Crypto

夺宝大冒险1

先看源码 ~~将里边不讲人话的命名修整一下~~

大概可以列出这个模板

```
class TX:
    A = int.from_bytes(os.urandom(8),'big')
    B = int.from_bytes(os.urandom(8),'big')
    C = int.from_bytes(os.urandom(8),'big')
    def __init__(self, seed):
        self.state = seed
    def next(self):
        self.state = (self.state * self.A + self.B) % self.C
        return self.state
```

这个好像是一个随机数生成器啊

百度一下发现这个是一个线性同余伪随机数生成器`LCG` `A`就是乘数 `B`就是增量 `C`就是模量

然后题目的要求就是让要攻击这个LCG 分别做到：
1.已知`A` `C`求`B`
2.已知`C` 求`A` `B`
3.求 `C`

先贴一个地址 防止我讲翻车

https://www.dazhuanlan.com/2019/10/18/5da8e637b51cb/

这个算法的加密递推公式是这样的

$$X_{i+1} = (A * X_i + B) \mod C$$
$$X_{i+1} + k_i * C = A * X_i + B$$

所以就只需要两个连续的输出，然后解关于`B` `k`的不定方程就好了

```
def crack_unknown_B(x, C , A):
    B = (x[1] - x[0] * A ) % C
    return A , B , C
```

在上一个结果的基础上 对相邻的两式子做差 得到

$$(X_{i+1} - X_i) + k' * C = A * (X_i - X_{i-1})$$
$$Y_i = A * Y_{i-1} \mod C$$
$$Y_i + k'' * C = A * Y_{i-1}$$

这样就能求出`A`了 然后再借用上边的函数 就可以解出`B`了

```
def crack_unknown_A( X , C ):
    A = (X[2] - X[1]) * libnum.xgcd(X[1] - X[0] , c)[0] % c
    return crack_unknown_B( X , C , A)
```

但是如果`c`未知呢 就可以左乘右 右乘左 构造一个神奇的方程

$$
\begin{aligned}
A * Y_{i+2} * Y_i &= A * Y_{i+1}^2 \mod C \\
A * (Y_{i+2} * Y_i - Y_{i+1}^2) &\mod C = 0 \\
Y_{i+2} * Y_i - Y_{i+1}^2 &\mod C = 0
\end{aligned}
$$

所以只要找到他们的公倍数就好了

```
def crack_unknown_C(X):
    D = [s1 - s0 for s0, s1 in zip( X , X[1:])]
    Z = [t2 * t0 - t1 * t1 for t0, t1, t2 in zip(diffs, diffs[1:], diffs[2:])]
    C = abs(libnum.reduce(libnum.gcd, Z))
    return crack_unknown_A( X, C )
```

还有一点 就是这道题是远程调试的 ~~除非你手速惊人~~ 还是用下`Pwntools`吧

夺宝大冒险2

看到class里边的`random` 推测这个也是一个随机数生成器

百度下加密算法 这个生成器好像叫做`LFSR` ~~好像没啥子用~~

看看代码 主要作妖的还是这两段

```
def next(self):
    nextdata = (self.init << 1) & self.lengthmask
    i = self.init & self.mask & self.lengthmask
    output = 0
    while i != 0:
        output ^= (i & 1)
        i = i >> 1
    nextdata ^= output
    self.init = nextdata
    return output

def random(self, nbit):
    output = 0
    for _ in range(nbit):
        output <<= 1
        output |= self.next()
    return output
```

`next`负责生成下一个随机数 生成原理是 先将`init`里的值左移一位并将长度截取成`mask`的长度`i`而且最后一位是0 然后就是这个神奇的`output`我不知道他是用什么神奇的算法算出来的 但是他就是被直接塞到了`init`的屁股后边

`random`就很舒适了 就是把`next`执行`nbit`次，并把后`nbit`位打印出来

所以只要我们执行一定次数 他就不知不觉的把seed替换了 而且又知道随机数生成的算法 就等于控制了随机数的生成

但是要如何让他不停输出呢 说明要丢一个一定错误的数 最简单的方法是塞一个负数

而且品一品需要试错的次数 只用10次 而我们的容错率是20% 那就好了

```
sh = remote("182.92.108.71", 30607)
for i in range(10):
    sh.sendline(str(-1))
data = sh.recvlines(20)[1::2]
s = ""
for i in range(10):
    s += bin(int(data[i][28:]))[2:].zfill(4)
init = int(s, 2)
prng = LXFIQNN(init, 0b10110010100010100001000010001111011110101, 40)
for _ in range(90):
    sh.sendline(str(prng.random(4)))
print(sh.recvlines(181)[-1])
```

MISC

Akira之瞳-1

先下载文件 发现这个压缩包的压缩率特别的高

然后得到一个RAW文件~~然后我就习惯性的把他拖到了 `Ps` - `Lr` 里，然后得到一个巨大的条形码（地铁老人看手机）~~

后来百度了一下 发现这个文件可能是内存文件 那么说明这个题应该是内存取证~~可能只是因为我比较弱鸡才会搞错~~

然后上 kali 用 `volatility` 先 `imageinfo` 康康这个是啥系统 是Win7SP1x64

然后`pslist`康康进程 发现了一个叫`important_work.exe`~~然后我用GIMP调一下午参数妄想看到进程截图 最后毫无疑问的白给~~

然这里我去问了下出题人 ~~他告诉我我方向完全错了~~ 并让我康康这个程序打开了什么

那就把他给`memdump`下来 然后`foremost`（`binwalk`好像也是可以的~~但是分离出的文件乱的离谱~~）分析一下 发现里边有一个压缩包~~然后我又双叒叕死昨一下阴间的环境问题上~~

压缩包里有一个压缩包和几个~~意义不明的~~图片 打开压缩包 毫无疑问是要密码 但是压缩包下边有个简介 `Password is sha256(login_password)`

好家伙我还要去找这个电脑的密码

还是用`volatility` 先`hivelist`获得`SYSTEM`和`SAM`位置 然后`hashdump`下来

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c08
9c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Genga03:1001:aad3b435b51404eeaad3b435b51404ee:84b0d9c9f830238933e7131d60ac6436::
:
```

然后密码是MD5编码的 解密完再sha256编码一下 这个压缩包就解压出来了 压缩包里有两个看起来一模一样的图片

我一开始以为是双图片隐写 拖到`Stegsolve`一阵异或分析 啥都没发现

再百度一下 发现这个应该是盲水印 ~~傅里叶变化昨也不懂啊 所以开始GitHub抄脚本~~

~~然后又是和环境斗智斗勇了不知道多久~~这个脚本跑成功了（python3和python2的使用方法会有点不大一样，而且搞不好结果也不大一样~~别问我怎么知道会解析出一坨雪花的~~）

最后就是flag了~~但是还是要和字体斗智斗勇半天~~