2019 KCTF 晋级赛Q1 | 第五题点评及解题思路

看雪CTF 看雪学院 1周前

郎骑竹马来，绕床弄青梅。同居长干里，两小无嫌猜。

当李白《长干行》中的“青梅竹马”穿越千年隐藏在代码中，会产生怎样的化学反应呢？接下来，让我们来探寻一下这其中的关联吧！

攻破此题的战队

排名	战队名	破解时间	获取积分
	pizzatql	18214s	100
	Nu1L	19418s	100
	萌新队	35587s	100
4.	tekkens	45512s	100
5.	AceHub	172137s	100
6.	fade-vivi	210167s	100
7.	金左手	214919s	100
8.	雨落星沉	259377s	100
9.	明月照大江1024	277155s	100
10.	1023	346245s	100

题目名称	第五题 青梅竹马
出题战队	ReadPage
题目简介	运行环境Windows，杀毒软件会误报，考察密码学知识。 [公告]2019看雪CTF新赛季！晋级赛每次6-15题，一次性放题，赛期14天。战队必须通过晋级赛，才能参加年底的总决赛！本比赛要求战队独立回答。在题目未结束前，请勿在论坛、QQ群等公共场所讨论试题相关信息，否则视为作弊。欢迎选手加比赛QQ群：8601428
题目下载	crackme_readyu2019.rar
提交答案	请输入注册码（序列号）提交 提交
解析文章	notwolf [原创]2019CTF晋级赛Q1第五题 青梅竹马 DCO [原创]2019看雪CTF_第五题 青梅竹马 HHHso [原创] KCTF 2019 Q1 第五题 欧拉函数

此题仅有 **20** 支队伍攻破此题目，围观人数为 **2206** 人。

出题战队

战队成员：readyu

个人主页：https://bbs.pediy.com/user-23454.htm

个人简介：毕业于中国科学技术大学自动控制专业，从事软件开发多年。在软件保护技术、加密算法方面有一些体验。曾在北京多看科技从事电子阅读加密技术的研究，以及在MIUI从事系统安全方面工作。

看雪CTF crownless 评委 点评

这道题结合了密码学和简单的逆向，需要参赛者对RSA算法、密码学基础十分熟悉。此外，这道题取名“青梅竹马”，意指小素数的组合，十分形象贴切。

题目设计思路

根据规则，crackme SN唯一且为大小写字母+数字。

本题CODE为： PEDIyV9102dVreadyu

正确提示：yes, correct sn!

crackme 取名 青梅竹马 ，意指代 小素数的组合， 字面隐含谜底 "两小无猜" （方程右边为2） 。

设计思路:

（1）原始解：M

考虑100以内的素数，顺序生成一个数列:
2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97
基于这些小素数的乘积，根据RSA的原理拓展出一个题目。

第一项2， 比较特殊，把它排除开来，然后考虑N适当的长度，经过计算，取
N = 3*5*7*11*13*17*19*23*29*31*37*41*43*47*53*59*61*67*71*73 =
203648402996245120753106661735

N的欧拉函数计算如下:

φ(N)=
2*4*6*10*12*16*18*22*28*30*36*40*42*46*52*58*60*66*70*72 =
513385515915890109972480000

然后在后续较大的其余项里，任取一个素数，比如取 e = 83，建立一个幂模方程。

求解大整数M (2 < M < N), 使得M^e = 2 mod N

等价于求解:
e*d = 1 mod φ(N) , 且 1 < d < φ(N)
2^d = M mod N , 且 2 < M < N

由前面的条件可知，

(a) N的素因子不包含2;

(b) 并且 φ(N) 的每个因子项都小于e， 且e是一个素数, 所以e，φ(N)互素, 那么逆元d存在。

所以上面方程有唯一解(d, M)。

计算可得
d = 1/e mod φ(N) = 185561029849116907218968674 7

所以
M = 2^d mod N = 66029406010295430504767654 33
转为16进制的大数：
M = 1555D30F38B0DBCAEC83C0F9

M就是最原始的解。

（2）M转换为有意义的CODE

为了使得M“看起来有意义”，用base64缩短，嵌入信息，伪代码如下：
M = HEX(1555D30F38B0DBCAEC83C0F9)，

M长度为12个字节， 相当于3个int32。

base64table=
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/

M_b64= FVXTDziw28rsg8D5

然而这个注册码并不具有明确意义。

做字母替换，可以手动计算取一个特殊的
custom_base64table=
ABCyVPGHTJKLMNOFQRSIUEWDYZgbc8sfah1jklmnopqret5v0xX9wi234u67dz+/

得到
SN = M_b64(custom_base64table)
= PEDIy9102dreadyu

起初打算把注册码定为

code = PEDIy-9102d-readyu

考虑到规则只允许用数字和字母，用V做分隔符，最终code定为：

CODE = PEDIyV9102dVreadyu

注册码判断的提示信息，也用前面的custom_base64table 编码了。
"bmdeTH8Xb2unTHN5TSVhTQ" // no, wrong sn!!!!
"bWzXZSB3b3JrTHRvTGRvTQ" // more work to do!
"sWE9LCBjb3JXZWNwTHN5TQ" // yes, correct sn!
"c24aZGzlc24n8CB3b3JrTQ" // sn doesn't work!

程序里的custom_base64table简单地异或加密，避免暴露明文，每一项 c[i] ^ 0x50

（3）验证注册码

1. 对code有效字符集和长度作检测, 不通过的，提示错误信息。trim后得到干净的sn。

2. sn 通过custom_base64 decode转换为hex M，也就是一个大整数。

3. 用筛法生成100以内的小素数列表，primes[i]。

4. 计算 N = 3*5*7...*73 （p=79时截止，79=0x4F，恰好是大写字母O的ascii值）。

5. 判断输入 2 =< M <= N，不通过的，提示错误信息。

6. 计算大整数 X = M^e mod N , 其中e=83, 程序里固定。

X转换回int值，最多只能转换32bit。

bits(X) >= 32 bit时，转换为INT_MAX， 提示 sn doesn't work!

当且仅当 bits(X) <32bit, 并且X=2 时，提示 yes, correct sn!

说明：

- 大整数运算采用polarssl-0.10 里的bignum.c/bignum.h，这个库比openssl, gmp, miracl 等常见大整数运算库小巧很多。
- https://tls.mbed.org/download-archive
- 为了使代码尽可能简单，M^e mod N 没有采用bignum里面的mpi_exp_mod()。
- 考虑到e是一个小指数，采用的是power left to right的简单算法。

解题思路

本题解题思路由 **梦游枪手** 提供

打开程序，随便输入点什么，弹出'no,wrong sn!!!"。

在MessageBoxA下断，回溯到402652，用IDA分析，下面的代码我已经分析过并重命名函数了。

```
//省略
GetDlgItemTextA(hDlg, 1000, &String, 3);    // String总是等于0x4F
v38 = GetDlgItemTextA(hDlg, 1001, &input, 255);
sub_402A90(0, &Text);
v39 = 0;
v41 = 0;
if ( v38 < 12 )    // len>12
v41 = 1;
if ( input == 'A' && v30 == 'A' )    // 1和2位不可以为A
++v41;
if ( v31 != 'V' || v32 != 'V' )    // 6和12位等于V
++v41;
for ( i = 0; i < v38; ++i )
{
if ( i != 5 && i != 11 )    // 当前位置不是6和12时
...//这部分代码主要是组合input并且初始化变量，省略
v5 = String;    // v5=0x4F
sub_402889(80, &v13, 0x40u);    // 生成base64table
sub_402270(&v13);    // 写到全局变量
len = base64decode((unsigned __int8 *)&code, &out);// 解码输入字符串
```

```
base64encode((unsigned __int8 *)&out, &encode, len);// 解码
if ( out )
{
if ( !memcmp(&encode, &code, v39) )    // 比较编码后是否相同，相同则进入下一步
{
v7 = check(100, v5, (int)&out, len);
GetText(v7, &Text);    // base64解码后返回
}
}
}
MessageBoxA(hDlg, &Text, &Text, 0);
```

分析代码，可以知道flag是base64编码后在6和12位处插入'V'，在解码时会将其删去。

可以在调试 base64decode或 base64encode 时提取到码表

ABCyVPGHTJKLMNOFQRSIUEWDYZgbc8sfah1jklmnopqret5v0xX9wi234u67dz

看下check的代码

```
int __cdecl check(int a1, int value, int code, int len)
{
signed int v4; // ebx
int *v5; // edi
int prime_t; // [esp+Ch] [ebp-1C0h]
int table[99]; // [esp+10h] [ebp-1BCh]
char int2; // [esp+19Ch] [ebp-30h]
char codeint; // [esp+1A8h] [ebp-24h]
char result; // [esp+1B4h] [ebp-18h]
char big_m; // [esp+1C0h] [ebp-Ch]
int prime; // [esp+1D4h] [ebp+8h]
int ret; // [esp+1E0h] [ebp+14h]

prime_t = 0;
memset(table, 0, sizeof(table));
prime = createprime(a1, &prime_t);    // 生成质数数组
big_init(&result);
big_init(&int2);
big_setvalue((int)&result, 0);
big_setvalue((int)&codeint, 0);
v4 = 1;
big_setvalue((int)&big_m, 1);
```

```
big_setvalue((int)&int2,  prime_t);    // 第一位是2
big_fromint((int)&codeint, code, len);    // 将code转换为bignum类型
ret = big_toint((int)&result);
if  ( prime >  1  )
{
v5 = table;    // table的值是prime的指针+1
while  ( *v5 != value )    // value=0x4F
{
if  ( *v5 )
sub_401B0A((int)&big_m, (int)&big_m, *v5);// 乘以table所有成员，结果在big_m，结果是0x41C
++v4;
++v5;
if  ( v4 >= prime )
goto  LABEL_9;
}
prime = table[v4];    // 取得value所在位置后一位的成员，在这个程序里一直等于0x53
}
LABEL_9:
if  ( big_cmp(&codeint, &int2) >=  0  && big_cmp(&codeint, &big_m) <=  0  )
{
codevaild((int)&result, (int)&codeint, prime, (int)&big_m);
if  ( big_cmp(&result, &int2) >=  0  && big_cmp(&result, &big_m) <=  0  )// result为2
ret = big_toint((int)&result);
}
free((int)&result);
free((int)&int2);
return  ret;
}
```

看过GetText就知道，这个函数的返回值必须为2

```
if  ( a1 )
{
if  ( a1 ==  1  )
{
v2 = aBwzxzsb3b3jrth;    // more work to do!
}
else  if  ( a1 ==  2  )
{
v2 = aSwe9lcbjb3jxzw;    // yes, correct sn!
}
else
{
```

```
v2 = aC24azgzlc24n8c;    // sn doesn't work!
}
}
else
{
v2 = aBmdeth8xb2unth;    // no, wrong sn!!!!
}
```

我们继续跟进codevaild

```
int __cdecl codevaild(int a1, int big_code, unsigned int a3, int big_m)
{
unsigned int i; // ebx
char ret; // [esp+4h] [ebp-24h]
char mulret; // [esp+10h] [ebp-18h]
char tmp; // [esp+1Ch] [ebp-Ch]

big_init(&ret);
mod(&tmp, big_code, (_DWORD *)big_m);    // 取模，结果给tmp
big_setvalue((int)&ret, 1);
for ( i = a3; i; i >>= 1 )    // i=0x53
{
if ( i & 1 )
{
mul_0(&mulret, &ret, &tmp);    // 相乘
mod(&ret, (int)&mulret, (_DWORD *)big_m);
}
mul_0(&mulret, &tmp, &tmp);
mod(&tmp, (int)&mulret, (_DWORD *)big_m);
}
copy((_DWORD *)a1, (int)&ret);
free((int)&ret);
return 0;
}
```

将过程列成数学表达式

```
ret1=(code*ret)%0x41CD66ACC237B22681A18067 //code^1
code1=(code*code)%0x41CD66ACC237B22681A18067 //2
ret2=(code1*ret1)%0x41CD66ACC237B22681A18067 //3
code2=(code1*code1)%0x41CD66ACC237B22681A18067 //4
code3=(code2*code2)%0x41CD66ACC237B22681A18067 //8
code4=(code3*code3)%0x41CD66ACC237B22681A18067 //16
```

```
ret5=(code4*ret2)%0x41CD66ACC237B22681A18067 //19
code5=(code4*code4)%0x41CD66ACC237B22681A18067 //32
code6=(code5*code5)%0x41CD66ACC237B22681A18067 //64
ret7=(code6*ret5)%0x41CD66ACC237B22681A18067 //83
ret7=2
```

可以把表达式整理成(flag^83) = 2
mod 0x41CD66ACC237B22681A18067

求解该高次剩余问题就能得到flag，下面用P代称
0x41CD66ACC237B22681A18067

一般高次剩余问题中P是质数，但是这个地方不同，P是合数，P是check函数的质数数组{0x03,0x05,0x07,0x0B,0x0D,0x11,0x13,0x17,0x1D,0x1F,0x25,0x29,0x2B,0x2F,0x35,0x3B,0x3D,0x43,0x47,0x49}相乘得到的，所以需要先质数分解P，求解flag1^83 = 2 mod 3，flag2^83 = 2 mod 5，flagn^83 = 2 mod Pn ，再用中国剩余定理合并flagn。

求解高次剩余的文章和代码网上有很多，后面我会附上链接，可以解得flagn={2,3,4,7,7,8,13,4,15,4,5,5,22,32,5,10,17,63,38,32}

接下来就是用中国剩余定理合并，网上找的代码大多数都是int64，没办法直接用在这里，所以用python改了改：

```
def egcd(a, b):
## 扩展欧几里得
if 0 == b:
return 1, 0, a
x, y, q = egcd(b, a % b)
x, y = y, (x - a // b * y)
return x, y, q
def CRT(r1,m1,r2,m2):
## 中国剩余定理
z=m1*m2
t=egcd(m1,m2)
x=t[0]
y=t[1]
return ((x%z*(r2-r1)%z*m1+r1)%z+z)%z
def dfs(p,r,mo):
k=[2,3,4,7,7,8,13,4,15,4,5,5,22,32,5,10,17,63,38,32]
m=[0x03,0x05,0x07,0x0B,0x0D,0x11,0x13,0x17,0x1D,0x1F,0x25,0x29,0x2B,0x2F,0x35,0x3B,0x3D
if(p+1==len(k)):
print('code :',hex(r),'mod P = 2')
```

```
print('P:',hex(mo))
return
_mo=m[p+1]
dfs(p+1,CRT(r,mo,k[p+1],_mo),mo*_mo)
if __name__=='__main__':
dfs(0,2,3)#k[0],m[0]<font color="#000000" face=""><span style="font-size:15px;">
</span></font>
```

输出：

('code :', '0x1555d30f38b0dbcaec83c0f9L', 'mod P = 2')
('P:', '0x41cd66acc237b22681a18067L')

0x1555d30f38b0dbcaec83c0f9 这个就是结果了，用提取的码表base64编码后得到"PEDIy9102dreadyu"。在6和12位加上'V'，得到最终flag:
PEDIyV9102dVreadyu

搞定收工。

参考文章和代码：

- 无代码 离散对数（关于方程x^A=B(mod C)的解)
- 有代码[51nod 1223] x^A mod B问题 - bsgs、原根、中国剩余定理、二进制分组
- 有代码 数论算法 剩余系相关 学习笔记
- 有代码 hdu 3930 X^N=a(mod) p 求X

看雪CTF晋级赛Q1 题解列表

1、2019KCTF 晋级赛Q1 | 第一题点评及解题思路

2、2019KCTF 晋级赛Q1 | 第二题点评及解题思路

3、2019 KCTF 晋级赛Q1 | 第三题点评及解题思路

4、2019KCTF 晋级赛Q1 | 第四题点评及解题思路

- End -

公众号ID：ikanxue

官方微博：看雪安全

商务合作：wsc@kanxue.com

戳原文，查看更多精彩writeup！

阅读原文