Feltőrések, ECN-felkészítő, 2013

Input: M.IN

Output: Standard output

Nagyon egyszerű, de nem teljesen biztonságos titkosítás az, amikor egy nyílt-szöveg bytejait úgy titkosítjuk, hogy a nyílt-szöveg byte-jait egy véletlenszerűen generált byte-szekvencia byte-jaival mod 2 szerint összeadjuk, ahol a véletlenszerű $b_0, b_1, \ldots b_l$ byteok előállítását a következőképpen végezzük:

- veszünk két prímszámot: p-t és q-t, ahol $p = q = 3 \pmod 4$, és egy tetszőleges $r \in \{2, \ldots, n-1\}$ értéket, ahol $n = p \cdot q$,
- meghatározzuk: $r_0 = r^2 \pmod n$,
- a byte-szekvencia: $b_i = r_i \pmod{256}$, $r_{i+1} = r_i^2 \pmod n$, $i \in \{0, 1, \ldots, l\}$.

A visszafejtésnél a p, q és r_{l+1} értékek állnak a rendelkezésünkre, melyek alapján meghatározható az r_0 érték. Ez után hasonlóképpen járunk el: újra generáljuk a byte-szekvenciát melyet mod 2 szerint hozzáadunk a rejtjelezett szöveghez.

Írjunk programot, mely adott titkosított byte-szekvenciára meghatározza a nyílt-szöveget.

Az M.IN állomány több tesztesetet tartalmaz. Első sora a tesztesetek számát jelzi. Minden tesztesetnél meg vannak adva, a titkosításhoz használt n és r_{l+1} értékek, majd a nyílt-szöveg titkosított bytejai. Egy üres sor után a következő teszteset adatai következnek.

A program az M.IN állományban található összes tesztesetre meg kell határozza a nyílt-szöveget. Minden visszafejtett nyílt-szöveg után tegyünk egy üres sort.

Példa:

M.IN
3 3083 3571 5610455 11 6 98 32 -117 -81 -60 124 41 -109 89 -31 4007 2503 3195750 24 -78 104 -32 -112 -5 -104 107 -56 -5 -117 -107 -55 46 -90 -107 -8 -70 -11 84 -36 24 77 -68 121 3163 2347 7209516 66 42 -44 79 117 30 -16 66 -84 36 -74 91 -76 47 24 7 -106 -87 72 -96 -112 11 37 85 -70 76 36 -107 -110 -71 -100 -87 127 94 23 62 -75 51 -120 62 -64 -117 87 -38 -81 -57 -72 3 79 84 -73 17 -81 68 -16 57 34 -4 99 -3 24 59 -114 91 -65 -106 56

Standard output
Hello vilag Hargita Programozó Tábor Sapientia EMTE, Marosvásárhely, Muszaki és Humántudományok Tanszék

Algoritmusok

- Moduláris hatványozás
- Kiterjesztett Euklideszi algoritmus
- Multiplikatív inverz
- Kínai maradék-tétel

A kiterjesztett Euklideszi algoritmus

```
XGCD(d, x, y, a, b)
    x0 = 1, x1 = 0;
    y0 = 0, y1 = 1;
    while ( b != 0 )
        r = a % b;
        q = a / b;
        a = b;
        b = r;
        xx = x1; yy = y1;
        x1 = x0 - q * x1;
        y1 = y0 - q * y1;
        x0 = xx; y0 = yy;
    d = a; x = x0; y = y0;
```

az "a" inverze "mod b" szerint "x" lesz, es ellenorizni hogy ha x ¡ 0, akkor x += b) a "b" inverze "mod a" szerint "y" lesz, es ellenorizni, hogy ha y ¡ 0, akkor y += a)

Az r_0 meghatározása

A p, q és r_{l+1} alapján, a Kínai maradék-tétellel megoldjuk a következő kongruencia rendszert:

$$\begin{aligned} r_0 &= r_{l+1}^{a_1} \pmod{p}, \\ r_0 &= r_{l+1}^{a_2} \pmod{q}, \end{aligned}$$

ahol

$$\begin{aligned} a_1 &= ((p+1)/4)^{l+1} \pmod{p-1}, \\ a_2 &= ((q+1)/4)^{l+1} \pmod{q-1}. \end{aligned}$$