2018强网杯部分writeup

原创

snowleopard_bin 于 2018-09-20 09:05:28 发布 1810 收藏 2
分类专栏： CTF 文章标签： CTF

0x00 签到题

操作内容：

打开题目就看到flag

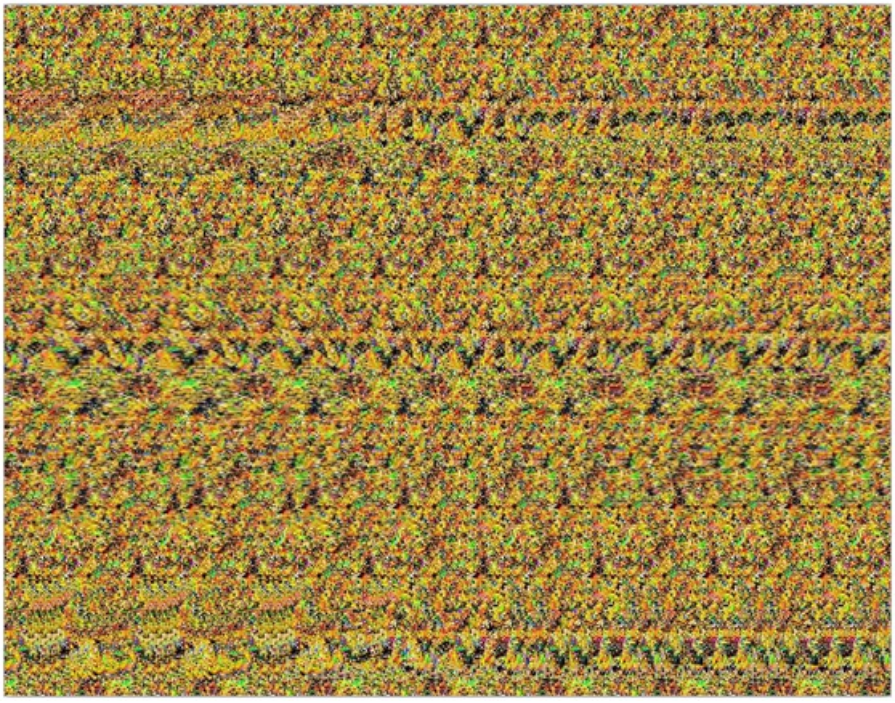
FLAG值：

flag{welcome_to_qwb}

0x01 Welcome

操作内容：

|拿到一张图片：



放进Stegsolve中，一点点偏移图片，慢慢发现有字，offset到100时就能清楚看到

得到flag

FLAG值：

QWB{W3lc0me}

0x02 web签到
操作内容：

感觉有些难。上来第一个md5挑战，前端提示如下：

```
<h2>The Fisrt Easy Md5 Challenge</h2>
<!--
if($_POST['param1']!=$_POST['param2'] &&
md5($_POST['param1'])==md5($_POST['param2'])){
die("success!"); }
-->
```

发现是php弱相等漏洞，需要构造md5开头为0e的字符串，有很多，输入QNKCDZO和s878926199a，通关

之后是第二个md5挑战，前端提示如下：

```
<h2>The Second Easy Md5 Challenge</h2>
<!--
if($_POST['param1']!==$_POST['param2'] &&
md5($_POST['param1'])===md5($_POST['param2'])){
die("success!"); }
-->
```

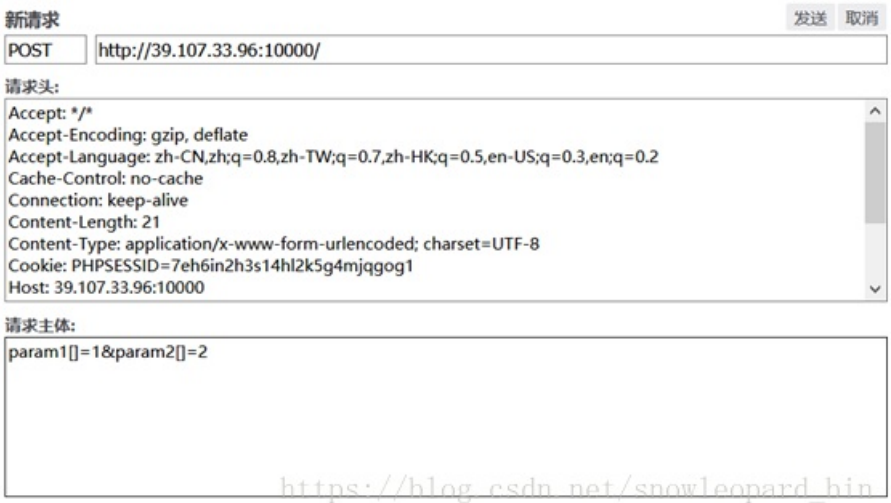
发现是强相等，只能利用php的md5函数对于php数组的漏洞绕过，阅读前端的submit函数发现前端是利用$.post("/",{'param1':param1,'param2':param2}函数提交的表单，便可以修改submit函数来提交数组从而绕过。修改的部分如图所示：

```
<input id="input-7" class="input__field input__field--
kuro" name="param1[]" type="text">
```

```
<input id="input-8" class="input__field input__field--
kuro" name="param2[]" type="text">
```

```
<script>
                    function submit(){
                        var param1[]=array(1=>$('input[name]').eq(0).val());
                        var param2[]=array(1=>$('input[name]').eq(1).val());
                        $.post("/",
{'param1':param1[],'param2':param2[]},function(data,status){
                            if(data.indexOf('success')!=-1){
                                alert("success!");
                                location.reload();
                            }
                            else{
                                alert("failed!");
                            }
                        })
                    }
```


构造发送包如下：



发送通关

最后是md5终极挑战，前端提示如下：

```
<h2>Md5 Revenge Now!</h2>
<!--
if((string)$_POST['param1']!==(string)$_POST['param2'] &&
md5($_POST['param1'])===md5($_POST['param2'])){ die("success!"); }
-->
```


看了很久都没有发现漏洞，只好强行产生碰撞，md5近年来已经被发现存在碰撞，Google到碰撞字符串对提交，payload如下

```
param1=%D11%DD%02%C5%E6%EE%C4i%3D%9A%06%98%AF%F9%5C%2F%CA%B5%87%12F%7E%A
```

提交即可获得flag

```
响应载荷
1  success! flag is QWB{s1gns1gns1gnaftermd5}
```


FLAG值：

QWB{s1gns1gns1gnaftermd5}

0x03 Simplecheck
操作内容：

将apk转换成dex2jar格式，用jd-gui打开，定位到关键函数：

```
if (a.a(localEditText.getText().toString()))
{
  Toast.makeText(jdField_this, "You get it~", 1).show();
  return;
}
```



```
package com.a.simplecheck;

public class a
{
  private static int[] a = { 0, 146527998, 205327308, 94243885, 138810487, 408218567, 77866117, 71548549, 563255818, 559010506, 449018203, 576200
  private static int[] b = { 13710, 46393, 49151, 36900, 59564, 35883, 3517, 52957, 1509, 61207, 63274, 27694, 20932, 37997, 22069, 8438, 33995,
  private static int[] c = { 38129, 57355, 22538, 47767, 8940, 4975, 27050, 56102, 21796, 41174, 63445, 53454, 28762, 59215, 16407, 64340, 37644,
  private static int[] d = { 0, -341994984, -370404060, -257581614, -494024809, -135267265, 54930974, -155841406, 540422378, -107286502, -1280569

  public static boolean a(String paramString)
  {
    if (paramString.length() != b.length) {
      return false;
    }
    int[] arrayOfInt = new int[a.length];
    arrayOfInt[0] = 0;
    byte[] arrayOfByte = paramString.getBytes();
    int i = arrayOfByte.length;
    int j = 0;
    int k = 1;
    while (j < i)
    {
      arrayOfInt[k] = arrayOfByte[j];
      k++;
      j++;
    }
    for (int m = 0;; m++)
    {
      if (m >= c.length) {
        break label175;
      }
      if ((a[m] != b[m] * arrayOfInt[m] * arrayOfInt[m] + c[m] * arrayOfInt[m] + d[m]) || (a[(m + 1)] != b[m] * arrayOfInt[(m + 1)] * arrayOfInt[
        break;
      }
    }
    label175:
    return true;
  }
```


就是解方程：a = b*x*x + c*x + d

上脚本：

```
import math
a =[ 0, 146527998, 205327308, 94243885, 138810487, 408218567, 77866117, 71548549
b =[ 13710, 46393, 49151, 36900, 59564, 35883, 3517, 52957, 1509, 61207, 63274,
c =[38129, 57355, 22538, 47767, 8940, 4975, 27050, 56102, 21796, 41174, 63445, 5
d=[0, -341994984, -370404060, -257581614, -494024809, -135267265, 54930974, -155
print(len(a))#35
print(len(b))#34
print(len(c))
print(len(d))
flag1=''
flag2=''
for i in range(34):
    res1 = math.sqrt((c[i]*c[i]+4*b[i]*(a[i]-d[i]))/(4*b[i]*b[i]))
    res2 = math.sqrt((c[i]*c[i]+4*b[i]*(a[i+1]-d[i]))/(4*b[i]*b[i]))
    x1=res1-c[i]/(2*b[i])
    x2=res2-c[i]/(2*b[i])
    if x1-int(x1)>0.5:
        x1+=1
    if x2-int(x2)>0.5:
        x2+=1
    flag1+=chr(int(x1))
    flag2+=chr(int(x2))
    print("flag1 =",flag1)
    print("flag2 =",flag2)
```


执行后得到flag

FLAG值：

flag{MAth_i&_GOOd_DON7_90V_7hlnK?}

0x03 StreamGame1

操作内容：

将压缩包下载下来之后打开发现key文件和一个python文件，用Python IDLE打开后发现Stream实现函数，flag经处理后生成若干个字节存放于key文件中，在python文件中提示了flag的长度1号、2号、4号flag的长度分别为24、25、27，去掉花括号和“flag”长度分别为17、19、21，并且由代码

可知flag中为零壹串。若爆破，求解的时间复杂度为

感觉可以用爆破求解！

步骤：

由于flag要达到一定的长度要求，将flag初始化为首位为1剩余位为0且与mask对齐的的N位（N=17,19,21）二进制数，然后循环处理，每次处理完毕后flag加一，将得到的结果与key中的数据一一对比，直到找到跟key中结果完全匹配的那个flag。

优化：

对比步骤优化：当比较第i个字节，若匹配，则比较下一个字节，直至结束，若不匹配、判断下一个flag是否匹配。

```
def lfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    while i!=0:
        lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)
'''
R=int(flag[5:-1],2)
'''
mask    =   0b10100110001000111100
flag    =   0b10000000000000000000
key     =   [0x55, 0x38, 0xf7, 0x42, 0xc1, 0x0d, 0xb2, 0xc7, 0xed, 0xe0, 0x24, 0
num = 12
while flag <= 0b11111111111111111111:
    R = flag
    no = 0
    for i in range(12):
        tmp=0
        for j in range(8):
            (R,out)=lfsr(R,mask)
            tmp=(tmp << 1)^out
        if key[no] == tmp:
            no += 1
        else:
            flag += 1
            break
    if no == num:
        print bin(flag)
        flag += 1
```


FLAG值：

flag{1110101100001101011}

0x05 StreamGame2
操作内容：

几乎与StreamGame1相同，进行了下优化：不用将所有字节对比完毕，经发现对比前三个字节得到的结果即可保证最后得到正确的结果，速度增加。

FLAG值：

flag{11011110010100110101001}

0x06 StreamGame4
操作内容：

思路还是一样，进行下优化。

对比方式优化：将对比3个字节改为对比前19个bit，发现速度增加了4倍左右。

FLAG值：

flag{10010011101010110101 1}

0x07 调查问卷
操作内容：

填完调查问卷就能得到flag

FLAG值：

flag{强网杯强国梦}