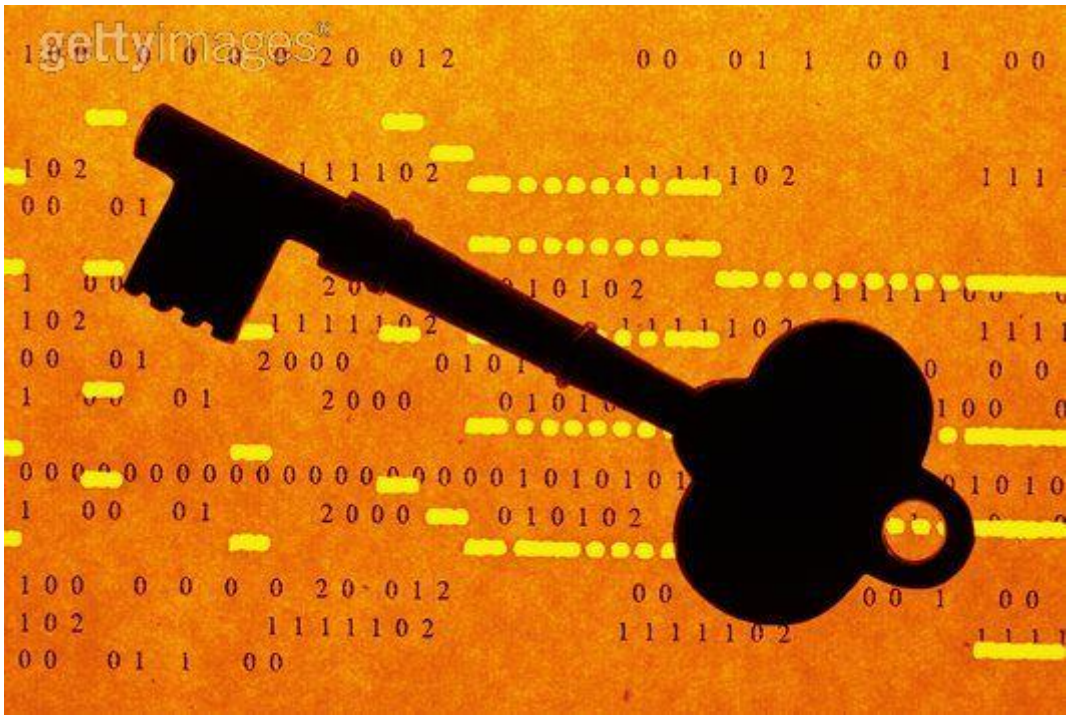
KRYPTERING PÅ TI-89



04-10-2007

Supplerende note til kryptologi

Dette er en supplerende note til Mikkel Kamstrup Erlandsens note "Introduktion til kryptologi". Den indeholder en vejledning til brug af TI-89 i kryptering og dekryptering af beskeder med RSA-kryptering.

Af Jakob Blaavand

Kryptering på TI-89

SUPPLERENDE NOTE TIL KRYPTOLOGI

I Noten "Introduktion til Kryptologi" gives et eksempel på, hvordan man kan krypterer en simpel besked med RSA. Der er imidlertid flere problemer med dette eksempel i forhold til brug på en TI-89, og derfor denne note.

- Først og fremmest bruges for store primtal til at kryptere med. Det betyder, at når klarteksten er oversat til tal, og blokparene er dannet, bliver potens af tallet alt for stort til, at TI-89 kan udregne restklassen. Denne note giver et andet eksempel, der bygger på mindre primtal og mindre blokpar. Derfor er selve eksemplet ikke praktisk anvendelig, men giver et teoretisk eksempel.
- Derudover er notens klartekst valgt meget pænt. Den simple tegntabel, starter nummereringen af bogstaverne med 0, så hvert bogstav repræsenterer en restklasse i Z_{29}. Det er meget pænt, men det giver visse problemer, der ikke redegøres for i noten. Det ser vi nærmere på her, og giver en tegntabel, der løser problemerne.

Ny tegntabel

Vi starter med at se på problemerne med den simple tegntabel.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Æ	Ø	Å
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28

Da den underliggende pointe med RSA-kryptering er, at tage et tal i en af restklasserne i Z_m, og lave det om til et tal i en anden restklasse, så er det smart at vælge et tal der er den naturlige repræsentant for en restklasse. Altså, at det tal der skal krypteres er et tal mellem 0 og *m*. Når det krypterede tal dekrypteres, får man netop den naturlige repræsentant ud, og hvis dette ikke var ens oprindelige tal, så er det umuligt at sige, hvilket af de uendeligt mange andre tal i restklassen, der var klarteksten. Netop derfor er det vigtigt, at antallet af cifre i *m* er mindst 1 større end blokparstørrelsen (naturligvis er det blot nødvendigt, at *m* er mindst 1 tal større end det største tal i klarteksten, men for at være på den sikre side siger man, at antallet af cifre skal være mindst 1 større).

Som antydet i indledningen vil dette betyde, at vores blokparstørrelse skal være mindre. Derfor vælger vi i dette eksempel, at kryptere et bogstav af gangen. Det kan synes at pointen går lidt af systemet, da vi så i virkeligheden bare har lavet et meget omstændigt substitutionssystem. Det er også rigtigt, men så længe, at man ikke bruger det i praksis, er det godt nok til at illustrere teknikken. Hvis man betragter det som et substitutionssystem, vil det også kræve, at man finder et system i tallene, der bliver transmitteret, og det er nok sværere end at gætte *m*'s primtalsfaktorisering.

Hvis vi derfor krypterer et tal af gangen, og vores klartekst er "ABEHÅND", er det oversat til tal med den simple tegntabel *0 1 4 7 28 13 3*. Hvis denne meddelelse skal krypteres, skal hvert bogstav opløftes i *k*'te. $0^k = 0$, og på samme måde er $1^k = 1$, så *A* og *B* bliver altså ikke krypteret. Det ville endda også gå galt, hvis der blev brugt en blokparstørrelse som i noterne. *AB* bliver da til *01*, og det jo igen *1*, krypteret. Det er ikke så smart. Derfor er det nødvendigt med en ny tegntabel.

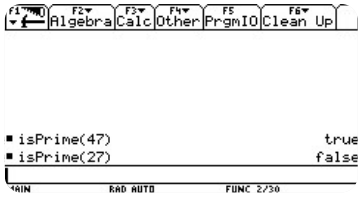
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Q	R	S	T	U	V	W	X	Y	Z	Æ	Ø	Å	_	.
26	27	28	29	30	31	32	33	34	35	36	37	38	39	40

I denne nye tegntabel oversættes ordet "ABEHÅND" til *10 11 14 17 38 23 13*.

Eksempel på RSA-kryptering

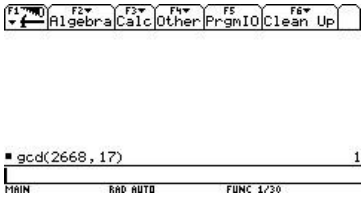
Først skal vi vælge os to primtal *p* og *q*. Her vælger vi $p = 59$ og $q = 47$. Er du i tvivl om, hvorvidt de tal du mistænker for at være primtal rent faktisk er det, findes der en funktion på TI-89'eren, der hedder isPrime(), der returnerer *true* hvis tallet er et primtal og *false*, hvis tallet ikke er et primtal.



Med de to primtal skal vi danne produktet $m = pq = 59 \cdot 47 = 2773$, som er første del af vores offentlige nøgle.

Vores hemmelige nøgle består af $\varphi(m) = \varphi(2773) = (59-1)(47-1) = 58 \cdot 46 = 2668$.

For at gøre vores offentlige nøgle færdig, skal vi vælge et tal *k*, der er indbyrdes primisk med $\varphi(m)$. Da vi ikke vil have for store tal at arbejde med, er det smarteste blot at vælge et lille primtal, der ikke går op i $\varphi(m)$. Da det eneste tal, der går op i et primtal pr. definition kun er 1 og tallet selv, er det største tal, der går op i både vores kandidat til *k* og $\varphi(m)$, altså 1. På TI-89 kan du tjekke om to tal er indbyrdes primiske. Dette gøres med funktionen *gcd()*, der står for *Greatest Common Divisor*. Funktionen giver altså den største fælles divisor i de to tal, der puttes ind i funktionen. Hvis dette tal er 1, er de to tal indbyrdes primiske. I vores tilfælde vælger vi $k = 17$.

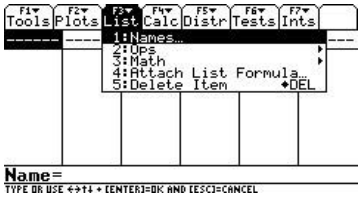


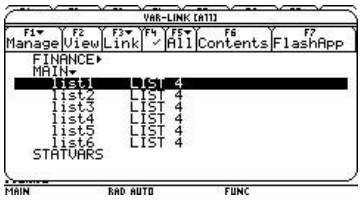
Nu har vi altså fået lavet vores offentlige nøgle og vores hemmelige nøgle:
Offentlige nøgle: $m = 2773$ og $k = 17$
Hemmelig nøgle: $\varphi(m) = 2668$

Nu er vi klar til at kryptere.

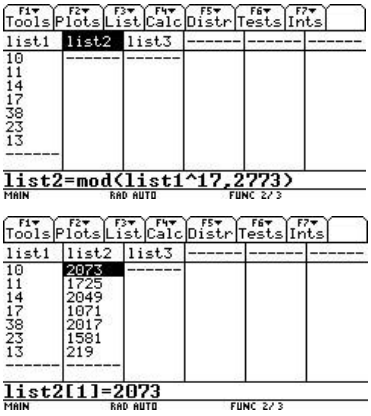
Lad os kryptere meddelelsen "ABEHÅND". Som det er vist ovenfor, bliver dette ord oversat til *10 11 14 17 38 23 13*.

For at gøre det så let som muligt for os selv, vil vi arbejde med lister. Åben derfor *Stats/List Editor*, og vælg *ok*. Der skal bruges tre lister. Hvis du ikke har defineret navne på dine lister skal du gøre det nu. Ellers kan man ikke referere til dem i de regnerier vi skal lave. Hav derfor markøren på det øverste felt. I statusbaren står der "*Name*=". Tryk på F3, og vælg "*Names…*". Under *MAIN*, finder du *list1* som navn. Vælg dette og tryk på ENTER og ENTER.
Navngiv nu på samme måde de følgende to lister *list2* og *list3*.





Nu skal vi have udfyldt *list1*, med vores klartekst. Sæt markøren på feltet med "------", og indtast den første værdi. Tryk på ENTER og indsæt på samme måde resten af klarteksten. Et tal i hver række.
Sæt markøren på feltet hvor der står *list2* og tryk ENTER. Vi skal nu lave regneoperationen der står på side 28 øverst i noterne. Når vi skal finde repræsentanten mellem *0* og *2773* af restklassen for fx 10^{17} ved division med *2773*, skal vi bruge funktionen *mod()*, der står for modulo. Denne findes i kataloget under *m*. Indtast nu følgende: *mod(list1^17,2773)*. Det betyder, at i første felt i *list2*, indsættes det tal, mellem 0 og 2773, som er kongruent med den 17. potens af første tal i *list1* modulo 2773.
Du skulle gerne få tallene: *2073 1725 2049 1071 2017 1581 219*. Der nu er vores kryptotekst.



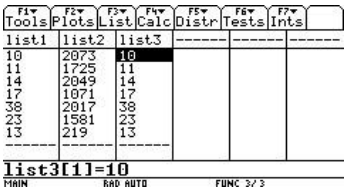
For at kunne dekryptere kryptoteksten, skal vi løse ligningen $1 = k \cdot u - \varphi(m) \cdot v$. I vores tilfælde bliver den til $1 = 17u - 2668v$. Dette gøres med Euklids algoritme. Der findes programmer til TI-89 som kan gøre dette automatisk[1].

I noternes Eksempel 4.5 er der givet en udførlig beskrivelse af hvordan ligningen $1 = 61u + 27v$ løses. Derfor vil vi ikke her gå i detaljer med udregningerne blot opskrive skemaet.

i	q_i	r_i	v_i	u_i
0	-	-2668	1	0
1	-156	17	0	1
2	-1	-16	1	0
3	16	1	1	157

Altså har vi at $1 = 157 \cdot 17 - 2668 \cdot 1$, og $u = 157$.

For at dekryptere skal vi nu tilbage i vores *Stats/List Editor*, og placere markøren i feltet, hvor der står *list3* og tryk på ENTER. Indtast nu følgende: *mod(list2^157,2773)*.
Resultatet skulle gerne være den oprindelige klartekst.



Hvis dine primtal *p* og *q* bliver store, vil du finde ud af, at når du skal dekryptere – det er oftest her, at tallene skal opløftes til høje potenser – at TI-89 står af.

Afslutning

TI-89 kan godt bruges til at lege lidt med kryptering. Men hvis man vil arbejde med bare lidt store primtal, der kan forhøje blokparstørrelsen, så det bliver rigtig sjovt at kryptere, skal man bruge meget mere regnekraft. Her kan matematikprogrammer på PC'en som Mathematica, Maple, Mathcad eller Derive være meget nyttige.

[1] http://www.cartesionline.it/materiali/algebra_numb_theory.cfm giver en meget detaljeret opskrift på hvordan man kan lave Euklids algoritme og andet relevant talteori på TI-89