

Kryptografins grunder: hemtal 3

Erik Tjernelund*

750519-0137

d98-etj

21 maj 2002

1 Pseudoslumptal 20p

2 Kunskapslösa protokoll 15p

Uppgiften är att lösa problem 13.1 i den äldre upplagan av Stinson. Man ska bevisa att protokollet är komplett, sunt och visa varför det inte är kunskapslost.

2.1 Kompletta protokoll

Om protokollet ska vara komplett måste den som utmanar (U) *alltid* acceptera svaret från svararen (S) om $x \notin \text{QR}(n)$, alltså om x *inte* är en kvadratisk rest modulo n .

Om vi antar att $x \notin \text{QR}(n)$ finns det två alternativ:

1. Om U väljer $i = 0$ blir $z = y \bmod n = v^2 \bmod n$ och z är alltså en kvadratisk rest. S svarar med $j = 0 = i$ och U accepterar.
2. Om U väljer $i = 1$ blir $z = x \cdot y \bmod n$. Vet att produkten av en kvadratisk rest och en kvadratisk icke-rest är en kvadratisk icke-rest och alltså är $z \notin \text{QR}(n)$. S svarar med $j = 1$ och då $i = 1 = j$ accepterar U.

U accepterar alltså alltid när x inte är en kvadratisk rest och protokollet är därmed komplett.

2.2 Sundhet

För att protokollet ska vara sunt ska sannolikheten för att U accepterar när $x \in \text{QR}(n)$ vara mycket liten. Om x är en kvadratisk rest kommer det z , som U skickar till S, alltid också vara en kvadratisk rest. Sannolikheten att U avslöjar S är därför $\frac{1}{2}$ vid varje utmaning.

Eftersom U utmanar S $\log_2 n$ gånger är sannolikheten att U accepterar $\frac{1}{2^{\log_2 n}} = \frac{1}{n}$. Det är alltså en väldigt liten sannolikhet att detta ska hända och därmed är protokollet sunt.

*I samarbete med Emanuel Viklund (d98-evi)

2.3 Inte kunskapslöst

Protokollet är inte kunskapslöst eftersom man utifrån konversationen mellan U och S kan avgöra om $z \in \text{QR}(n)$ eller inte.

3 Diskret logaritm 15p

Uppgiften var att hitta x så att $2^x = s \pmod p$, där $p = 6211382198459$ (ett primtal och 2 är en generator till \mathbb{Z}_p^*) och s är mitt eget personnummer. Ekvationen blir alltså:

$$2^x = 7505190137 \pmod{6211382198459}$$

Jag valde att implementera Shanks algoritm för att hitta x . Skrev två små java program som beräknade de två listorna. Dessa blev i storleksordningen ~ 60 megabyte. Sorterade dem med *sort* i unix och skrev ett litet program i perl som hittade en rad som innehöll samma tal.

Fann således de två talen $i = 70820$ och $j = 1169498$ och då blir $x = 176503376798$.

För att övertyga sig själv och andra om att detta stämde användes Maple:

```
> p := 6211382198459;
> m := ceil(sqrt(p));
> i := 70820;
> j := 1169498;
> x := m*i+j mod p;
> 2 &^ 176503376798 mod p;
```

```
p := 6211382198459
m := 2492265
i := 70820
j := 1169498
x := 176503376798
7505190137
```

Har alltså hittat ett korrekt x .

4 Hashkollisioner och födelsedagsparadoxen 10p

Uppgiften är att beräkna hur många hashvärden man är tvungen att tillverka för att få tre som hashar till samma värde.

Till att börja med så antar vi att hashfunktionen är slumpmässig, dvs. att det är lika stor sannolikhet att få varje hashvärde. Det finns $l = 2^n$ olika bitsträngar av längd n . Om man tar tre sådana bitsträngar på måfå, så kommer man att få $2^{n^3} = 2^{3n}$ möjliga tripplar. Av dessa kommer 2^n bestå av tre identiska bitsträngar. Alltså är sannolikheten att ta en trippel med identiska bitsträngar:

$$P(\text{tre lika}) = \frac{2^n}{2^{3n}} = \frac{1}{2^{2n}}$$

Vi definierar en stokastisk indikatorvariabel \mathbb{X}_{ijk} för varje trippel (x_i, x_j, x_k) där $1 \leq i < j < k \leq l$:

$$\mathbb{X}_{ijk} = \begin{cases} 1 & \text{om } x_i = x_j = x_k \\ 0 & \text{annars} \end{cases}$$

Väntevärdet blir:

$$E(\mathbb{X}_{ijk}) = P(\text{tre lika}) = \frac{1}{2^{2n}}$$

Inför den stokastiska variabeln \mathbb{Z} enligt följande:

$$\mathbb{Z} = \sum_{i=0}^l \sum_{j=i+1}^l \sum_{k=j+1}^l \mathbb{X}_{ijk}$$

Vi vill att väntevärdet ska vara 1 och då får vi följande ekvation:

$$E(\mathbb{Z}) = \sum_{i=0}^l \sum_{j=i+1}^l \sum_{k=j+1}^l E(\mathbb{X}_{ijk}) = \binom{l}{3} \frac{1}{2^{2n}} = \frac{l(l-1)(l-2)}{6 \cdot 2^{2n}} \approx 1$$

Löses ekvationen $l^3 - 3l^2 + 2l - 6 \cdot 2^{2n} = 0$ för l så kan man beräkna antalet strängar man måste ta innan det blir en förväntad kollision. Ekvationen har en reell rot då $n \geq 1$ som fås efter lite räkningar:

$$\begin{aligned} l &= \sqrt[3]{3 \cdot 2^{2n} + 3\sqrt{2^{4n} - 3}} + \sqrt[3]{3 \cdot 2^{2n} - 3\sqrt{2^{4n} - 3}} \approx \\ &\approx \sqrt[3]{2^{2n} + \sqrt{2^{4n}}} + \sqrt[3]{2^{2n} - \sqrt{2^{4n}}} = \\ &= \sqrt[3]{2^{2n} + 2^{2n}} + \sqrt[3]{2^{2n} - \sqrt{2^{4n}}} = \\ &= \sqrt[3]{2^{2n}} = 2^{\frac{2n}{3}} \end{aligned}$$

5 Delad hemlighet 20p

6 Företagssäkerhet 20p