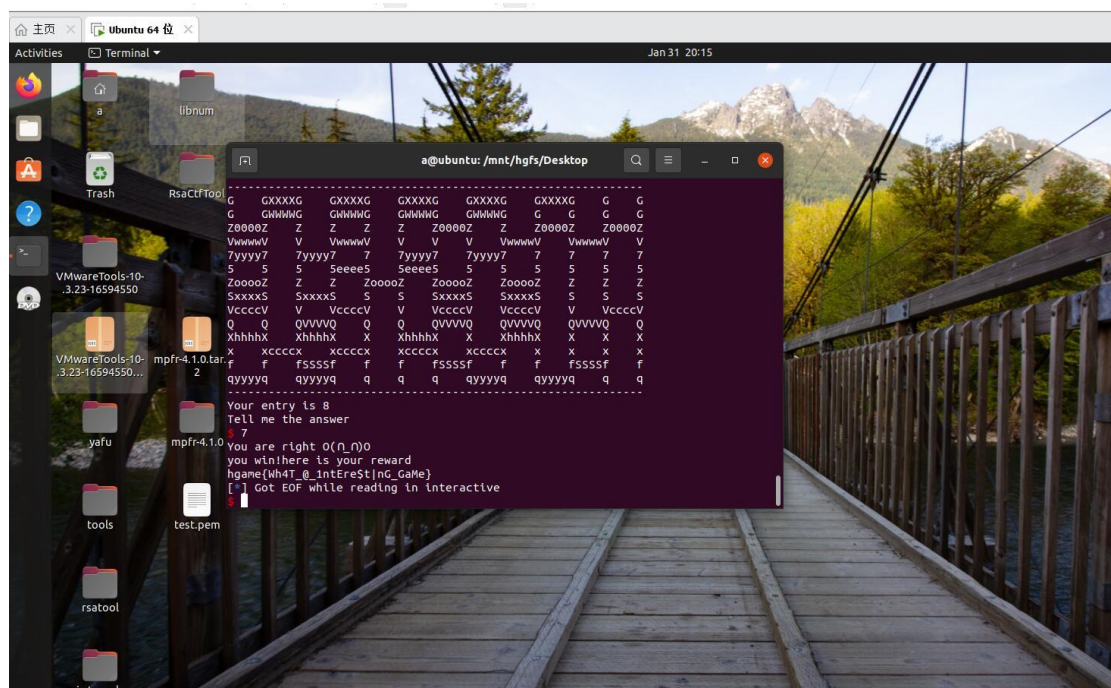


HGAME 2022 Week2 writeup by nerowander

MISC

奇妙小游戏

规则：鬼脚图，而且需要注意 0 相当于数组 `a[0]`，即为第一个元素的意思，需要一定的细心能力和手速，解完关卡之后得到 flag



你上当了 我的很大

诈骗压缩包，根据 hint 找到四个不一样的二维码(其中 2 个题目已给，另外 2 个需要在不一样的诈骗视频中的末尾找到)

将这几个二维码一个一个解码，发现内含 base64，猜测 base64 可能暗藏了一些文件，拿去解码，得到四个分开的二维码。

利用 ps 将它们拼在一起，扫码得到 flag

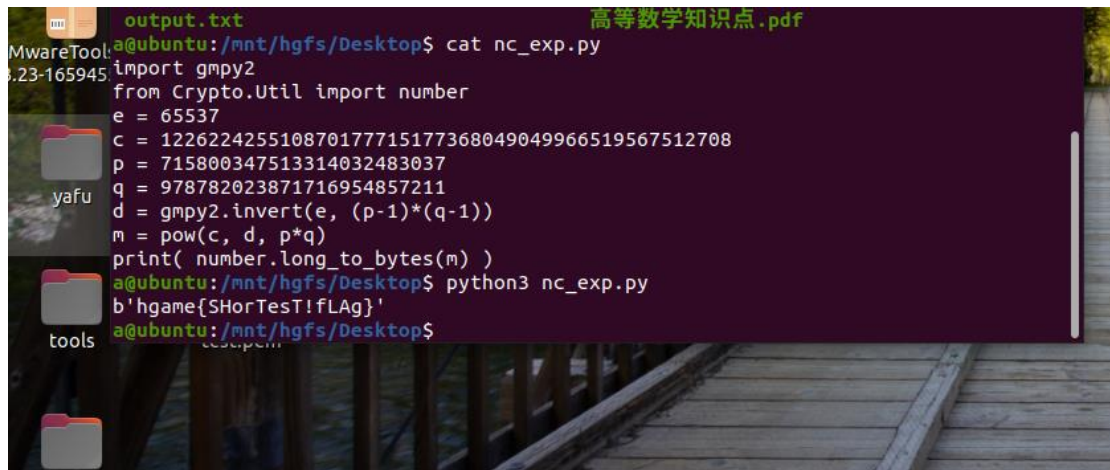
这二维码真小.....不是应该很大吗



CRYPTO

RSA Attack

n 算是比较小的数字了，可以利用在线工具暴力拆解得到 p 和 q ，然后用正常的套路就可以得到 flag 了



```
output.txt 高等数学知识点.pdf
MwareTool: a@ubuntu:/mnt/hgfs/Desktop$ cat nc_exp.py
.23-165945 import gmpy2
from Crypto.Util import number
e = 65537
c = 122622425510870177715177368049049966519567512708
p = 715800347513314032483037
q = 978782023871716954857211
d = gmpy2.invert(e, (p-1)*(q-1))
m = pow(c, d, p*q)
print( number.long_to_bytes(m) )
a@ubuntu:/mnt/hgfs/Desktop$ python3 nc_exp.py
b'hgame{SHorTeSt!fLAG}'
a@ubuntu:/mnt/hgfs/Desktop$
```

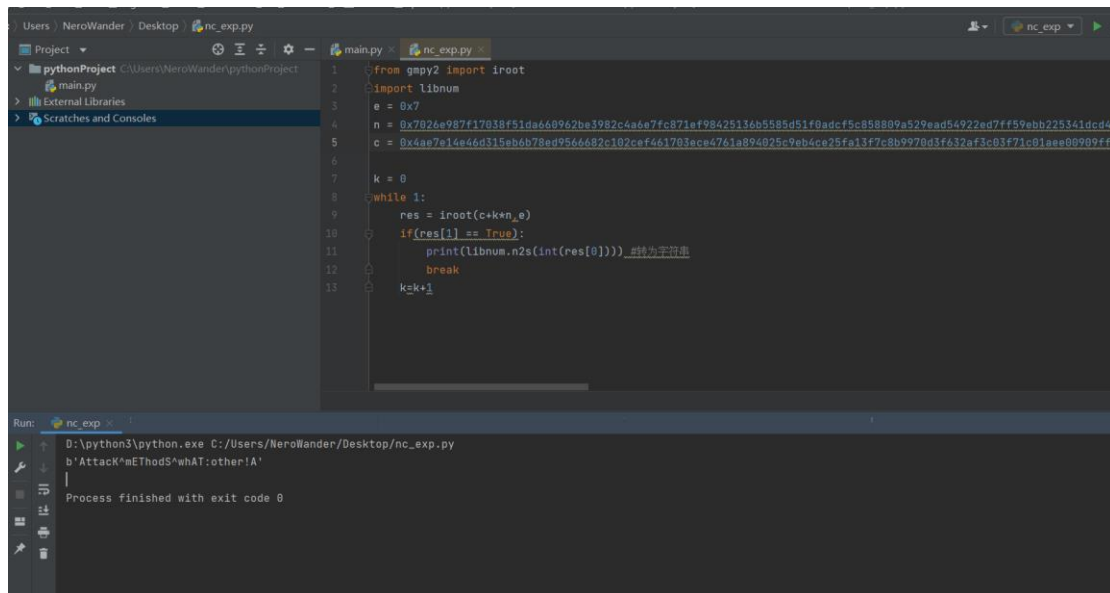
RSA Attack2

首先是不能大数分解的

part1: 利用 n_1, n_2 的公约数求解(脚本找不到了，这里 part1 的截图暂无了.....之后去看官方 wp 好了，附上结果)

```
# task1 (已解决) hgame{RsA@hAS!a&VAriETY?of.
e = 65537
n1 = 146115456051079508275810051653276947828231886031517681697314314183613062311149850377
c1 = 965075803554932988664271816439183802328812013694203741320763105376036912584995031647
n2 = 209374787251099838030791854504496165674645969613487274538172490351100475855801428231
c2 = 115365069453137471804424734616589123071544608690033927321784576432240579698382246010
# task2
```

part2: 低加密指数攻击



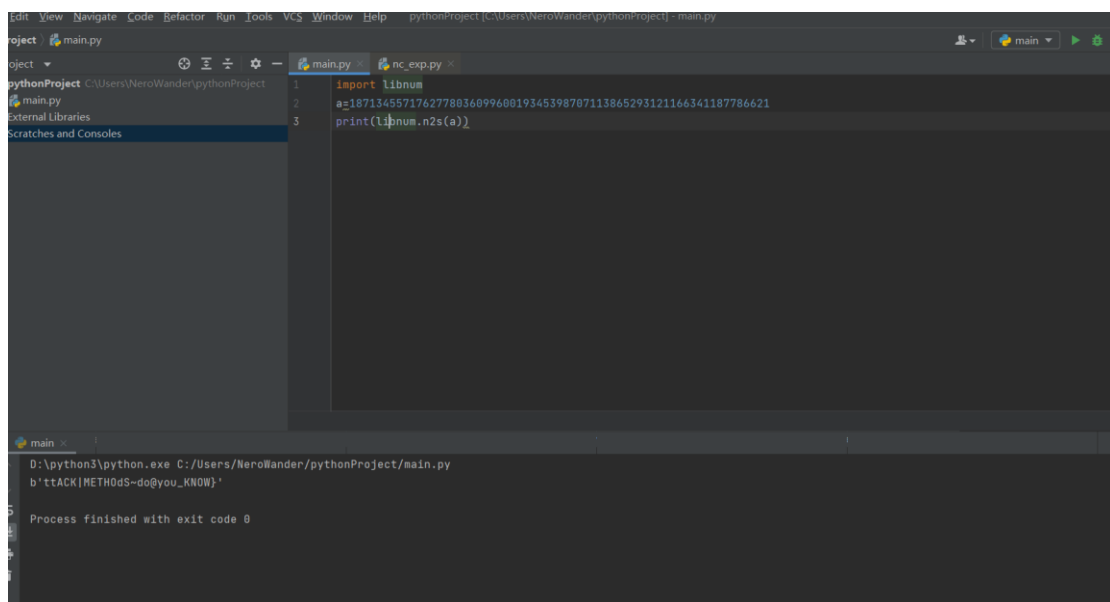
```
1 from gmpy2 import iroot
2 import libnum
3 e = 0x7
4 n = 0x7026e987f17038f51da660962be3982c4a6e7fc871ef98425136b5585d51f0adcf5c858809a529ead54922ed7ff59ebb225341dcd4b
5 c = 0x4ae7e14e46d315eb6b78ad9566682c102cef461703ace4761a894025c9eb4ce25fa13f7c8b9970d3f632af3c03f71c01aee60909ff7
6
7 k = 0
8 while 1:
9     res = iroot(c+k*n,e)
10    if(res[1] == True):
11        print(libnum.n2s(int(res[0]))) #转为字符串
12        break
13    k+=1
```

Run: nc_exp x

```
D:\python3\python.exe C:/Users/NeroWander/Desktop/nc_exp.py
b'Attack*methodS^hAT:other!A'
```

Process finished with exit code 0

part3: 两个加密指数的方法



```
1 import libnum
2 a=187134557176277803609960019345398707113865293121166341187786621
3 print(libnum.n2s(a))
```

main x

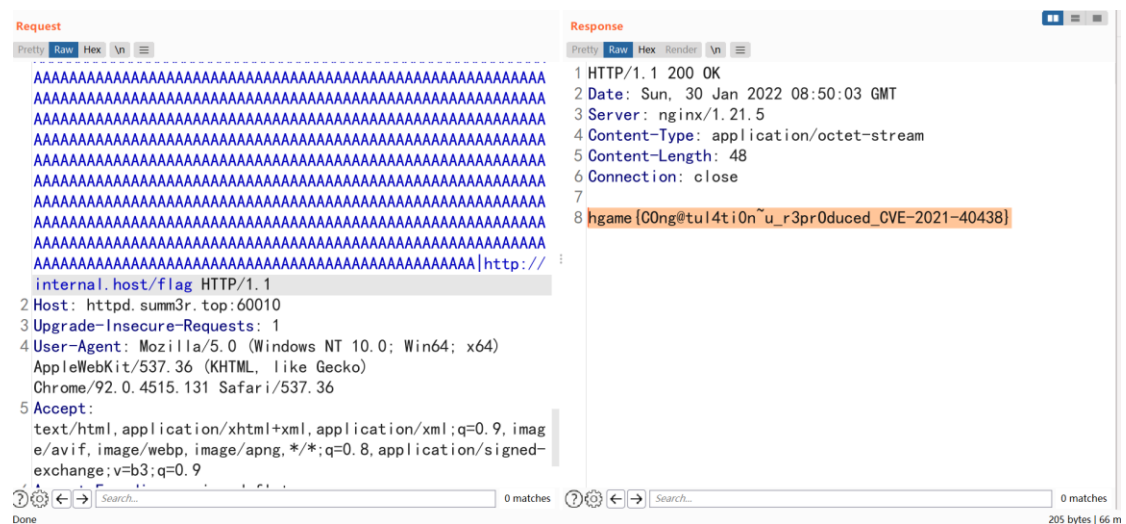
```
D:\python3\python.exe C:/Users/NeroWander/pythonProject/main.py
b'ttACK|METHODS~do@you_KNOW}'
```

Process finished with exit code 0

WEB

Apache!

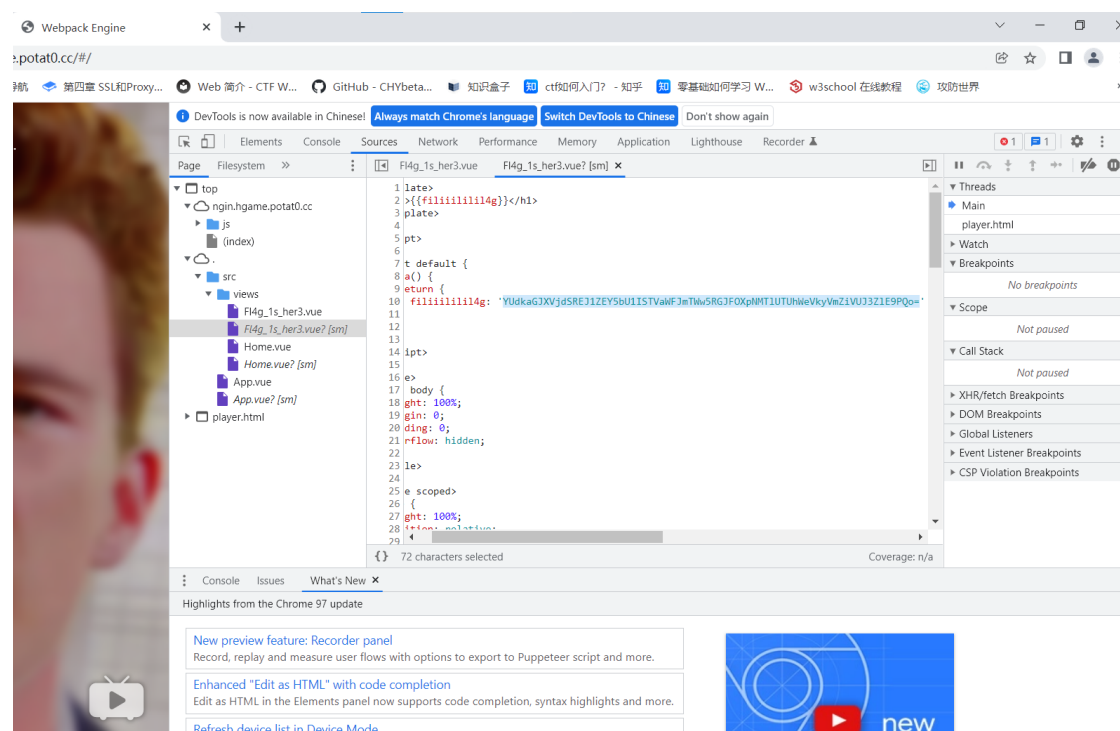
漏洞类型已经给出，利用现有的 payload 求解即可



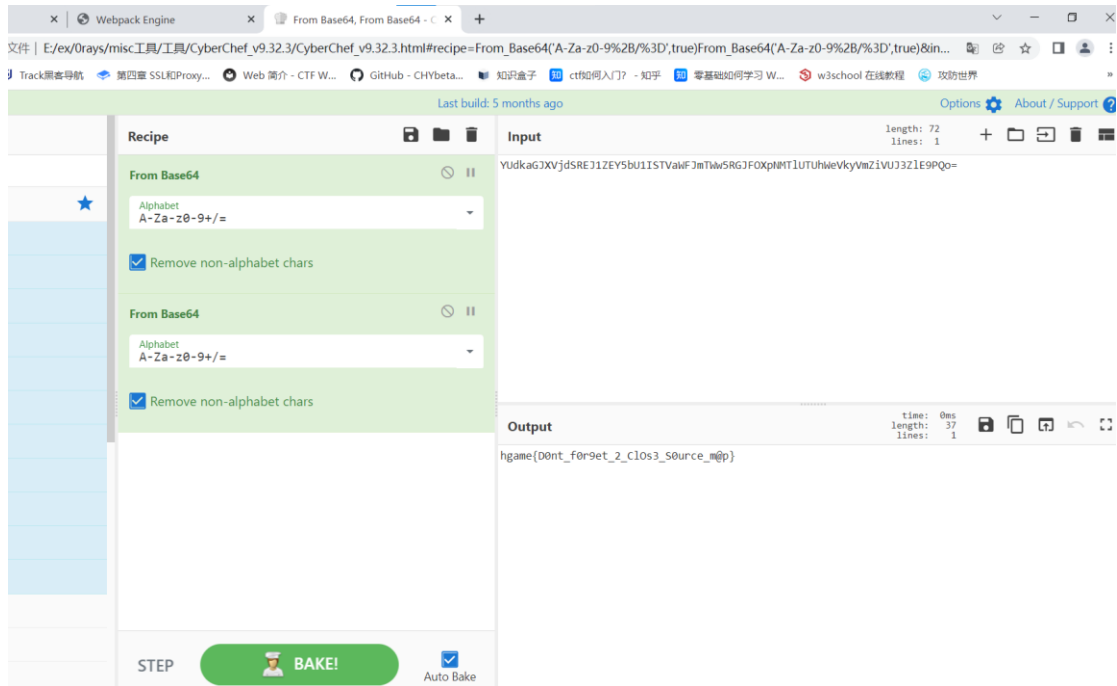
webpack-engine

又被骗了，其实是我心甘情愿的

打开检查，发现 flag 的文件



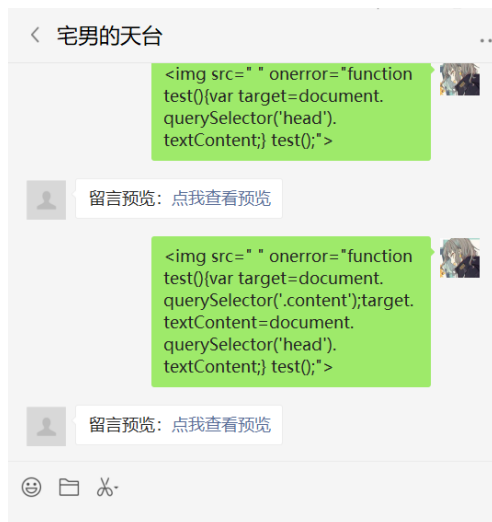
base64 解码 2 次，得到 flag



At0m 的留言板

既然是图片的兼容性测试，那就说明图片方面的留言存在 bug

利用标签，并构造 payload 得到内部信息即可（制造一个不存在的 src 资源，然后触发 onerror 事件）



Pokemon

搜索数据库

搜索表

搜索表的列名

然后根据列名搜索，得到 flag

注意这里的 union select 搜索的字段是 2，可以自行测试，且 information 中的 or 也会被过滤掉，太 baby 辣

一本单词书

两个任务：绕过登录和构造序列化字符

根据 hint 找到 php 源码文件

以下是我的一些理解写在了注释中

```
function decode(string $data): Array { #decode执行unserialize
    $result = [];
    $offset = 0;
    $length = \strlen($data);
    while ($offset < $length) {
        if (!strstr(substr($data, $offset), '|')) { #substr用于返回字符串的一部分（取决于offset的数值），一开始offset
            return []; #strstr查找substr返回的结果中'|'的第一次出现，并返回data的剩余部分
            #!xxx相当于xxx=0，满足if条件的为空字符串，result返回值为空数组
        }
        $pos = strpos($data, '|', $offset); #查找'|'第一次出现的位置，当offset=0的时候从第一个字符开始搜索
        $num = $pos - $offset;
        $varname = substr($data, $offset, $num);
        $offset += $num + 1;
        $dataItem = unserialize(substr($data, $offset)); #因'|'添加在序列化字符串的首部分，则可以使得整个data反序列化
        $result[$varname] = $dataItem;
        $offset += \strlen(serialize($dataItem));
    }
    return $result; #返回整个result数组字符串
}

class Evil {
    public $file; #可以将file的值赋值为"/flag"
    public $flag;

    public function __wakeup() {
        $content = file_get_contents($this->file); #将file文件读入字符串中，也暗示要将"/flag"读入字符串当中，从而得到f
        if (preg_match("/hgame/", $content)) {
            $this->flag = 'hacker!';
        }
        $this->flag = $content;
    }
} #可以根据类来构造序列化的数据从而对该数据进行反序列化处理（好像一般都要反序列化）

function encode($data): string { #encode函数，执行serialize
    $result = ''; #result一开始为空字符串，后续的foreach是对result的填充
    foreach ($data as $k => $v) { #data是一个数组，foreach用于遍历整个数组，k即为key，相当于数组的下标，
        $result .= $k . '|' . serialize($v); #对变量v进行序列化，||将key和value分隔开来，key因此无法序列化
        #个人理解类似于一种截断
    }

    return $result; #遍历完后返回result的值
}

function saveSessionData() {
    $filename = "/tmp/".$_SESSION['unique_key'].'.session'; #$_SESSION['unique_key'] = md5(randomString(8));
    $data = json_decode(file_get_contents("php://input")); #file_get_contents()把整个文件读入一个字符串中，然后进行
    $str = encode($data); #调用encode函数对data变量进行编码
    file_put_contents($filename, $str, FILE_APPEND); #向filename文件中在文件末尾追加str变量的内容
}

if ($_SERVER['REQUEST_METHOD'] == 'POST') { #post提交的方式，一般都能通过if检查
    saveSessionData();
} else {
    echo 'method not allowed';
}
```

```

if (is_numeric($_POST['password'])) {
    die(alert('密码不能设置为纯数字，我妈都知道(△;)'));
} else {
    if ($_POST['password'] == 1080) { //数字绕过，可以在password中添加1080+绕过登录检查
        $_SESSION['username'] = 'admin';
        $_SESSION['unique_key'] = md5(randomString(8));
        header('Location: index.php'); //顺序: login.php->index.php
    } else {
        die(alert('这你都能输错?'));
    }
}
}

```

payload 如下： |O:4:"Evil":2:{s:4:"file";s:5:"/flag";s:4:"flag";N;}

将 payload 填入第一个空白，第二个空白随意填

得到 flag