SCTF 2018 Qual Writeup

by horoo

1. BankRobber

Solidity 파일의 취약점을 패치하는 문제이다.

```
    //withdraw my balance
    function withdraw(uint256 value) public{
        require(balance[msg.sender] >= value);
        // patch reentrancy
        balance[msg.sender] -= value;
        msg.sender.call.value(value)();
    }
```

[그림1] withdraw

```
//transfer my balance to others
function multiTransfer(address[] to_list, uint256 value) public {
    require(balance[msg.sender] >= value*to_list.length);

    for(uint i=0; i < to_list.length; i++){
        // patch unsigned int overflow
        transfer(to_list[i], value);
    }
}
```

[그림2] multiTransfer

```
//Only bank owner can deliver donations to anywhere he want.
function deliver(address to) public {
    // patch tx.origin -> msg.sender
    require(msg.sender == owner);
    to.transfer(donation_deposit);
    donation_deposit = 0;
}
```

[그림3] deliver

reentrancy와 overflow 버그를 패치했고 deliver에서는 bank owner인지를 비교하려면 tx.origin이 아닌 msg.sender와 비교해야 할 것이라고 생각했다.

Flag: SCTF{sorry_this_transaction_was_sent_by_my_cat}

2. dingJMax

사용자 키 입력에 따라 flag를 계산한다.

```
v6 = wgetch(stdscr);
if ( v6 == 'f' )
{
  v9 = 1;
  sub_401C9A((unsigned int)('f' * v16));
  sub_400C5E(dest);
  goto LABEL_20;
}
if ( v6 > 'f' )
{
  if ( v6 == 'j' )
  {
    v10 = 1;
    sub_401C9A((unsigned int)('j' * v16));
    sub_400C5E(dest);
    goto LABEL_20;
  }
  if ( v6 == 'k' )
  {
    v11 = 1;
    sub_401C9A((unsigned int)('k' * v16));
    sub_400C5E(dest);
    goto LABEL_20;
  }
}
else if ( v6 == 'd' )
{
  v8 = 1;
  sub_401C9A((unsigned int)('d' * v16));
  sub_400C5E(dest);
  goto LABEL_20;
}
```

[그림3] key 입력

입력한 key와 v16을 곱한 값을 인자로 받아 계산하는데 루프를 20번 돌 때마다 off_603280에서 note를 하나씩 읽어 길이 dword_67600 배열의 처음에 저장하고 한칸씩 올린다.

dword_67600[19]의 note와 key 입력이 같다면 perfect를 받는다.

```
if ( v16 == 20 * ((unsigned __int64)(0x0CCCCCCCCCCCCCCDLL * (unsigned __int128)v16 >> 64) >> 4) )
{
  for ( k = 19; k > 0; --k )
    dword_607600[k] = dword_607600[k - 1];
  dword_607600[0] = *(_DWORD *)off_603280[(unsigned __int64)(0x0CCCCCCCCCCCCCCDLL * (unsigned __int128)v16 >> 64) >> 4];
}
```

[그림4] note 저장

```
off_603280      dq offset asc_402275    ; DATA XREF: main+338↑o
                                        ; main+632↑r
                                        ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset a0            ;       "0"
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset asc_402275    ;       ""
                dq offset a0_0          ;       "0"
```

[그림5] note

따라서 v16 = note의 위치 * 20 + 380으로 계산하면 flag를 구할 수 있다.

Flag: SCTF{I_w0u1d_l1k3_70_d3v3l0p_GUI_v3rs10n_n3x7_t1m3}

3. LowerLevel

빵판을 잘 읽으면 input을 a, b, c, d, e라고 했을 때 output을 구할 수 있다.

```
o1 = (b | d) & a
o2 = (((b & e) | (((b+1)%2) & ((e+1)%2))) | (a | d)) ^ c
o3 = (((e & d) | (((e+1)%2) & ((d+1)%2))) | ((b+1)%2)) ^ c
o4 = ((b | e) | ((d+1)%2)) ^ c
o5 = ((((((b+1)%2) & ((e+1)%2))) | (((b+1)%2) & d)) | ((b & ((d+1)%2)) & e) | ((d & ((e+1)%2) | a))) ^ c
o6 = ((((b+1)%2) & ((e+1)%2)) | ((d & (e+1)%2))) ^ c
o7 = (((((e+1)%2) & ((d+1)%2)) | (b & ((d+1)%2))) | (b & ((e+1)%2)) | a) ^ c
o8 = (((b & ((d+1)%2)) | a) | ((((b+1)%2) & d) | (d & (e+1)%2))) ^ c
```

Flag: SCTF{NUMBERDECODERINBREADBOARD}