Sharing is Caring

Yaman Yağız Taşbağ
@yamantasbagv2

November 11, 2019

1 Introduction

"Sharing is caring" was a Crypto 500 task from CSAW 2019 Finals. We, as Bilkent University Security Club *Exploit Studio*, attended the finals in MENA (Middle East and North Africa) division and ended up in the 4th place. This tasks generates a random secret each time it is ran and if the challenger is able to find this secret it gives the flag. We were able to get the second blood on this task in the world.

2 Reversing

We are given a a python script that is running on the server as the challenge which can be found here. In this section we will try to understand what the task does mathematically.

```
class Num:
    def __init__(self, _a, _b):
        self.a = _a
        self.b = _b
    def __add__(self, o):
        return Num(self.a + o.a, self.b + o.b)
    def __mul__(self, o):
        return Num(self.a * o.a - self.b * o.b, self.a * o.b + self
    .b * o.a)
    def __mod__(self, n):
        return Num(self.a % n, self.b % n)
    def powmod(self, n, p):
        if n == 0:
            return Num(1, 0)
        x = self.powmod(n / 2, p)
        if n % 2 == 1:
            return (x * x * self) % p
        else:
            return (x * x) % p
```

Figure 1: Shortened Num Class

In this class we have a integer tuple which are defined with following operations:

- $(a, b) + (c, d) \rightarrow (a + c, b + d)$
- $(a, b) * (c, d) \rightarrow (a * c - b * d, a * d + b * c)$
- $(a, b) \bmod n \rightarrow (a \bmod n, b \bmod n)$

The multiplication is equivalent to the complex multiplication hence the Num class is an element of $\mathbb{F}_p \times \mathbb{F}_p$ for some prime p.

```
def mod_poly(P, p, x):
  ...

def make_poly(s, p):
  ...

def E(P, p, x):
    s = Num(0, 0)
    for e in P.keys():
        s += x.powmod(e, p) * Num(P[e], 0)
        s %= p
    return s
```

Figure 2: mod_poly, make_poly and E

make_poly converts string to a polynomial such as: "1xˆ1 - 1xˆ2" to the list "[(1, 1), (-1, 2)]" and mod_poly will take modulo p of the coefficients. E will calculate the polynomial P at x under modulo p.

The challenge works as follows:

1. Generate and publish prime p
2. Take a polynomial from the challenger. (with constraints)
3. Take threshold value (Shamir Secret Sharing Scheme)
4. Generate secret
5. Ask for the number of secret shares
6. Calculate and publish secret shares.
7. Ask for the secret from the challenger.

The user given polynomial has to allow the following constraints:

- It cannot have a non zero coefficient for each term where the degree divides p ie. $0, p, 2p...$

- Coefficient of the x cannot be zero.
- Degree of the polynomial has to be greater than the threshold, which is at least 9000.

The secret shares are calculated as follows:

```
print "here are your shares"
for i in xrange(1, n + 1):
    # hard mode
    y = (i * secret) % p
    share = E(P, p, Num((13 * y) % p, (17 * y) % p))
    print "(%d,%s)" % (i, share)
```

Figure 3: Secret share generation

3 Solution

The key observation is that the Num class that we are working with is an element of a field of size p^2. We can extend the Fermat's little theorem to solve this task. The Fermat's Little Theorem states that:

$$a^{p-1} \equiv 1 \pmod{p} \tag{1}$$

This can be thought as the Lagrange's Theorem for $\langle a \rangle$ in $\mathbb{Z}_p^*$ ie. the order of $\langle a \rangle$ has to divide $p-1$. We can apply this to our finite field of size p^2. Hence $|\langle a \rangle|$ divides $p^2 - 1$.

Another crucial point to notice is *make_poly* does not reduce the polynomial in $x^{p^2-1} - 1$. Hence if we set out polynomial to $x - x^2 + x^{p^2+1}$, $-x^2$ and x^{p^2+1} will cancel each other out and will result in x. With only a single share we can calculate the secret value.

```
from Crypto.Util.number import inverse
p = int(input("p: "))
P = f"1x^1 - 1x^2 + 1x^{p*p + 1}"
print(P)

x = int(input("x: "))
y = int(input("y: "))

s1 = x * inverse(13, p) % p
s2 = y * inverse(17, p) % p
assert s1 == s2
print("Secret:", s1)
```

Figure 4: Our solution

This gave us the flag "flag{i_cant_believe_someone_else_found_a_cheese}"

4 Conclusion

I had fun solving this challenge and to be able to get the second blood worldwide made me real excited and good about myself. Though, I don't believe this challenge was worth 500 points but it was definitely a learning experience. As always you can contact me over twitter for feedback.