode(user, level))].length;    //Retrieve the number of children of the specified level
    }

    function getChildrenLength(address user) external view returns (uint256) { //Retrieve the number of valid children
        uint256 len;
        for (uint256 i = 1; i <= 10; ++i) {
            len += children[keccak256(abi.encode(user, i))].length;
        }
        return len;
    }

    function getChildren(
        address user,
        uint256 level,
        uint256 pageIndex,
        uint256 pageSize
    ) external view returns (address[] memory) {
        bytes32 key = keccak256(abi.encode(user, level));
        uint256 len = children[key].length;
        address[] memory list = new address[](
            pageIndex * pageSize <= len
                ? pageSize
                : len - (pageIndex - 1) * pageSize
        );
        uint256 start = (pageIndex - 1) * pageSize;
        for (uint256 i = start; i < start + list.length; ++i) {
            list[i - start] = children[key][i];
        }
        return list; //Return all children

    }
```

Safety advice：NONE.

8.3.14. Business logic design of getMoyFee【security】

Audit description： Conduct security audit on the design of the retrieval logic of the hierarchical relationship in the contract, and check whether the design of the relevant business logic is reasonable.

Audit results： The business logic design of contract middle level relationship retrieval is reasonable and correct.

Code file： Moy.sol 743~763

Code information：

```
    function getMoyFee(address[] calldata addr)
        external
        view
        returns (uint256[3][] memory r)
    {
        uint256 lp = IPancakePair(pair).totalSupply();    //Retrieve LP Total
        uint256 moy = balanceOf(pair);    //Retrieve the total amount of moy
        r = new uint256[3][](addr.length);
        for (uint256 i = 0; i < addr.length; ++i) {
            uint256 lpBalance = IPancakePair(pair).balanceOf(addr[i]); //Retrieve the LP quantity
corresponding to the address
            uint256 effectLp = Math.min(
                liquidityPerAccount[addr[i]],
                lpBalance
            ); //Get valid LP
            r[i] = [
                (effectLp * moy) / lp,
                feePerAccount[addr[i]],
                buymoyPerAccount[addr[i]]
            ]; //Calculate Fee
        }
    }
```

Safety advice：NONE.

8.3.15. Design of generateCode business logic 【Low risk】

Audit description： Conduct security audit on the code generation of generateCode function in the contract, and check whether the function business logic design is reasonable.

Audit results： The generateCode function in the contract uses keccak256 to generate a random number when generating a code, but the random number parameter uses block Number leads to predictability. Since the random number here is only a code used to identify hierarchical relationships, and does not involve guessing, it is assessed as a low risk .

Code file： Moy.sol 765~782

Code information：

```
    function generateCode(address adr) public view returns (bytes6) {
        for (uint8 i = 0; i < 256; ++i) {
            bytes6 c6 = bytes6(
                keccak256(abi.encodePacked(adr, i, block.number))
            ); //Use keccak256 to generate random numbers. Here, block.number is used to make it
predictable
            bytes memory c = new bytes(6);
            for (uint8 j = 0; j < 6; ++j) {
                uint8 t = uint8(c6[j]) % 36;
                c[j] = bytes1(t < 26 ? t + 65 : t + 22);
            }
            bytes6 rc = bytes6(c);
            if (codeToAccount[rc] == address(0)) { //If the address is zero, return rc directly
                return rc;
            }
        }
        revert("generate code error"); //If no rc meeting the L776 condition is found after traversal,
code generation fails
    }
```

Safety advice：The generating method of optimized random number.

8.3.16. Contract authority concentration detection【Moderate risk】

Audit description： Detect the degree of authority concentration in the contract and check whether the relevant business logic is reasonable.

Audit results： There are owner operations related to the contract, such as adding a white list address, removing a white list address, setting an EXI sell address, and setting an EAI sell address.

Code file： Moy.sol 136~142 148~154

Code information：

```
  function addSellWhitelist(address adr) public onlyOwner {    //Only contract owner calls are allowed
        sellWhitelist.set(uint256(uint160(adr))); //Add to sell white list
    }

    function removeSellWhitelist(address adr) public onlyOwner {//Only contract owner calls are allowed
        sellWhitelist.unset(uint256(uint160(adr)));//Remove the self whitelist address
    }

    function addBuyWhitelist(address adr) public onlyOwner { //Only contract owner calls are allowed
        buyWhitelist.set(uint256(uint160(adr)));   //Add to buy white list
    }

    function removeBuyWhitelist(address adr) public onlyOwner { //Only contract owner calls are allowed
        buyWhitelist.unset(uint256(uint160(adr)));   //Remove buy whitelist address
    }
    function setSellExiAddress(address adr) external onlyOwner {   //Only contract owner calls are allowed
        sellExiAddress = adr; //Set the sell EXI address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }
    function setSellEaiAddress(address adr) external onlyOwner {//Only contract owner calls are allowed
        sellEaiAddress = adr; //Set sell EAI address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setSellTreaturyAddress(address adr) external onlyOwner {//Only contract owner calls are allowed
        sellTreaturyAddress = adr; //Set the sell Feature address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);
    }

    function setbuyLeaderAddress(address adr) external onlyOwner {//Only contract owner calls are allowed
        buyLeaderAddress = adr; //Set buy leader address
        addBuyWhitelist(adr);
        addSellWhitelist(adr);

    }
```

Safety advice： Properly keep the high authority account and adjust some high authority codes.

9. Official website penetration

Carry out security tests on the MOY official website for wallet detection, sign in, voting, DAO data display and other interactive processes to check whether there are business logic vulnerabilities and common Web security vulnerabilities

Official website address： https://www.moytoken.com/

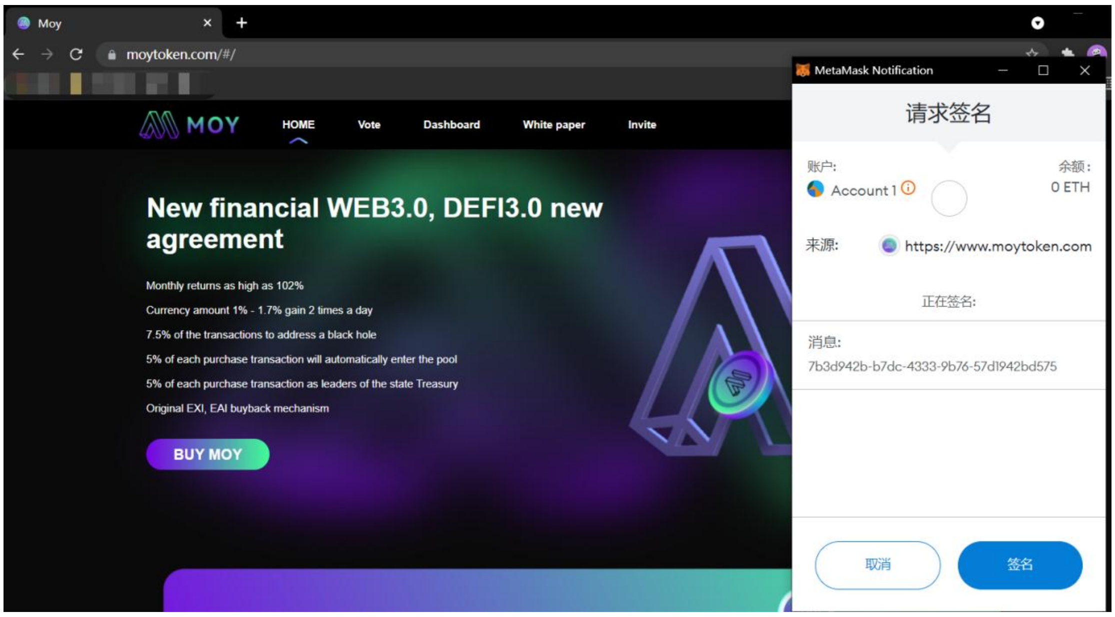
9.1.1. Sign in inspection【security】

Test description： Carry out a security test for the automatic detection of wallet on the official website to check whether there are security risks in the process of automatic recognition of wallet and automatic triggering of signature verification login

Test result： The official website will automatically detect the wallet plug-in and automatically trigger the sign in process。

Test process：

Step 1： Use the MeatMask wallet plug-in as a test. After installing the plug-in and registering for login, accessing the official website address will automatically trigger the signature login



moytoken.com/#/

New financial WEB3.0, DEFI3.0 new agreement

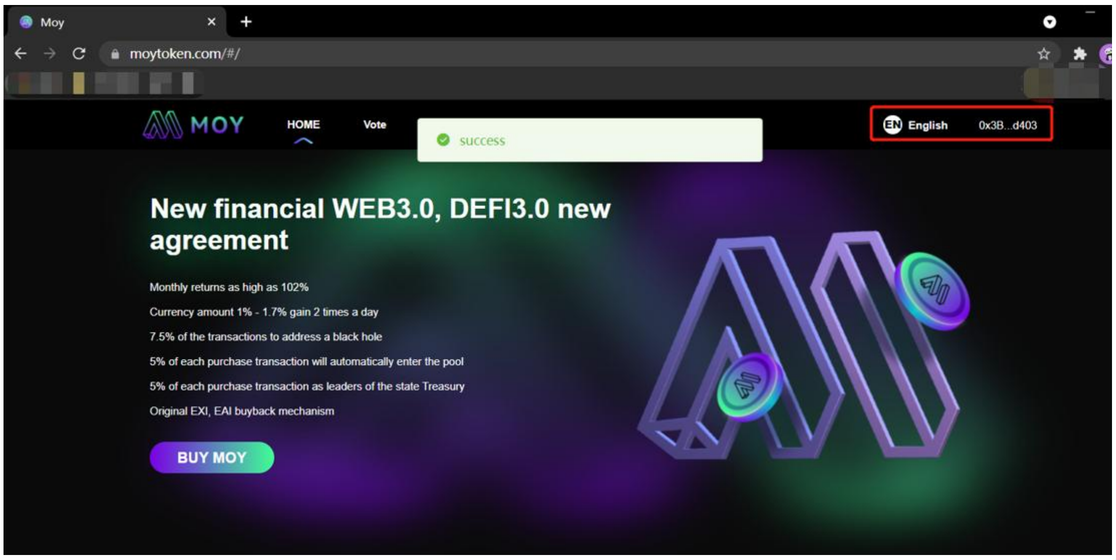
Monthly returns as high as 102%
Currency amount 1% - 1.7% gain 2 times a day
7.5% of the transactions to address a black hole
5% of each purchase transaction will automatically enter the pool
5% of each purchase transaction as leaders of the state Treasury
Original EXI, EAI buyback mechanism

MetaMask Notification

请求签名

账户: Account 1
余额: 0 ETH
来源: https://www.moytoken.com
正在签名:
消息: 7b3d942b-b7dc-4333-9b76-57d1942bd575

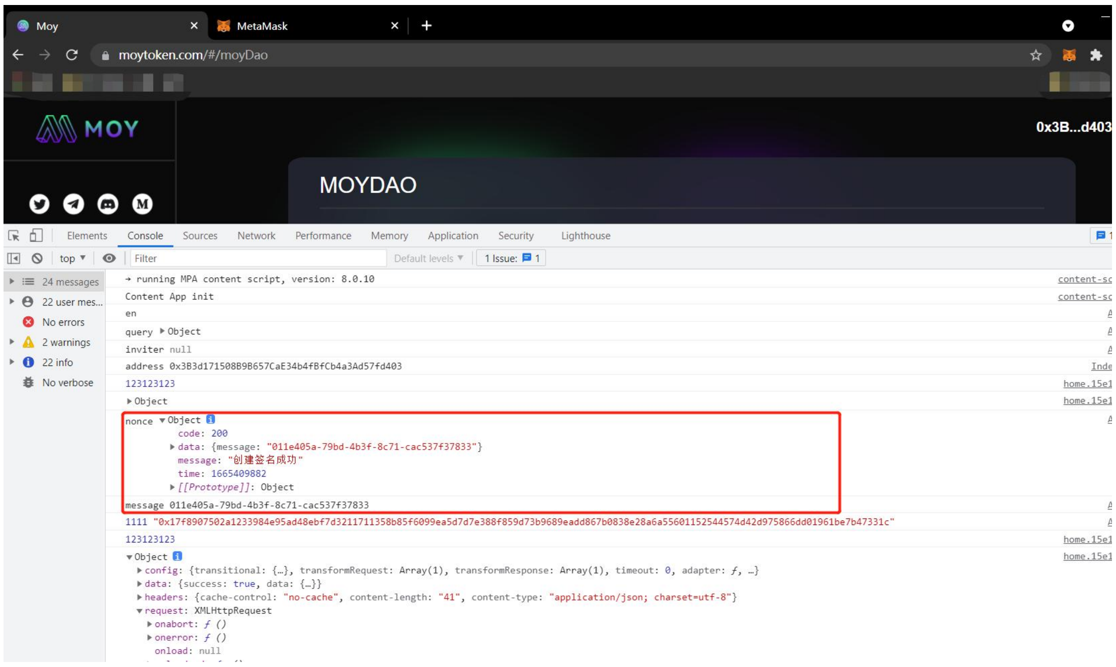
Step 2： Address information can be seen after signing



New financial WEB3.0, DEFI3.0 new agreement

Monthly returns as high as 102%
Currency amount 1% - 1.7% gain 2 times a day
7.5% of the transactions to address a black hole
5% of each purchase transaction will automatically enter the pool
5% of each purchase transaction as leaders of the state Treasury
Original EXI, EAI buyback mechanism

Create signature record：



```
→ running MPA content script, version: 8.0.10
Content App init
en
query ▸Object
inviter null
address 0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fd403
123123123
▸Object
nonce ▾Object
        code: 200
      ▸data: {message: "011e405a-79bd-4b3f-8c71-cac537f37833"}
        message: "创建签名成功"
        time: 1665409882
      ▸[[Prototype]]: Object
message 011e405a-79bd-4b3f-8c71-cac537f37833
1111 "0x17f8907502a1233984e95ad48ebf7d3211711358b85f6099ea5d7d7e388f859d73b9689eadd867b0838e28a6a55601152544574d42d975866dd01961be7b47331c"
123123123
▾Object
  ▸config: {transitional: {…}, transformRequest: Array(1), transformResponse: Array(1), timeout: 0, adapter: ƒ, …}
  ▸data: {success: true, data: {…}}
  ▸headers: {cache-control: "no-cache", content-length: "41", content-type: "application/json; charset=utf-8"}
  ▾request: XMLHttpRequest
    ▸onabort: ƒ ()
    ▸onerror: ƒ ()
     onload: null
```

Safety advice： NONE.

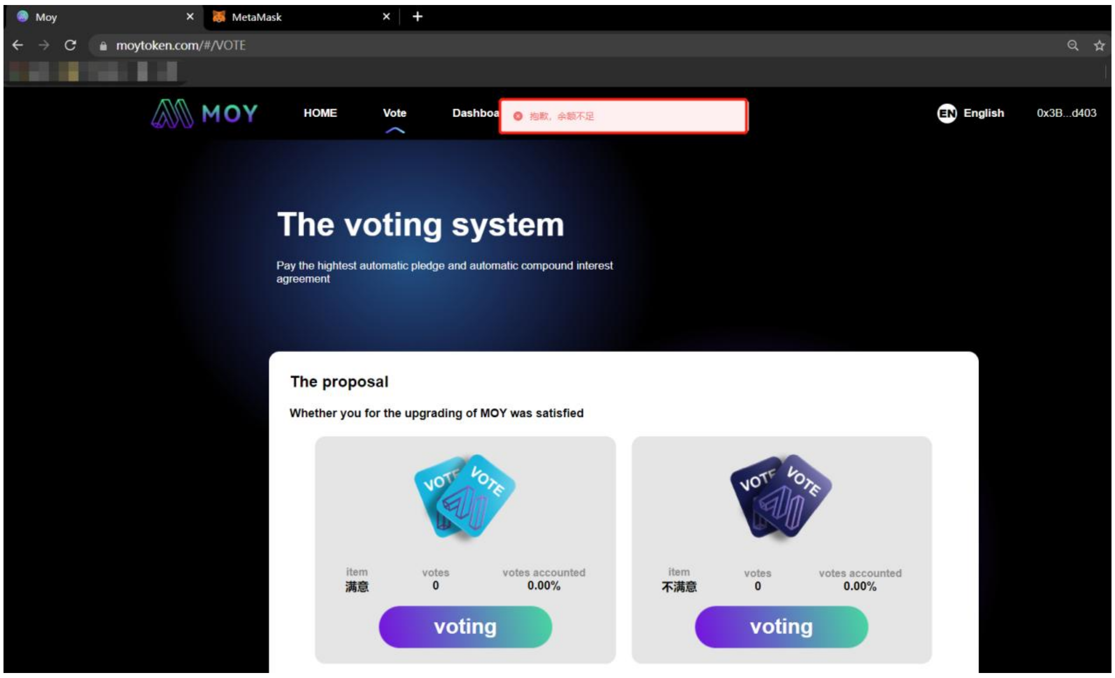
9.1.2. Voting authorization detection 【security】

Test description： Conduct security tests on the business logic of the voting authorization module on the official website to check whether there are security risks in the voting process, such as zero asset voting.

Test result： When voting, priority will be given to checking the number of assets owned by the current wallet address. If there is no asset, voting is not allowed.

Test process：

The number of assets owned by the current wallet address will be checked when voting



抱歉，余额不足

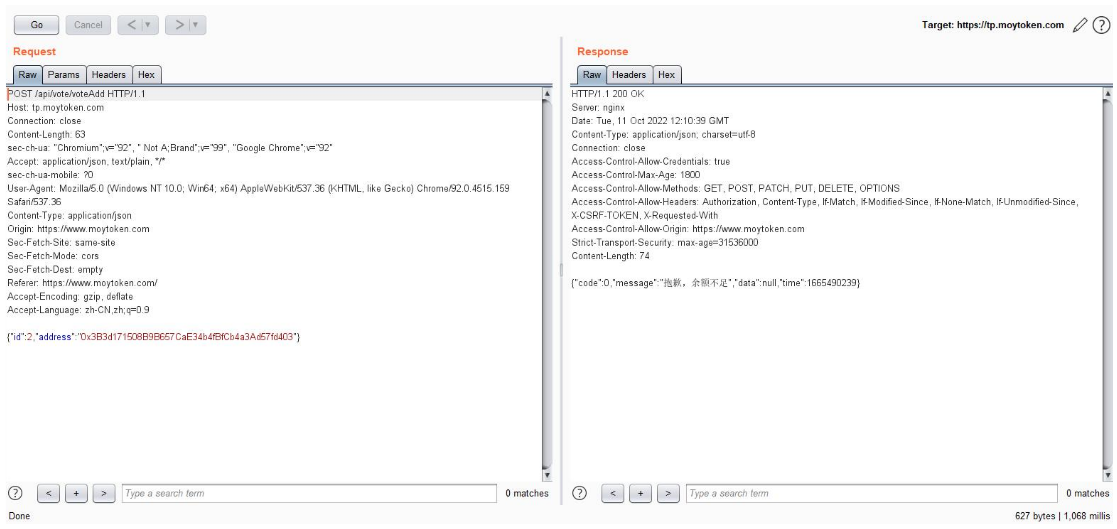
The voting system

Pay the hightest automatic pledge and automatic compound interest agreement

The proposal

Whether you for the upgrading of MOY was satisfied

item	votes	votes accounted
满意	0	0.00%
不满意	0	0.00%



```
POST /api/vote/voteAdd HTTP/1.1
Host: tp.moytoken.com
Connection: close
Content-Length: 63
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Content-Type: application/json
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

{"id":2,"address":"0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fd403"}
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 11 Oct 2022 12:10:39 GMT
Content-Type: application/json; charset=utf-8
Connection: close
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since, If-None-Match, If-Unmodified-Since,
X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: https://www.moytoken.com
Strict-Transport-Security: max-age=31536000
Content-Length: 74

{"code":0,"message":"抱歉，余额不足","data":null,"time":1665490239}
```

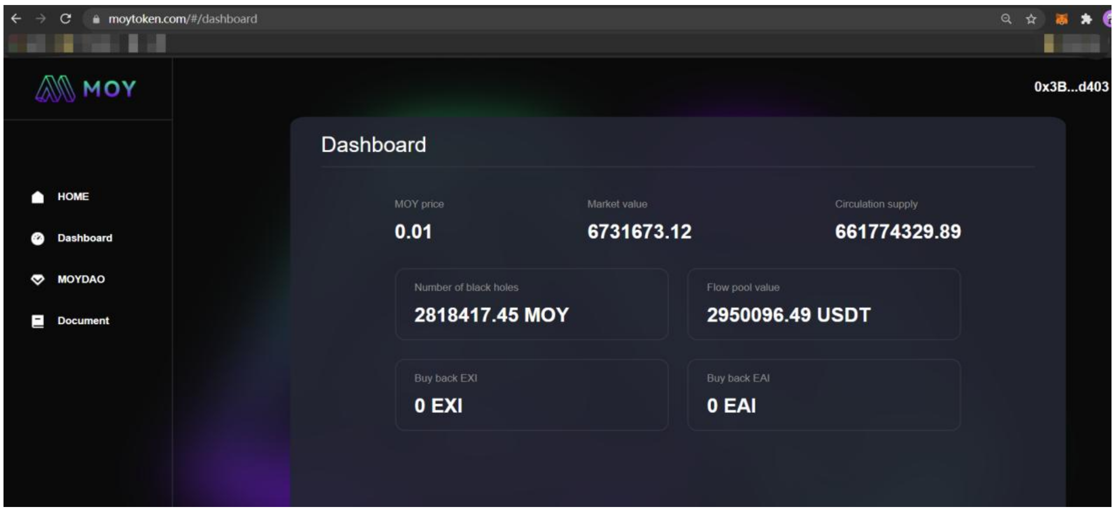
Safety advice：　NONE.

9.1.3. DOA data display【security】

Test description： Carry out security test on the display of official website DAO panel data to check whether there is a Web type security vulnerability.

Test result： No Web related security risks were found in the presentation of the official website DAO panel data.

Test process：



moytoken.com/#/dashboard

[redacted]

MOY

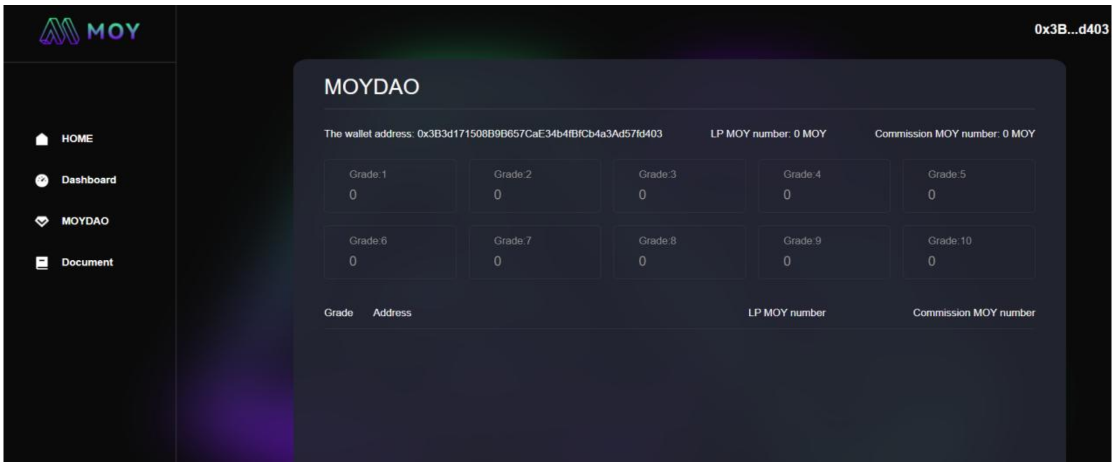
0x3B...d403

HOME
Dashboard
MOYDAO
Document

Dashboard

MOY price	Market value	Circulation supply
0.01	6731673.12	661774329.89

Number of black holes 2818417.45 MOY	Flow pool value 2950096.49 USDT
Buy back EXI 0 EXI	Buy back EAI 0 EAI



MOY

0x3B...d403

HOME
Dashboard
MOYDAO
Document

MOYDAO

The wallet address: 0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fd403 LP MOY number: 0 MOY Commission MOY number: 0 MOY

Grade:1 0	Grade:2 0	Grade:3 0	Grade:4 0	Grade:5 0
Grade:6 0	Grade:7 0	Grade:8 0	Grade:9 0	Grade:10 0

Grade	Address	LP MOY number	Commission MOY number

Safety advice： NONE.

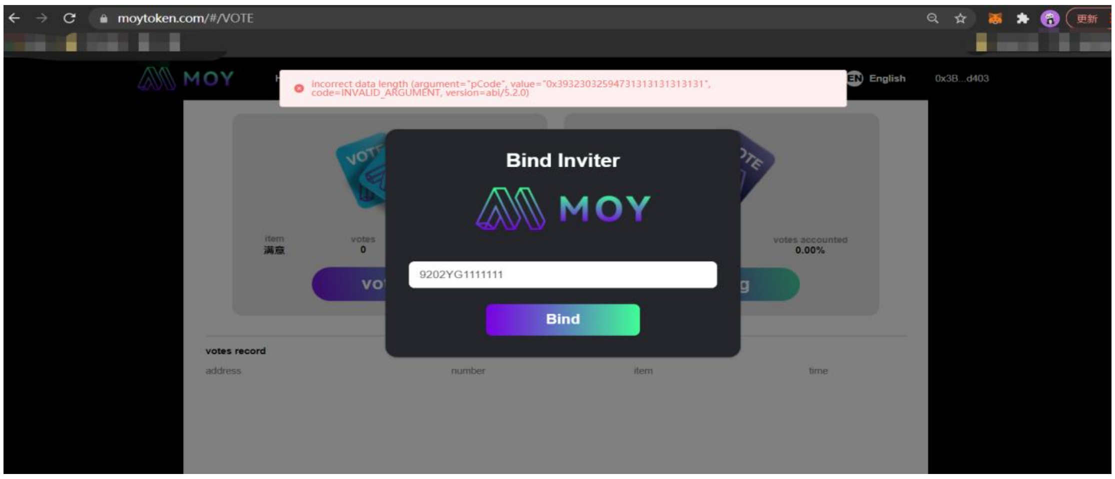
9.1.4. Invitation binding【security】

Test description： Carry out security test on the business logic design of binding the invitation relationship between the superior and subordinate of the official website to check whether there is a Web type security vulnerability.

Test result： When the official website is accessed for the first time, the wallet sign in process will be triggered, and the Bind Inviter process will also be triggered at the same time. That is, the invitation relationship is bound hierarchically, and the legitimacy of the invitation code is checked. Later, the user tries to bypass Bind Inviter by deleting page elements in the front end, and finds that the data will not be displayed normally. Bind Inviter verification must be completed, so there is no security problem.

Test process：

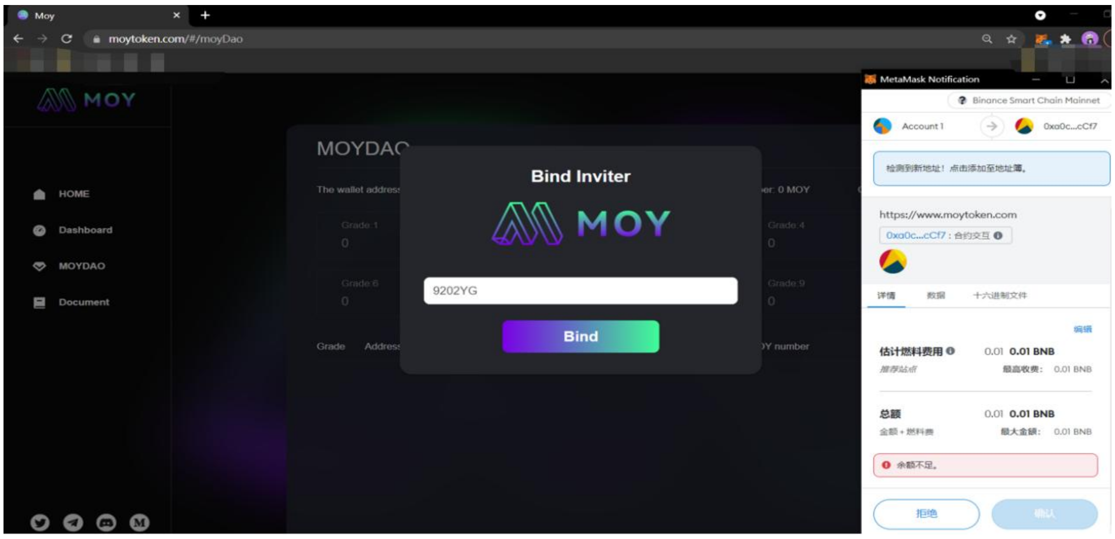
Validity check of invitation code



```
Incorrect data length (argument="pCode", value="0x393230325947313131313131", code=INVALID_ARGUMENT, version=abi/5.2.0)
```

9202YG1111111

You need to complete Bind Inviter to perform subsequent operations：



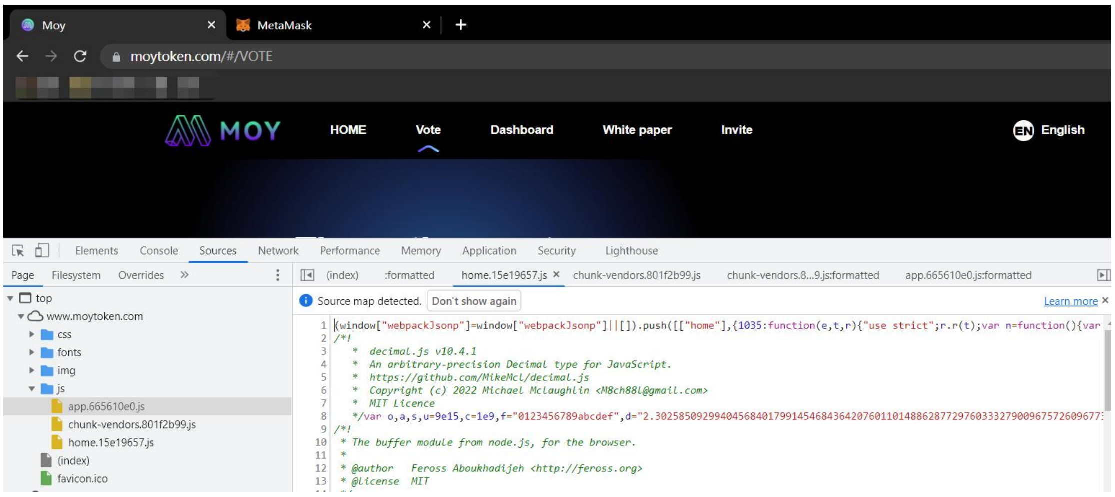
Safety advice： NONE.

9.1.5. Front end frame assembly 【security】

Test description： Conduct security tests on front-end framework components used in the official website to check whether vulnerable third-party libraries are used, such as Jquery.

Test result： Front end frame components do not pose security risks.

Test process：



```
(window["webpackJsonp"]=window["webpackJsonp"]||[]).push([["home"],{1035:function(e,t,r){"use strict";r.r(t);var n=function(){var
/*!
   *  decimal.js v10.4.1
   *  An arbitrary-precision Decimal type for JavaScript.
   *  https://github.com/MikeMcl/decimal.js
   *  Copyright (c) 2022 Michael Mclaughlin <M8ch88l@gmail.com>
   *  MIT Licence
   */var o,a,s,u=9e15,c=1e9,f="0123456789abcdef",d="2.302585092994045684017991454684364207601101488628772976033327900967572609677
/*!
 * The buffer module from node.js, for the browser.
 *
 * @author   Feross Aboukhadijeh <http://feross.org>
 * @license  MIT
```

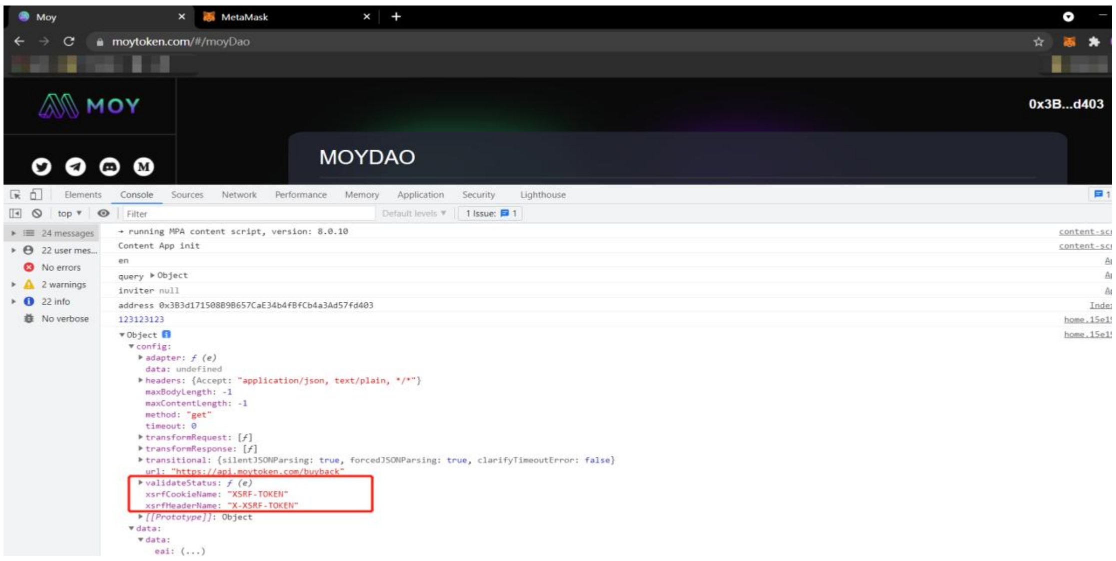
Safety advice： NONE.

9.1.6. Cross site attack risk 【security】

Test description： Check whether CSRF protection is provided during the interaction between the official website and the wallet to avoid the risk of cross site scripting attacks.

Test result： During the interaction between the official website and the wallet, the request header contains an X-XSRF-Token header to prevent attackers from conducting CSRF attacks.

Test process：



```
→ running MPA content script, version: 8.0.10
Content App init
en
query ▶Object
inviter null
address 0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fd403
123123123
▼Object
  ▼config:
    ▶adapter: ƒ (e)
     data: undefined
    ▶headers: {Accept: "application/json, text/plain, */*"}
     maxBodyLength: -1
     maxContentLength: -1
     method: "get"
     timeout: 0
    ▶transformRequest: [ƒ]
    ▶transformResponse: [ƒ]
    ▶transitional: {silentJSONParsing: true, forcedJSONParsing: true, clarifyTimeoutError: false}
     url: "https://api.moytoken.com/buyback"
    ▶validateStatus: ƒ (e)
     xsrfCookieName: "XSRF-TOKEN"
     xsrfHeaderName: "X-XSRF-TOKEN"
    ▶[[Prototype]]: Object
  ▼data:
    ▼data:
       eai: (...)
```

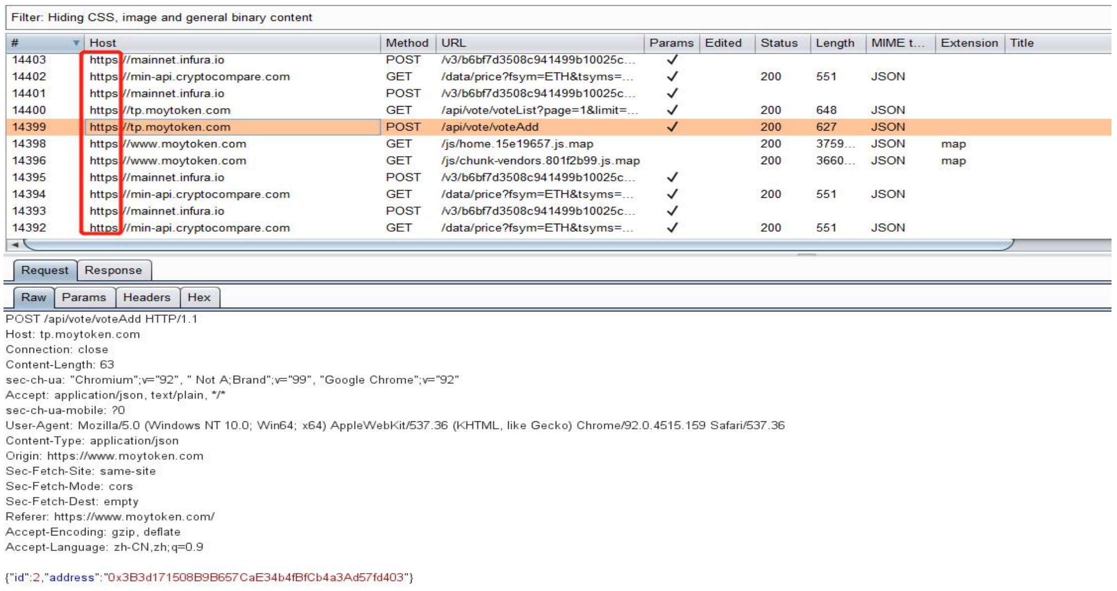
Safety advice： **NONE.**

9.1.7. Website communication protocol 【security】

Test description： Check whether the official website uses the https encrypted transmission protocol.

Test result： Official website uses Https encryption transmission protocol.

Test process：


Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length	MIME t...	Extension	Title
14403	https://mainnet.infura.io	POST	/v3/b6bf7d3508c941499b10025c...	✓						
14402	https://min-api.cryptocompare.com	GET	/data/price?fsym=ETH&tsyms=...	✓		200	551	JSON		
14401	https://mainnet.infura.io	POST	/v3/b6bf7d3508c941499b10025c...	✓						
14400	https://tp.moytoken.com	GET	/api/vote/voteList?page=1&limit=...	✓		200	648	JSON		
14399	https://tp.moytoken.com	POST	/api/vote/voteAdd	✓		200	627	JSON		
14398	https://www.moytoken.com	GET	/js/home.15e19657.js.map			200	3759...	JSON	map	
14396	https://www.moytoken.com	GET	/js/chunk-vendors.801f2b99.js.map			200	3660...	JSON	map	
14395	https://mainnet.infura.io	POST	/v3/b6bf7d3508c941499b10025c...	✓						
14394	https://min-api.cryptocompare.com	GET	/data/price?fsym=ETH&tsyms=...	✓		200	551	JSON		
14393	https://mainnet.infura.io	POST	/v3/b6bf7d3508c941499b10025c...	✓						
14392	https://min-api.cryptocompare.com	GET	/data/price?fsym=ETH&tsyms=...	✓		200	551	JSON		

```
POST /api/vote/voteAdd HTTP/1.1
Host: tp.moytoken.com
Connection: close
Content-Length: 63
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
Content-Type: application/json
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

{"id":2,"address":"0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fd403"}
```

Safety advice： NONE.

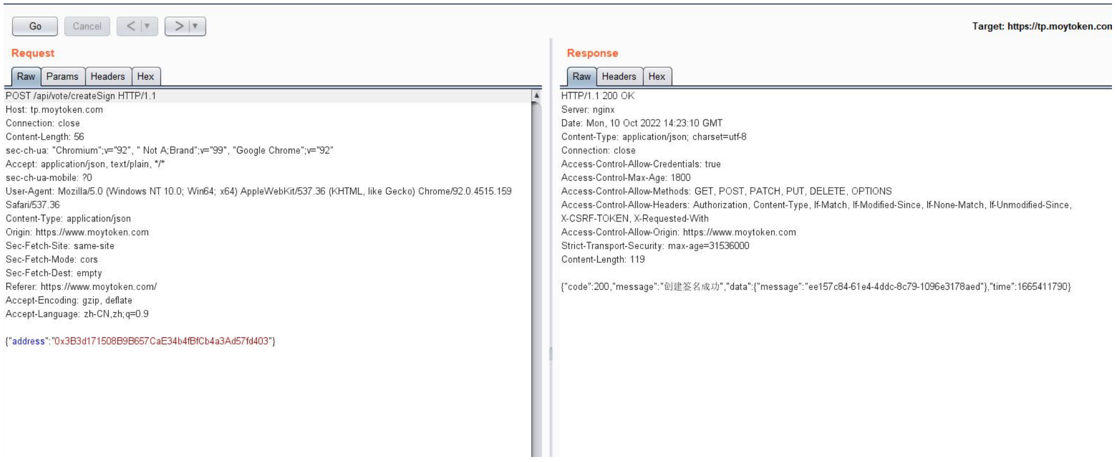
9.1.8. Create signature process【security】

Test description： Perform security test on the created signature API interface to check whether there are security risks.

Test result： Check the validity of the parameters when creating the signature API interface, and there is no security problem.

Test process：

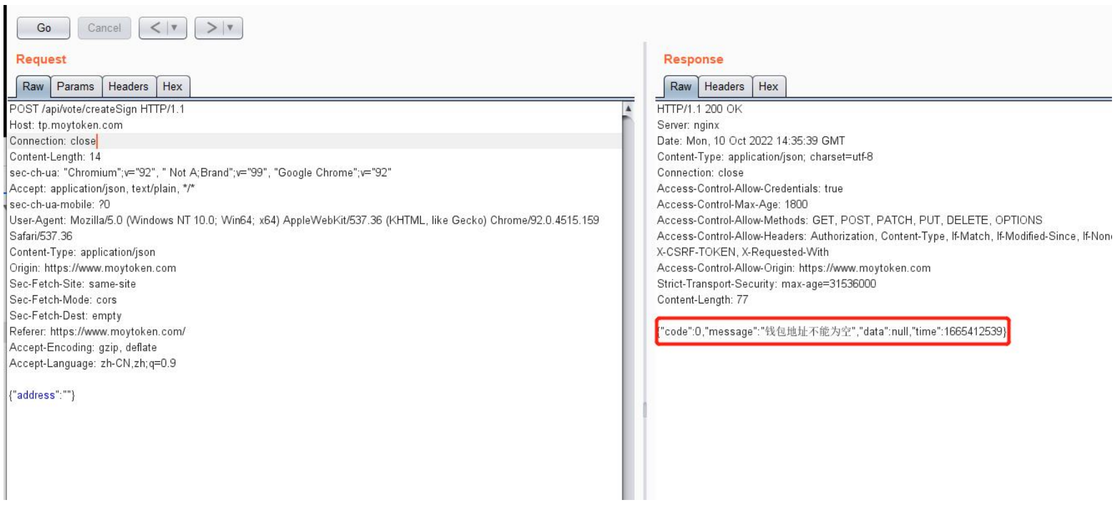
Normal requests are as follows：



```
POST /api/vote/createSign HTTP/1.1
Host: tp.moytoken.com
Connection: close
Content-Length: 56
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Content-Type: application/json
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

{"address":"0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fd403"}
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 10 Oct 2022 14:23:10 GMT
Content-Type: application/json; charset=utf-8
Connection: close
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since, If-None-Match, If-Unmodified-Since,
X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: https://www.moytoken.com
Strict-Transport-Security: max-age=31536000
Content-Length: 119

{"code":200,"message":"创建签名成功","data":{"message":"ee157c84-61e4-4ddc-8c79-1096e3178aed"},"time":1665411790}
```

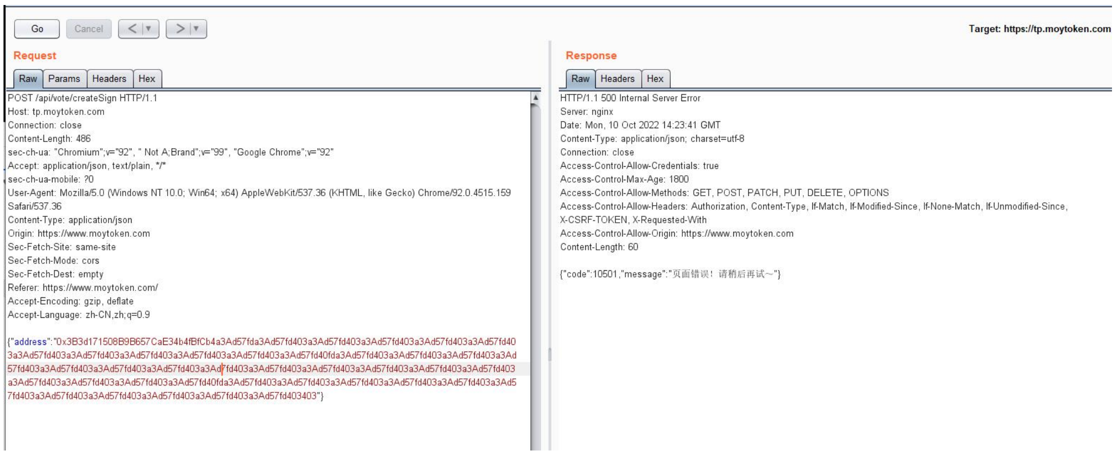
Check if the wallet address is empty：



```
POST /api/vote/createSign HTTP/1.1
Host: tp.moytoken.com
Connection: close
Content-Length: 14
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Content-Type: application/json
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

{"address":""}
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 10 Oct 2022 14:35:39 GMT
Content-Type: application/json; charset=utf-8
Connection: close
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since, If-Non
X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: https://www.moytoken.com
Strict-Transport-Security: max-age=31536000
Content-Length: 77

{"code":0,"message":"钱包地址不能为空","data":null,"time":1665412539}
```

Construction of super long address request：



```
POST /api/vote/createSign HTTP/1.1
Host: tp.moytoken.com
Connection: close
Content-Length: 486
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Content-Type: application/json
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

{"address":"0x3B3d171508B9B657CaE34b4fBfCb4a3Ad57fda3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd40
3a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd40fda3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad
57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad7fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403
a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd40fda3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad5
7fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403a3Ad57fd403403"}
```

```
HTTP/1.1 500 Internal Server Error
Server: nginx
Date: Mon, 10 Oct 2022 14:23:41 GMT
Content-Type: application/json; charset=utf-8
Connection: close
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since, If-None-Match, If-Unmodified-Since,
X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: https://www.moytoken.com
Content-Length: 60

{"code":10501,"message":"页面错误！请稍后再试~"}
```

Safety advice： NONE.

9.1.9. Poll list retrieval 【Moderate risk】

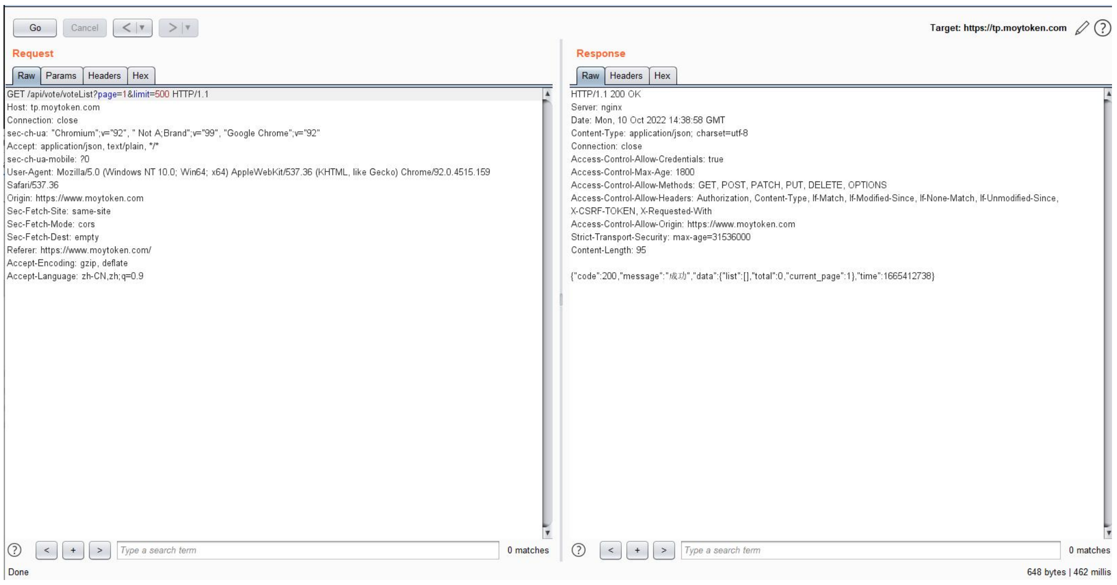
Test description： Perform security test on the polling list retrieval API interface to check whether there are security risks.

Test result： The API interface for polling list retrieval does not limit the number of limits, and there is a DOS risk.

Test process：

Normal requests are as follows：

```
GET /api/vote/voteList?page=1&limit=500 HTTP/1.1

Host: tp.moytoken.com

Connection: close

sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"

Accept: application/json, text/plain, */*

sec-ch-ua-mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36

(KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36

Origin: https://www.moytoken.com

Sec-Fetch-Site: same-site

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://www.moytoken.com/

Accept-Encoding: gzip, deflate

Accept-Language: zh-CN,zh;q=0.9
```



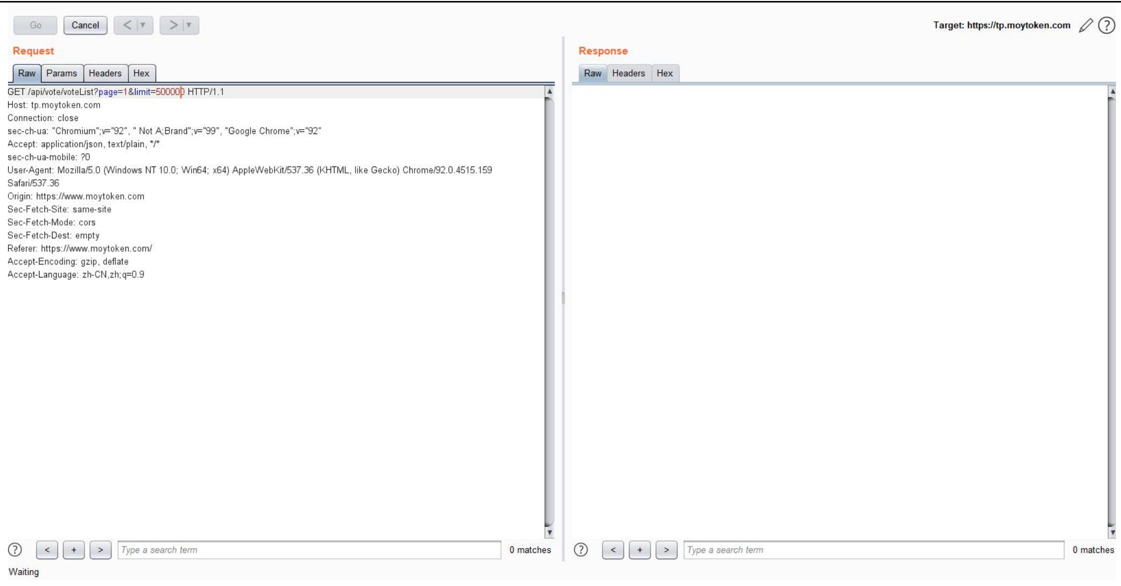
```
GET /api/vote/voteList?page=1&limit=500 HTTP/1.1
Host: tp.moytoken.com
Connection: close
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 10 Oct 2022 14:38:58 GMT
Content-Type: application/json; charset=utf-8
Connection: close
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since, If-None-Match, If-Unmodified-Since,
X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: https://www.moytoken.com
Strict-Transport-Security: max-age=31536000
Content-Length: 95

{"code":200,"message":"成功","data":{"list":[],"total":0,"current_page":1},"time":1665412738}
```

The construction of an overlength limit causes an overlength blank request delay：

```
GET /api/vote/voteList?page=1&limit=500000 HTTP/1.1

Host: tp.moytoken.com

Connection: close

sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"

Accept: application/json, text/plain, */*

sec-ch-ua-mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36

(KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36

Origin: https://www.moytoken.com

Sec-Fetch-Site: same-site

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://www.moytoken.com/

Accept-Encoding: gzip, deflate
```

```
Accept-Language: zh-CN,zh;q=0.9
```



```
GET /api/vote/voteList?page=1&limit=500000 HTTP/1.1
Host: tp.moytoken.com
Connection: close
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
Accept: application/json, text/plain, */*
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Origin: https://www.moytoken.com
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
```

Safety advice： Limit the range of limit, for example: 1-50

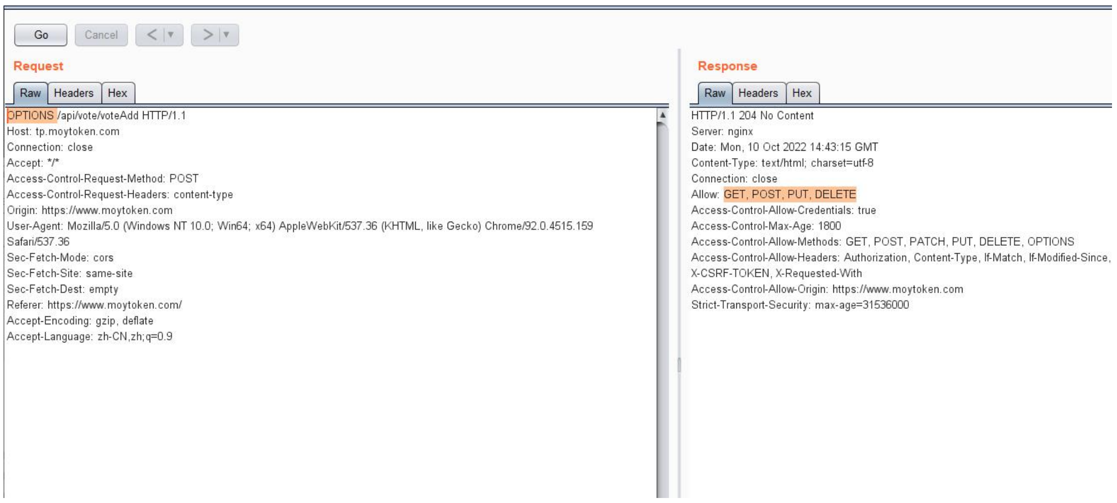
9.1.10. Dangerous HTTP methods 【Low risk】

Test description： Check the HTTP methods supported by the Web side of the official website to see if they support dangerous HTTP methods, such as DELETE, PUT, etc.

Test result： The official website supports dangerous HTTP methods.

Test process：

Normal requests are as follows：



```
OPTIONS /api/vote/voteAdd HTTP/1.1
Host: tp.moytoken.com
Connection: close
Accept: */*
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: https://www.moytoken.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Sec-Fetch-Dest: empty
Referer: https://www.moytoken.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
```

```
HTTP/1.1 204 No Content
Server: nginx
Date: Mon, 10 Oct 2022 14:43:15 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Allow: GET, POST, PUT, DELETE
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since,
X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: https://www.moytoken.com
Strict-Transport-Security: max-age=31536000
```

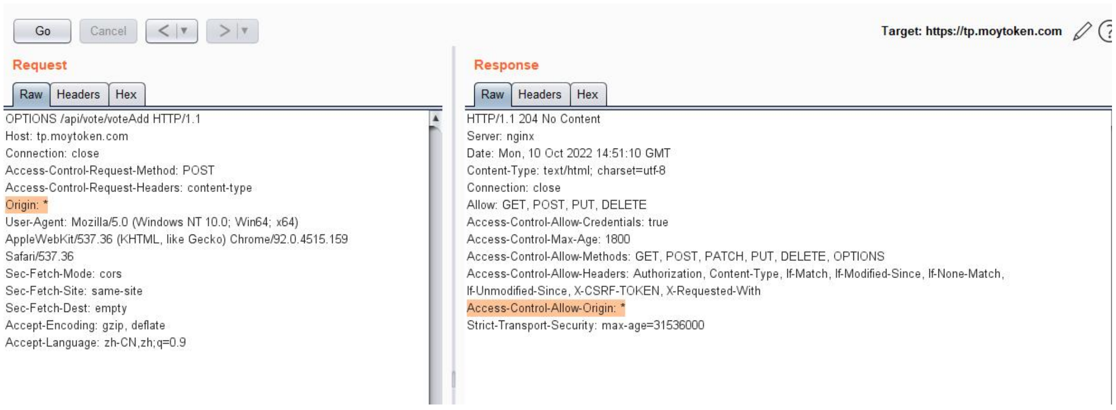
Safety advice： It is recommended to remove the HTTP method of PUT and DELETE.

9.1.11. CORS cross domain risk【Low risk】

Test description： Check the use of the official website to allow CORS cross domain requests.

Test result： There is a risk of CORS cross domain request attack on the official website.

Test process：



```
OPTIONS /api/vote/voteAdd HTTP/1.1
Host: tp.moytoken.com
Connection: close
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: *
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159
Safari/537.36
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Sec-Fetch-Dest: empty
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
```

```
HTTP/1.1 204 No Content
Server: nginx
Date: Mon, 10 Oct 2022 14:51:10 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Allow: GET, POST, PUT, DELETE
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1800
Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS
Access-Control-Allow-Headers: Authorization, Content-Type, If-Match, If-Modified-Since, If-None-Match,
If-Unmodified-Since, X-CSRF-TOKEN, X-Requested-With
Access-Control-Allow-Origin: *
Strict-Transport-Security: max-age=31536000
```

Safety advice： Reasonably configure Access Control Allow as needed, for example, configure the trust domain instead of using * to run all domain accesses

10. Appendix:Analysis tools

10.1.Solgraph

Solgraph is used to generate a graph of the call relationship between smart contract functions, which is convenient for quickly understanding the call relationship between smart contract functions.

Project address：https://github.com/raineorshine/solgraph

10.2.Sol2uml

Sol2uml is used to generate the calling relationship between smart contract functions in the form of UML diagram.

Project address：https://github.com/naddison36/sol2uml

10.3.Remix-ide

Remix is a browser based compiler and IDE that allows users to build contracts and debug transactions using the solid language.

Project address：http://remix.ethereum.org

10.4.Ethersplay

Etherplay is a plug-in for binary ninja. It can be used to analyze EVM bytecode and graphically present the function call process.

Project address：https://github.com/crytic/ethersplay

10.5.Mythril

Mythril is a security audit tool for EVM bytecode, and supports online contract audit.

Project address：https://github.com/ConsenSys/mythril

10.6.Echidna

Echidna is a security audit tool for EVM bytecode. It uses fuzzy testing technology and supports integrated use with truss.

Project address：https://github.com/crytic/echidna

11. DISCLAIMERS

Chainlion only issues this report on the facts that have occurred or existed before the issuance of this report, and assumes corresponding responsibilities. For the facts occurring or existing after the issuance, chainlion cannot judge the security status of its smart contract, and is not responsible for it. The security audit analysis and other contents in this report are only based on the documents and materials provided by the information provider to chainlion as of the issuance of this report. Chainlion assumes that the information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed or reflected inconsistent with the actual situation, chainlion shall not be liable for the losses and adverse effects caused thereby. Chainlion only conducted the agreed safety audit on the safety of the project and issued this report. Chainlion is not responsible for the background and other conditions of the project.

CHAINLION

Blockchain world patron saint building blockchain ecological security