رامتین احسانی - ۹۷۵۲۱۰۱۸

تمرین چهارم – امنیت سیستم های کامپیوتری

۱) میدانیم که در الگوریتم RSA باید رابطه زیر برای e برقرار باشد:

$$1 = \gcd(e, \phi(n)) = \gcd(e, (p-1)(q-1)).$$

میدانیم که p و q دو عدد اول متفاوت از هم هستند. برای مثال 13 و 7.
وقتی در رابطه بالا این اعداد را قرار میدهیم حداقل یکی از آن ها عدد زوج تشکیل میدهد و برای اینکه e و (p-1)(q-1) نسبت به هم اول باشند، باید e عددی فرد باشد.
پس بهتر است پارامتر e عددی فرد باشد.

۲) اعداد فرما به اعدادی میگویند که در فرمت زیر قرار دارند:

$$F_n = 2^{2^n} + 1,$$

مثال هایی از اعداد فرما:
3, 5, 17, 257, 65537

از اعداد فرما در RSA برای تولید کلید عمومی e استفاده میکنند.

۳) الگوریتم های زیادی برای عملیات به توان رسانی در محاسبات پیمانه ای وجود دارند که در این قسمت به چند تا از این الگوریتم ها اشاره میکنم.

$$a^b \pmod n$$

الگوریتم اول که ساده ترین الگوریتم است، محاسبه ساده این عملیات است یعنی a را به توان b برسانیم و mod را حساب کنیم که اینکار با بزرگتر شدن اعداد خیلی الگوریتم را آهسته میکند.

الگوریتم دوم به صورت زیر عمل میکند:

```
def modexp_mul(a, b, n):
    r = 1
    for i in xrange(b):
        r = r * a % n
    return r
```

این الگوریتم از الگوریتم اول بهتر عمل میکند.

روش دیگری که برای این محاسبات وجود دارد روش repeated squaring است:

```
def modexp_rl(a, b, n):
    r = 1
    while 1:
        if b % 2 == 1:
            r = r * a % n
        b /= 2
        if b == 0:
            break
        a = a * a % n

    return r
```

روش چهارم روش The k-ary LR method است:

```
def modexp_lr_k_ary(a, b, n, k=5):
    """ Compute a ** b (mod n)

        K-ary LR method, with a customizable 'k'.
    """
    base = 2 << (k - 1)

    # Precompute the table of exponents
    table = [1] * base
    for i in xrange(1, base):
        table[i] = table[i - 1] * a % n

    # Just like the binary LR method, just with a
    # different base
    #
    r = 1
    for digit in reversed(_digits_of_n(b, base)):
        for i in xrange(k):
            r = r * r % n
        if digit:
            r = r * table[digit] % n

    return r
```

و روش آخر هم طبق معادله زیر عمل میکند:

```
(ab) mod p = ( (a mod p) (b mod p) ) mod p

For example a = 50,  b = 100, p = 13
50  mod 13  = 11
100 mod 13  = 9

(50 * 100) mod 13 = ( (50 mod 13) * (100 mod 13) ) mod 13
or (5000) mod 13 = ( 11 * 9 ) mod 13
or 8 = 8
```

کد:

```python
def power(x, y, p) :
    res = 1      # Initialize result

    # Update x if it is more
    # than or equal to p
    x = x % p

    if (x == 0) :
        return 0

    while (y > 0) :

        # If y is odd, multiply
        # x with result
        if ((y & 1) == 1) :
            res = (res * x) % p

        # y must be even now
        y = y >> 1       # y = y/2
        x = (x * x) % p

    return res
```

۴) الگوریتم RSA 1024 معادل با یک الگوریتم رمزنگاری بلوکی 80 بیتی است.
الگوریتم RSA 2048 معادل با یک الگوریتم رمزنگاری بلوکی 112 بیتی است.