Kryptering

ELLER

xhafgra ng tøer hyæfryvtg

P0

Anders Rune Jensen *Jasper Kjersgaard Juhl*
Ole Laursen *Martin Qvist*

21. september 2001

AALBORG UNIVERSITET

Det Teknisk-Naturvidenskabelige Fakultet
Aalborg Universitet

Basisuddannelsen

Titel:

Kryptering – kunsten at gøre ulæseligt

Projektperiode:
P0,
4.–21. sept, 2001

Projektgruppe:
B330

Gruppemedlemmer:
Anders Rune Jensen
Jasper Kjersgaard Juhl
Ole Laursen
Martin Qvist

Vejleder:
Hans Hüttel

Antal kopier: 9

Rapport – sideantal: 20

Appendiks – sideantal: 3

Synopsis:

Kryptering handler om at holde data hemmelige for uvedkommende. Der findes flere forskellige metoder at kryptere med. I denne rapport gennemgås ROT13 og XOR samt DES og RSA som er to af de mest brugte metoder i dag.

DES og RSA repræsenterer hver for sig to forskellige paradigmer inden for kryptologien; DES som et konventionelt system med symmetriske, private nøgler og RSA som et moderne system hvor en nøgle offentliggøres og dermed tillader signering og vilkårlige sikre forbindelser over ellers usikre kanaler som internettet.

Udviklingen inden for kryptologi har gjort at ubrydelige kryptosystemer nu kan være hvermandseje. Dette udmønter sig i sammenstød mellem modstridende interesser og har medført forsøg på lovgivning på området. Lovgivninger som rejser etiske spørgsmål og som i praksis ikke har virket.

Forord

Denne rapport er resultatet af vores første projekt, P0, på Aalborg Universitet. Så læseren må have den undskyldt.

Vi har i projektet lagt stor vægt på at samle os en så stor faglig viden inden for projektperioden som overhovedet muligt, og gennem rapporten forsøge at dokumentere den viden vi har fået og forsøge at besvare de spørgsmål som vi er stødt på.

Rapporten må frit distribueres i en hvilken som helst form.

En særlig tak til vores vejleder Hans Hüttel, som har været en stor hjælp gennem projektperioden.

God fornøjelse med rapporten

Jasper, Martin, Ole og Anders

Indhold

Kapitel 1

Indledning

Kryptering er de seneste år blevet mere og mere hverdag for computerbrugere verden over. I takt med elektroniske overførelser af f.eks. bankoplysninger er behovet for sikker kryptering blevet større og større – forestil dig at en cracker kan overvåge og endda ændre dine bankoverførelser. Dette behov stiller stadig strengere krav til de systemudviklere som arbejder med overførelser af fortroligt data.

Men faktisk er kryptering langt mere end blot at køre en eller flere funktioner i sit foretrukkede programmeringssprog. Bag krypteringen ligger en kompleks og spændende teori. Teorien går helt tilbage til oldtiden hvor man så de første simple krypteringsformer. Jo større regnekraft man senere har fået til rådighed, jo mere komplekse algoritmer ligger der bag de metoder der bruges.

1.1 Problemformulering

Rapporten vil blive baseret på følgende hovedspørgsmål:
Hvordan fungerer kryptering og hvilken betydning har det haft og vil det få for samfundet?
Herunder vil vi koncentrere os om følgende spørgsmål:

1. Hvad er kryptering, hvad er den historiske baggrund og hvilken rolle har det spillet gennem tiden?

   Vi ved at kryptering spillede en væsentlig rolle under 2. verdenskrig og vi vil forsøge at belyse denne rolle. Desuden vil vi kigge på hvad det betyder at en virksomhed som Tvind (næsten) kunne skjule bevismateriale for politiet.

2. Hvilke klassiske krypteringsmetoder findes der og hvordan virker disse?

   Af klassiske krypteringsmetoder mener vi ikke dem man brugte for lang tid siden. Vi lader vores tidslinje gå frem til i dag, men vi kalder de krypteringsmetoder som udelukkende anvender private nøgler for klassiske fordi ideen i dem har været kendt i årtusinder.

3. Hvilke moderne krypteringsmetoder findes der og hvordan virker disse?

   Her har vi især tænkt os at komme ind på RSA som er af de mest benyttede moderne krypteringsmetoder – det bliver for omfattende at kigge på andre systemer (som f.eks. El Gamal).

4. Hvilke tekniske og samfundsmæssige overvejelser er der i forbindelse med kryptering i fremtiden?

   Her vil vi forsøge at beskrive de interessekonflikter der kan være i forbindelse med krypteringslovgivning. Vi tænker både politisk og på den voksende lobbyisme.

Kapitel 2

Introduktion

2.1 Definitioner

I det følgende vil vi beskrive nogle generelle definitioner, som vil blive brugt meget igennem rapporten.

2.1.1 Kryptologi

Ordet *kryptologi* stammer fra latin, og er sammensat af 2 latinske ord; krypto (at skjule) og logi (læren om). Det dækker over tre forskellige begreber [Landrock og Nissen, 1997; Bauer, 1997], *steganografi* som drejer sig om helt at skjule at der overhovedet bliver sendt en meddelelse, *kryptografi* som drejer sig om at forvandle en meddelelse til noget der er uforståeligt for alle andre end de tilsigtede modtagere, og endelig *kryptoanalyse* som handler om at bryde krypteringen uden at være en tilsigtet modtager.

Selve ideen i at kryptere er at en part ønsker at holde en eller anden form for information hemmelig under transporten af informationen.

Hvis en meddelelse blev opsnappet i gamle dage, kunne man ofte finde ud af om informationen var faldet i de forkerte hænder – budbringeren dukkede simpelthen ikke op i hel tilstand.

I dag er det lidt anderledes. Transportvejene er også usikre i og med at de er åbne (det er f.eks. svært at kontrollere hvor et radiosignal havner henne), og ydermere er det meget sværere at finde ud af om meddelelsen er blevet opsnappet. Desuden er der i dag adgang til store mængder regnekraft i form af f.eks. pc'ere til at forsøge at bryde krypteringen.

Derfor er der i dag brug for meget effektive krypteringsformer.

2.1.2 Kryptografi

Et *kryptosystem* [Landrock og Nissen, 1997, s. 12] består af en metode til at foretage *kryptering* der gør en given klartekst m ulæselig, samt en metode til *dekryptering* der oversætter kryptoteksten c til den oprindelig besked m igen. Krypteringen $E(m)$ og dekrypteringen $D(c)$ kan evt. være sammenfaldende eller hinandens inverse funktioner.

Oftest skal der ved kryptering og dekryptering bruges noget ekstra information, en *nøgle* K. Så $E_k(m)$ betegner m krypteret med nøglen K og $D_k(c)$ betegner c dekrypteret samme nøgle.

2.2 Private nøgler

Til et kryptosystem med private nøgler [Hankerson et al., 2000, s. 230], den simpleste og klassiske form for kryptering, skal der bruges én nøgle. Nøglen man bruger til at kryptere med, er den samme som nøglen til at dekryptere med. Systemet kaldes af denne grund også for et symmetrisk kryptosystem.

Ideen i symmetrisk kryptering er meget ligetil fordi dekryptering foregår ved at gøre det modsatte af krypteringen, og eksempler går helt fra de simple krypteringsmetoder som bogstaverstatning til DES som er en moderne krypteringsstandard der stadig bruges.

De private nøgler skal naturligvis holdes hemmelige, sikkerheden i systemet hviler på dette.

2.2.1 Problemerne med symmetriske kryptosystemer

Der er imidlertid nogle problemer [Bauer, 1997, s. 171]:

1. Nøglerne skal overbringes ad en kommunikationskanal der er mere sikker end den der bruges til almindelig kommunikation – hvilket kan være umuligt, f.eks. hvis ens kommunikationsmiddel er et traditionelt computernetværk.

2. Hvis mange personer ønsker adgang til kommunikation med hinanden, kan antallet af nøgler blive meget stort. En nøgle skal jo holdes privat mellem to parter, så med et netværk med n parter der alle ønsker at kunne kommunikere med hinanden, bliver antallet af nøgler $\binom{n}{2}$. For blot 10.000 brugere (som f.eks. på et universitet) drejer det om i alt ca. 50 mill. nøgler.

3. Og endelig kan man ikke bruge systemet til at signere meddelelser med (altså som en erstatning for en underskrift). Så længe der er to parter som besidder nøglen, kan begge have krypteret den.

2.3 Offentlige nøgler

Disse svagheder ved symmetrisk kryptering har i moderne tid skabt behovet for andre løsninger.

2.3.1 Asymmetriske kryptosystemer

Med et *asymmetrisk* kryptosystem er der én nøgle til at kryptere med, den *offentlige nøgle*, og én nøgle til at dekryptere med – sidstnævnte skal stadig holdes hemmelig. Den nøgle der benyttes til at kryptere, kan derimod offentliggøres og bruges af alle til at sende en meddelelse der ikke kan læses af andre modtageren som jo sidder med den eneste nøgle der kan „låse“ beskeden op igen.

Hvis K_p betegner den private nøgle og K_o den offentlige, virker systemet altså ved

$$D_{K_p}(E_{K_o}(m)) = m$$

Eftersom kryptosystemet er offentligt kendt, skal det være en så beregnsmæssig tung opgave at bryde systemet vha. en udtømmende søgning, at det ikke kan lade sig gøre inden for en overskuelig fremtid. Man kan altså opstille nogle grundlæggende krav til et kryptosystem baseret på offentlige nøgler [Landrock og Nissen, 1997, s. 58]:

1. Konstruktionen af den offentlige og den private nøgle skal være simpel.

2. Det skal være simpelt at foretage krypteringen vha. den offentlige nøgle og klarteksten.

3. Retablering af klarteksten vha. den hemmelige nøgle af kryptoteksten skal være simpel.

4. At bestemme den hemmelige nøgle ud fra den offentlige nøgle skal være et beregningsmæssigt svært problem.

5. Det skal være svært at retablere klarteksten ud fra et kendskab til kun den offentlige nøgle og kryptoteksten.

Et kryptosystem der besidder disse egenskaber, løser de to første problemer med symmetriske kryptosystemer: når de to nøgler er genereret, kan den ene distribueres over åbne kanaler uden risiko for kompromittering af systemet. Og der er også kun brug for ét nøglepar pr. person, altså $2n$ nøgler i alt i forhold til $\binom{n}{2} = n \cdot (n-1)/2$ ved et symmetrisk system. De offentlige nøgler kan opbevares i et fælles katalog[1] som man så kan slå en givens person nøgle op i når man har behov for at sende en hemmelig meddelelse til vedkommende, lidt på samme måde som en telefonbog.

[1] Med et fælles opslagskatalog er der dog naturligvis brug for en eller anden form for system eller myndighed til at sikre at de enkelte personers offentlige nøgler faktisk tilhører de givne personer. Hvis det lykkes en ondsindet person at lægge en falsk offentlig nøgle ind i en andens navn, vil det jo være muligt for vedkommende at læse de krypterede meddelelser indtil falskneriet er opdaget.

2.3.2 Signeret kryptering

Hvis kryptosystemet ydermere har egenskaben

$$E_{K_o}(D_{K_p}(m)) = m \tag{2.1}$$

kan det også benyttes til signering [Bauer, 1997, s. 173] hvilket løser den sidste hage ved et symmetrisk system. Kryptering med signering virker som følger:

Hvis A ønsker at sende en meddelelse til B, krypterer A først klarteksten med sin private nøgle:

$$c_A = D_{K_{p,A}}(m)$$

Det kan virke lidt mærkeligt at man bruger dekrypteringsfunktionen på klartekst, men det følger af kriteriet for signeringen givet ved (2.1). Derefter tilføjer A sit navn, „A“, og krypterer „A“ og c_A med B's offentlige nøgle som A har slået op til lejligheden:

$$c_{BA} = E_{K_{o,B}}(\text{„}A\text{“}, c_A)$$

Nu kan c_{BA} sendes over en åben kanal til B.

Herefter dekrypterer B kryptoteksten med sin private nøgle:

$$D_{K_{p,B}}(c_{BA}) = (\text{„}A\text{“}, c_A)$$

B bemærker ud fra „A“ at afsenderen er A, og kan bruge A's offentlige nøgle til at genoprette klarteksten:

$$E_{K_{o,A}}(c_A) = m$$

Hvis det sidste skridt giver en meningsfuld klartekst m, kan B være sikker på at A faktisk *er* afsenderen fordi kun A's private nøgle kan have været brugt til at kryptere klarteksten med.

Kapitel 3

Historisk set

De første kendte eksempler på kryptering stammer helt tilbage fra 500 før vor tidsregning hvor det blev brugt af den spartanske regering til at sende hemmelige beskeder til deres generaler [Beutelspacher, 1994, s. 3].

Et andet eksempel fra oldtiden er Cæsars rotering [Hankerson et al., 2000, s. 231], hvor en variation af denne stadig bruges i dag.

3.1 Cæsars rotering

Cæsars rotering var det første kendte eksempel på bogstaverstatningskryptering. Det virker ved at man giver alle bogstaverne i alfabetet en værdi, og derefter foregår krypteringen ved at lægge K til det oprindelige bogstav m fra kildeteksten. Hvis tallet så bliver over 26 (i det engelske alfabet er der kun 26 bogstaver), trækkes 26 fra så man får et c mellem 1 og 26[1].

Eksempel med $K = 3$:

Klartekst:	t	e	s	t	t	e	s	t	t	e	s	t
Som tal:	20	5	19	20	20	5	19	20	20	5	19	20
Krypteret:	23	8	22	23	23	8	22	23	23	8	22	23
Kryptotekst:	w	h	v	w	w	h	v	w	w	h	v	w

Cæcar benyttede efter sigende selv $K = 3$, men $K = 13$ er mere populær i vore dage, måske fordi krypteringen og dekrypteringen kan foregå med nøjagtigt samme funktion. Denne specielle kryptering kaldes også ROT13:

$$ROT_{13}(m) = c \quad \text{og} \quad ROT_{13}(c) = m$$

Da hvert bogstav i kildeteksten kun kan blive oversat til ét bogstav er denne krypteringsform meget sårbar over for frekvensanalyse, hvor man tæller frekvensen af de enkelte bogstaver og sammenholder dem med det man tror teksten er. F.eks. er "e" det mest brugte bogstav i de fleste europæiske sprog, derfor vil det mest brugte bogstav i den krypterede tekst sandsynligvis svare til "e". Med 26 kombinationsmuligheder ville en udtømmende søgning dog heller ikke tage al for lang tid.

Dette gør dog ikke ROT_{13} ubrugelig til andre formål. F.eks. bruges det i nyhedsgrupper til at skjule budskaber hvis nogen finder materialet fornærmende [Hankerson et al., 2000, s. 231], eller til at skjule svaret på vittigheder.

3.2 Anden verdenskrig

Under 2. verdenskrig brugte man i stor stil radioer til at klare langdistancekommunikationen. Fordelen var naturligvis at man ikke behøvede at have fysisk kontakt, men samtidig giver radiobølgerne uønskede personer mulighed for ubemærket at lytte med. Så nu måtte man ved alle meddelelser regne med at fjenden lyttede med.

[1] Egentlig foregår her en modulo-operation, se bilag B

For at undgå at fjenden forstod budskaberne, det kunne f.eks. være ret kritisk hvis den fremtidige position af en ubåd blev røbet, begyndte man at udvikle kryptering i stor stil.

Enigma er et kendt eksempel på en af krigstidens frembringelser [Bauer, 1997, s. 106–109]. Det var en såkaldt rotorkrypteringsmaskine som virkede vha. et kompliceret system af bogstaverstatninger. Tyskerne benyttede anordningen i stor stil – man regner med at der var måske omkring 10.000 maskiner i omløb – men uheldigvis for dem blev systemet allerede brudt i starten af krigen, hjulpet på vej af nogle kaprede eksemplarer. Forskellige senere modeller formåede ikke at rette op på problemet i længere tid ad gangen.

De andre stormagter benyttede sig af lignende anordninger, og visse af dem blev brudt på tilsvarende måde af aksemagterne.

Af modstandsbevægelserne blev der også benyttet forskellige metoder til at skjule budskabet for uønskede lyttere. En kendt og noget utraditionel måde bestod i helt at skjule budskabet ved at indrykke uskyldigt udseende annoncer, f.eks. kunne "Skyd kl. 8" omsættes til: "Søren Kristensen yder dig og klassen lækker og opkogt tilberedt tyksteg efterbehag" (forbogstaverne indeholder budskabet mens "og" svarer til mellemrum).

Under 2. verdenskrig kunne krypteringens effektivitet altså betyde forskellen mellem liv og død.

3.3 Tvind

Et helt aktuelt eksempel er da politiet i foråret 2001 fra det berygtede Tvind-imperium beslaglagde en række materialer der viste sig at være krypterede.

Flere sikkerhedseksperter, heriblandt en dansk professor med speciale i kryptologi, Peter Landrock, udtalte til Computer World [Thorhauge, 2001b] at politiet ikke ville kunne bryde krypteringen som skulle benytte sig af nøgler på 128 og 256 bit. Det lykkedes dog alligevel politiet under mystiske omstændigheder at få dekrypteret dataene.

Politiet sagde til Computer World [Thorhauge, 2001a] at de ved hjælp af "normale" hackermetoder har brudt den kryptering som beskyttede fem af de beslaglagte computere. Det afviser flere kilder dog: det er helt usandsynligt at man skulle kunne bryde systemet. Berlingske Tidende har ifølge Computer World [Thorhauge, 2001a] fået oplyst at det er en af Tvinds samlevere som har udleveret et password til politiet.

3.4 Virus

Traditionen tro skal en ny teknologi naturligvis udnyttes ondsindet [Bjørk, 2001]. Dette var bestemt tilfældet da ukendte virus-programmører udsendte en email, se bilag A på side 22, som påstod at være fra Microsoft.

En fil der var vedhæftet brevet, indeholdt en virus som unægtelig havde en lidt kreativ ødelæggende effekt. Ifølge IT-Avisen.dk krypterede den alle kørbare filer på harddisken med en vilkårlig nøgle. Dette gør dem naturligt nok ubrugelige for brugeren. Uden at finde nøglen for krypteringen er man reelt ude af stand til at redde ramte maskiner.

Kapitel 4

Klassiske kryptosystemer

4.1 XOR

XOR er en boolsk operator ligesom AND eller OR (som man normalt skriver symbolsk med $\wedge$ og $\vee$, symbolet for XOR er $\oplus$) – den kan defineres bitvist som $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $0 \oplus 1 = 1$ og $0 \oplus 0 = 0$. Ud fra dette kan man vise at $(k_p \oplus m) \oplus k_p = m$ så en måde at kryptere på, er at XOR'e en given nøgle med klarteksten hvorefter den kan dekrypteres ved at XOR'e den igen med nøglen. På lignende måde har man at $(k_p \oplus m) \oplus m = k_p$. Dvs. kender man kryptoteksten $c = k_p \oplus m$ og klarteksten, kan man nemt finde nøglen.

Lad os prøve XOR-kryptering i praksis. Lad "tim" være m og "mit" være vores nøgle K_p. En talværdi for bogstavet "m" kunne være 109 der svarer til det binære tal 01101101, for "i" 105 svarende til 01101001 og for "t" 116 svarende til 01110100.

	$t \oplus m$		$i \oplus i$		$m \oplus t$
t:	01110100	i:	01101001	m:	01101101
m:	01101101	i:	01101001	t:	01110100
c_1:	00011001	c_2:	00000000	c_3:	00011001

Hvis vi går ud fra at vi kun kender c og m, kan vi finde K_p:

	$c_1 \oplus t$		$c_2 \oplus i$		$c_3 \oplus m$
t:	01110100	c_2:	00000000	c_3:	00011001
c_1:	00011001	i:	01101001	m:	01101101
K_1:	01101101	K_2:	01101001	k_2:	01110100
	m		i		t

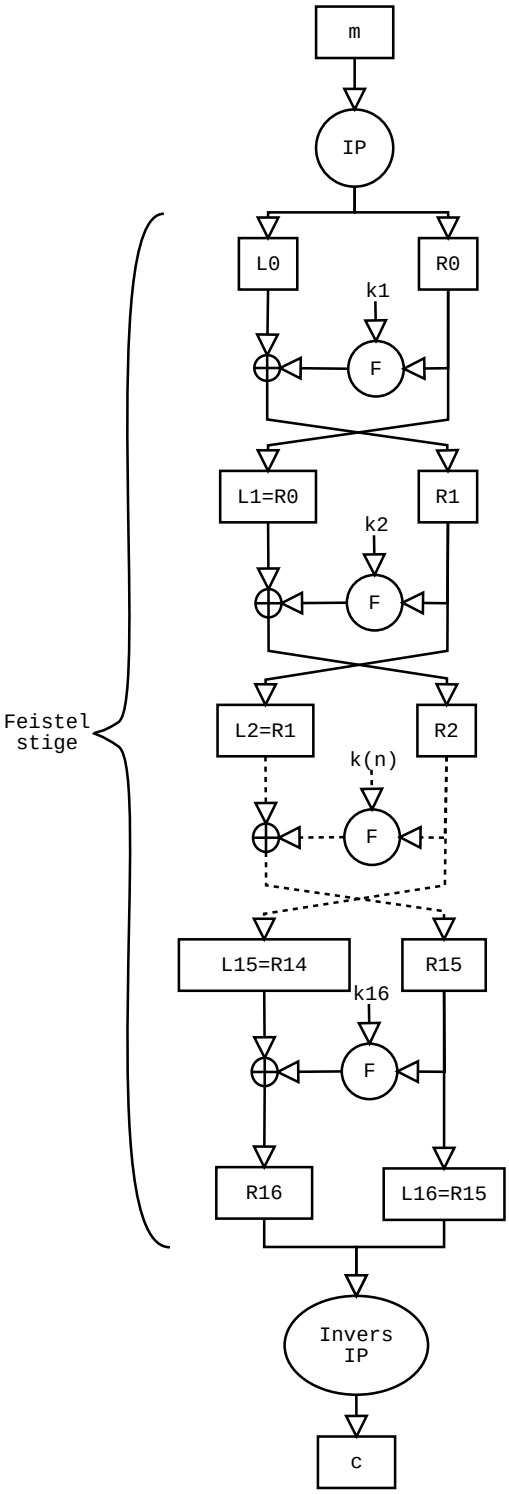
Som ses i de ovenstående to eksempler er der med XOR tale om en ret simpel regneform, mens resultatet bliver relativt godt skjult. Faktisk så godt at hvis nøglen kun bruges en gang, og hvis nøglen er tilfældig, så er kryptoteksten ubrydelig, uanset hvor meget beregningskraft man har til rådighed [Hankerson et al., 2000, s. 236].

Dette hænger sammen med at alt efter hvilken nøgle man har, kan man faktisk lave et hvert stykke tekst ud fra en given kryptotekst, blot ved at ændre nøglen tilstrækkeligt. Derfor har man ingen grund til at favorere den ene tekst frem for den anden. Brugen af engangsnøgler kaldes one-time pads – det er naturligvis ikke praktisk til almindeligt brug.

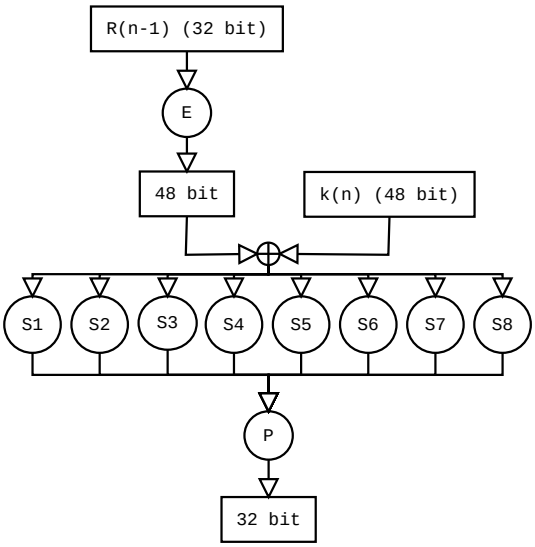
4.2 DES

DES, Data Encryption Standard, blev optaget som standard i 1977, og er siden blevet til den mest kendte symmetriske krypteringsmetode [Hankerson et al., 2000, s. 243]. Den bygger på et antal permutationer og en metode kaldet en Feistel-stige og XOR.

Ligesom med andre krypteringsmetoder starter man med en besked m, og denne deles op i blokke af 64 bit som hver gennemgår en kryptering. Der er så forskellige måder at samle de krypterede blokke på, hver med deres fordele.



Figur 4.1: Feistel-stigen – teksten sendes ind foroven og kombineres efter den indledende permutation med undernøglerne efter tur.



Figur 4.2: F-funktionen

De 64 bit blokke går igennem en indledende permutation IP, herefter deles de op i to dele L_0 og R_0, hvor L_0 er de første 32 bit og R_0 de næste 32 bit. R_0 krypteres via en funktion F med undernøgle k_1 (udvælgelse af disse nøgler beskrives senere), og bliver så XOR'et med L_0. Resultatet fra dette bliver så til $R_1 = L_0 \oplus F_{k_1}(R_0)$ og den gamle R_0 bliver til $L_1 = R_0$. Dette gentages 16 gange med 16 forskellige nøgler (dog med den undtagelse at resultatet ved den sidste gentagelse byttes om) som vist på figur 4.1 på foregående side. Dette har den fordel at dekryptering kan foregå på samme måde som kryptering, blot med undernøglerne i modsat rækkefølge. Til sidst permuteres dataene igen, med det der svarer til den inverse af IP.

4.2.1 Indledende permutation

Permutation betyder at ombytte, og det er faktisk også det der sker [DES, 1993]. Dataene fra m bliver flyttet rundt bitvist til andre placeringer. Dette sker efter et fast skema, hvoraf første linie er vist nedenfor.

58 50 42 34 26 18 10 2

Det betyder at den første bit data hentes fra position 58 i m, den anden bit data hentes fra position 50 osv. I standarden står ikke grunden til denne permutation, men hvis man tænker på at dataene skal deles op i to dele giver det mere mening – permutationen sørger for at alle lige bits bliver til L_0 og alle ulige til R_0.

Det næste skridt på stigen er at R_0 går igennem funktionen F.

4.2.2 F-funktionen

Det første skridt i funktionen er E som laver de 32 bit data fra R om til 48 bit data – disse deles op i grupper af 6 bit [DES, 1993]. På samme måde som i IP sker det efter et fast skema.

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Dvs. første bit der kommer ud af E, bliver bit nr. 32 i R, næste bit nr. 1 osv. Bemærk at visse bit optræder flere gange – på den måde får de 48 bit ud af 32.

Nu har vi 48 bit data som vi kan XOR'e med undernøglen k_n. Herefter bliver dataene inddelt i 8 blokke på 6 bit og ledt ind i hver sin S-boks hvor en given S-boks laver de 6 bit om til 4 bit data. I S-boks 1's tilfælde sker dette efter følgende skema [DES, 1993]:

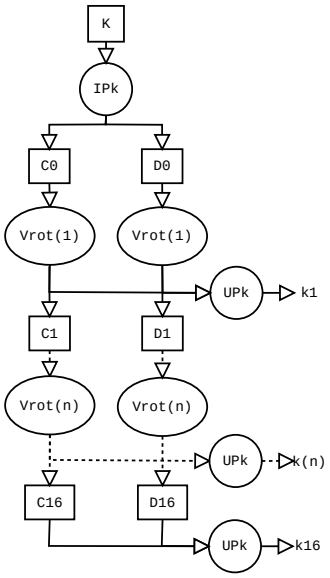
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Den første og den sjette bit i en blok beskriver et tal mellem 0 og 3, dette angiver linjenummeret i skemaet. De inderste 4 bit angiver så kolonnenummeret, og resultatet fra dette bliver et tal fra skemaet som er mellem 0 og 15, dvs. 4 bit.

Herefter gennemgår dataene endnu en fast permutation P.

4.2.3 Undernøglerne

Det sidste vi mangler at forklare er hvordan undernøglerne k_1 til k_{16} opstår.



Figur 4.3: Generering af undernøglerne

Den benyttede nøgle K, bliver først permuteret ved $IP(k)$. Dette sker som de andre permutationer efter et fast skema, og har igen til formål at blande bittene ud før dataene bliver delt i to.

Herefter gennemgår henholdsvis C_0 og D_0 en V_{rot} funktion. Denne funktion roterer alle bit *en* (ved undernøglerne 1, 2, 9 og 16) eller *to* (ved de andre undernøgler) pladser til venstre.

Funktionens resultater bliver så sendt, se figur 4.3, til $UP(k)$ som også er en fast permutation. Dette giver så undernøglerne k_1 til k_{16}.

4.2.4 Lidt om TripleDES

DES bruges stadig i dag, og der er heller ikke fundet nogle svagheder i krypteringsalgoritmen[1]. Men nøglelængden på 56 bit er ved at være for lille i forhold til den beregningsstyrke som moderne computere har. Derfor er man i dag gået over til at bruge DES-krypteringen flere gange i træk, f.eks. tre gange som med 3DES [Hankerson et al., 2000, s. 246]. Dette forøger antallet af mulige nøgler fra 2^{56} til $(2^{56})^3 = 2^{168}$ og skulle holde uønskede personer hen de næste par år.

[1]Bortset fra nogle meget få, bestemte nøgler hvor krypteringen bliver triviel (se evt. [Menezes et al., 1996]), men disse bruges naturligvis ikke i praksis

Kapitel 5

Moderne kryptosystemer – RSA

RSA-systemet blev udviklet i 1977 af Rivest, Shamier og Adleman [Landrock og Nissen, 1997, s. 101], deraf navnet. For at kunne forstå RSA er det nødvendigt at kende lidt til nogle talteoretiske begreber som derfor vil blive beskrevet først. Resten af de mest grundlæggende begreber er gennemgået i bilag B.

5.1 Eulers ϕ-funktion

For et positivt helt tal n betegnes antallet af tal mellem 0 og n som er indbyrdes primiske med n, med $\phi(n)$ [Landrock og Nissen, 1997, s. 89]. Udtrykt mere formelt har man (idet (a, n) her betegner den største fælles divisor i a og n):

DEFINITION 5.1.1 *For $n \in \mathbb{N}$ er $\phi(n)$ antallet af elementer i mængden*

$$\{a \in \mathbb{Z} \mid 0 < a < n \wedge (a, n) = 1\}$$

Hvis man kender primfaktoriseringen af n, kan man via en sætning altid beregne $\phi(n)$. For RSA er det dog kun nødvendigt med et specialtilfælde af sætningen, nemlig for produktet af to primtal:

SÆTNING 5.1.1 *Lad p og q være primtal og $p \neq q$. Da er*

$$\phi(pq) = (p-1)(q-1)$$

Bevis: Det samlede antal af hele tal a der opfylder $0 < a < pq$, er $pq - 1$. Hvis et tal a skal tælles med i beregningen af $\phi(pq)$, skal det desuden opfylde at $(a, pq) = 1$, dvs. at hvis det *ikke* skal tælles med, skal $(a, pq) \neq 1$.

Nuvel, hvis $(a, pq) \neq 1$, må der for en fælles divisor $d \neq 1$ i a og pq gælde at $d = p \vee d = q$ da p og q er primtal. Med andre ord skal p eller q gå op i a. Det sker hvis a antager en af værdierne $p, 2p, 3p, \ldots, (q-1)p$ eller $q, 2q, 3q, \ldots, (p-1)q$ (bemærk at pq ikke selv skal med da $a < pq$ pr. definition).

Længden af disse to talrækker ses at være henh. $(q-1)$ og $(p-1)$; fratrukket det samlede antal af tal mellem 0 og pq må dette være $\phi(pq)$:

$$\phi(pq) = pq - 1 - ((q-1) + (p-1)) = pq - p - q + 1$$

Tilbage er blot at bemærke at $(p-1)(q-1) = pq - p - q + 1$.

5.2 Ideen i RSA

Genereringen af et RSA-nøglesæt sker på følgende måde:

1. To store primtal p og q vælges, og produktet $n = pq$ beregnes.
2. $\phi(n) = (p-1)(q-1)$ (se (5.1.1)) beregnes.
3. Et tal e vælges således at $0 < e < \phi(n)$ og $(e, \phi(n)) = 1$.

4. Til sidst vælges d således at d opfylder $ed \pmod{\phi(n)} = 1$.

RSA-nøglesættet består så af den offentlig nøgle i form af talparret (n,e) og den privat nøgle i form (n,d). Krypteringen foregår ved:

$$c = m^e \pmod{n}$$

Dekrypteringen foregår ved:

$$m = c^d \pmod{n}$$

Dette kan vi sammenfatte til grundstenen i RSA:

$$D_{k_p}(E_{k_o}(m)) = (m^e)^d \pmod{n} = m^{ed} \pmod{n} = m$$

for alle $0 < m < n$. At dette er rigtigt, skal naturligvis bevises.

Sikkerheden i RSA bygger på at man skal kende $\phi(n)$ for at kunne finde d ud fra n og e. $\phi(n)$ kan enten bestemmes direkte eller ved at primfaktorisere n til p og q, men begge dele er et beregningsmæssigt problem sålænge n er tilstrækkeligt stort (håber man da i hvert fald [Landrock og Nissen, 1997, s. 101]).

5.2.1 Bevis for $m^{ed} \pmod{n} = m$

SÆTNING 5.2.1 *Med m, n, e og d valgt som beskrevet i forrige afsnit, er*

$$m^{ed} \pmod{n} = m$$

Bevis: Vi deler beviset (som bygger på [Landrock og Nissen, 1997, s. 105]) op i to tilfælde, $(m,n) = 1$ og $(m,n) \neq 1$.

For $(m,n) = 1$ giver (B.5.1) at

$$m^{ed} \pmod{n} = m^{ed \pmod{\phi(n)}} \pmod{n}$$

og da $ed \pmod{\phi(n)} = 1$ og $0 < m < n$, fås

$$m^{ed \pmod{\phi(n)}} \pmod{n} = m^1 \pmod{n} = m$$

For $(m,n) \neq 1$ må, eftersom n netop er valgt som et produkt af to forskellige primtal, $n = pq$, enten p eller q være divisor i m.

Hvis p går op i m, er $m \pmod{p} = 0$ og dermed også $m^{ed} \pmod{p} = 0$ (følger af potensregnereglen for modulo-udtryk). Dvs. at $m^{ed} \pmod{p} = m \pmod{p}$. Ifølge den kinesiske restsætning (B.4.1) er beviset så fuldendt hvis det også lykkes at vise at $m^{ed} \pmod{q} = m \pmod{q}$.

Da $ed \pmod{\phi(n)} = 1$, findes et helt tal k sådan at $ed = 1 + k\phi(n)$. Man får

$$m^{ed} = m^{1+k\phi(n)} = m \cdot m^{k(q-1)(p-1)} = m\left(m^{(q-1)}\right)^{k(p-1)}$$

Før vi kan regne videre på hvad der sker når man tager modulo til dette udtryk, skal vi bruge et andet resultat. Da $m < n$ og $pq = n$, kan p og q ikke begge være divisorer i m. Vi har antaget at p er divisor m, og så følger at $(q,m) = 1$. Fermats lille sætning (B.5.2) giver da at $m^{q-1} \pmod{q} = 1 \pmod{q}$. Vha. dette og regnereglerne for modulo-udtryk får man så

$$\begin{aligned}
m^{ed} \pmod{q} &= m\left(m^{(q-1)}\right)^{k(p-1)} \pmod{q} \\
&= \left(m \pmod{q}\left(m^{(q-1)} \pmod{q}\right)^{k(p-1)} \pmod{q}\right) \pmod{q} \\
&= \left(m \pmod{q} \cdot 1^{k(p-1)} \pmod{q}\right) \pmod{q} \\
&= m \pmod{q}
\end{aligned}$$

5.3 Et eksempel

En nøglegenerering i RSA kan foregå på følgende måde:

1. $p = 37$ og $q = 89$ hvilket giver $n = pq = 3293$. p og q vælges her små for at lette overblikket og udregningerne
2. $\phi(n)$ bliver så: $\phi(n) = (p-1)(q-1) = 36 \cdot 88 = 3168$
3. Tallet e vælges til at være 25 – at $(e,\phi(n)) = 1$ er opfyldt kan f.eks. testes med en berømt algoritme af Euklid [Beutelspacher, 1994, s. 108] som giver den største fælles divisor.
4. Til sidst findes tallet $d = 2281$ som opfylder at $ed \pmod{\phi(n)} = 1$.

Vi har nu bestemt den private nøgle til $(n,d) = (3293,2281)$ og den offentlige nøgle til $(n,e) = (3293,25)$. Bemærk at primtallene p og q samt $\phi(n)$ ikke bruges til noget efter nøglesættet er bestemt. Man skal faktisk sikre sig at de er slettet, idet hele sikkerheden i RSA hviler på at det er svært at bestemme p og q ud fra n.

Med nøglesættet i hånden kan vi nu kryptere klarteksten EN KRYPTERET BESKED. Hvis vi tildeler bogstaverne værdier efter $A = 01$, $B = 02$ osv. helt op til $Å = 29$ (mellemrum får så 30). For klarteksten skal der gælde $0 < m < n$ så beskeden skal deles op i blokke – med den valgte værdi af n kan man kryptere 2 bogstaver af gangen:

```
E N    K  R Y  P T  E R  E T    B  E S  K E  D
0514 3011 1825 1620 0518 0520 3002 0519 1105 0430
```

Allerede ved en så lille n-værdi, støder man på det praktisk problem at beregne $0514^{25} = \pmod{3293}$. Bare den første del $0514^{25} \approx 10^{67}$ er så stor at det besværligt at regne med. Men eftersom 514^{25} kan skrives som

$$\begin{aligned} 514^{25} &= 514 \cdot (514)^{24} = 514 \cdot (514^2)^{12} \\ &= 514 \cdot ((514^2)^2)^6 = 514 \cdot (((514^2)^2)^2)^3 \\ &= 514 \cdot (((514^2)^2)^2)^2 \cdot ((514^2)^2)^2 \end{aligned}$$

kan man vha. en af moduloregnereglerne (sætning B.2.2) undgå de store tal.

Kapitel 6

Det fremtidige perspektiv

6.1 Hvilke algoritmer har en fremtid?

Man skulle jo tro at hele verden straks ville gå over til RSA da den har et smartere nøglesystem end de symmetriske algoritmer. Men RSA er omkring 100 gange langsommere end DES, og derfor bruges DES og specielt TripleDES stadig i praksis. F.eks. ved client/server-programmel kan dette have stor betydning for hastigheden af programmet.

En anden fordel ved symmetrisk kryptering er at den kan gøres stærkere i takt med computere bliver stærkere, uden det har stor betydning for krypteringens hastighed. Asymmetriske kan også gøres stærkere ved at vælge større primtal, men dette kræver flere beregninger og går dermed ud over hastigheden.

Men asymmetriske krypteringsalgoritmer er uundværlige ved kommunikation over Internettet, hvor de to parter måske sjældent er i fysisk kontakt med hinanden. Her har bl.a. RSA gjort det muligt at kommunikere krypteret uden forudgående at kende en hemmelig nøgle.

6.1.1 Hybridløsninger

En løsning der er blevet mere og mere populær, er hvad man kunne kalde en hybridløsning der bruger fordelene fra begge typer.

Hvis A skal snakke med B, starter A med at bede B om en nøgle. Derefter bruger B en asymmetrisk kryptering til at sende en krypteringsnøgle K_e til A. A har nu en sikker kanal, som han ved at kun B kan læse. Herefter sender af en tilfældig dannet nøgle til kommunikation som han krypterer med K_e. B dekrypterer så denne nøgle og de kan nu bruge denne nøgle til kommunikation med f.eks. TripleDES. På samme måde kan dette kombineres med signering så B er sikker på at han ikke får en nøgle fra en anden.

6.1.2 Nutidige anvendelser

I det følgende vil vi kort beskrive to systemer baseret på hybridløsningen som er uundværlige for mennesker i dag. Det første er SSH som er et program med en protokol til sikkert at udføre kommandoer over et netværk på en anden maskine, det andet, PGP, bruges til at signere og kryptere email.

Secure Shell

Den første version af Secure Shell blev udviklet i 1995 af en universitetsstuderende fra Helsinki Universitet, fordi han var træt af at få sniffet sine adgangskoder [Barrett og Silverman, 2001]. I dag anslås der at være omkring 2 mill. SSH-brugere [Acheson, 2001].

SSH bruger et kryptosystem med offentlige nøgler, som f.eks. RSA, til identifikation og sikring af at det er den rigtige maskine man snakker med. Herefter foregår al kryptering med et symmetrisk kryptosystem, f.eks. Triple DES eller et ved navn Blowfish. Herved fungerer SSH som en erstatning for tidligere løsninger som telnet eller rlogin [Myers, 2000]. Med til SSH hører også krypterede filoverførsler så man kan sende sine filer uden risiko for opsnapning.

En ting i forbindelse med SSH er at flere af krypteringsalgoritmerne er beskyttede af patenter, hvilket især giver problemer for de frie, åbne udgaver der er i omløb. Enten må man så undvære visse algoritmer eller også må man forsøge at arbejde sig rundt om det idet ikke alle lande tillader denne form for patenter.

Pretty Good Privacy

Pretty Good Privacy blev startet af Phil R. Zimmermann, og programmets omtumlede historie afspejler meget godt USA's holdning til kryptering – den viser NSA's og den amerikanske regeringens forsøg på at hindre udbredelsen af stærk kryptering (hele historien kan læses her [Back, 2001]). For at komme uden de amerikanske eksportrestriktioner blev kildekoden en overgang udgivet som en bog som man så kunne skanne direkte ind; dermed blev der jo ikke eksporteret noget i elektronisk form . . .

En forudsætning for signeringen og krypteringen af email er at man får PGP til at generere en privat og offentlig nøgle. Den offentlige nøgle kan derefter sendes til en nøgle-server så andre har let adgang til den – der er et indviklet system hvor folk siger god for hinandens nøgler til at sikre at de faktisk er ægte [Ashley, 1999].

Krypteringen af selve teksten sker på samme måde som beskrevet i foregående afsnit med private nøgler, simpelthen fordi det er hurtigere.

6.2 Kryptosystemernes sikkerhed

Et hyppigt anvendt mål for sikkerheden i kryptosystemer er antallet af kombinationsmuligheder der skal afprøves ved en udtømmende søgning. Men den udtømmende søgning er blot et udtryk for arbejdsmængden i det værst tænkelige tilfælde – f.eks. kan simple bogstaverstatningssystemer hurtigt brydes ved at udnytte anden kendt viden om klarteksten, nemlig at de forskellige bogstaver i et almindeligt sprog optræder med bestemte hyppigheder.

Man kan ikke afvise at lignende analyser eller andre smarte metoder der udnytter ekstra viden om situationen, kan bruges til at skære ned på antallet af muligheder ved de systemer som bruges i dag, som DES og RSA, og derved rokke ved deres sikkerhed. En sådan afvisning ville skulle bygge på matematisk stringente beviser, og dem har man ikke [Landrock og Nissen, 1997, s. 120]. Kryptering med de offentlige nøglesystemer er i denne sammenhæng specielt sårbare fordi den samme nøgle bruges over lange perioder.

Og man skal også huske på at et kryptosystem er tæt forbundet med den tid det er opfundet i, ingen af de hidtidige systemer er rigtigt fremtidssikrede selvom de måske kan udvides i takt med tidens gang, f.eks. i form af længere nøgler. Lidt ekstrapolering på den hidtidige eksponentionelle vækst i regnekraft for computere giver at de nøgler man benytter i dag til RSA om blot et par årtier sandsynligvis være inden for rækkevidden af hvad man kan nå at regne sig igennem selv med en udtømmende søgning [Landrock og Nissen, 1997, s. 120].

Men DES og RSA har trods alt efterhånden været brugt i mange år uden at der er blevet fundet nogen alvorlige fejl ved dem.

6.3 Interessekonflikter og krypteringslovgivning

Efterhånden som krypteringsmetoderne bliver bedre og sikre, bliver det nødvendigt at tage politisk stilling til kryptering. Forskellige parter har forskellige interesser:

- Den almindelige borger er interesseret i at beskytte sine private oplysninger og at kunne kommunikere via internettet uden andre kan følge med (beskyttelse af privatlivets fred er en menneskerettighed [Menneskerettighederne, 1948, art. 12]).
- Virksomheder vil gerne kunne beskytte fortrolige data for at forhindre industrispionage – spionagenetværket Echelon menes at have kostet europæiske virksomheder milliarder af kroner [Langvad, 2001].
- Terrorister og andre forbrydere ønsker af indlysende grunde at kunne kommunikere uden politiet kan lytte med.
- Staten har interesse i at kunne få efterretninger om trusler mod landet, både indre og ydre. Det er derfor i inderrigsefterretningstjenestens interesse at borgerne *ikke* kan skjule noget, og det er i udenrigsefterretningstjenestens interesse at andre lande ikke bliver i stand til holde deres kommunikation hemmelig. Under en krig er dette vitalt for militæret, som det sås under sidste verdenskrig.

Den enkelte stats interesser er altså i konflikt med borgernes og andre staters interesser. Dette udmønter sig i tiltag til at lovgive om f.eks. tvungne bagdøre eller svagheder i kryptosystemerne eller forbud mod stærk

kryptering (dvs. kryptering der rent faktisk kan holde noget hemmeligt). Eller restriktioner på eksporten af krypteringsalgoritmer.

Internt i et land er lovgivningen klart præget af etiske overvejelser. Ønsker man en overvågningsstat eller er retten til brevhemmelighed vigtigere end politiets muligheder for at opklare kriminelle handlinger? Problemet er med det nylige terroristangreb mod World Trade Center blevet sat yderligere på spidsen. Eksternt er det nok mere et spørgsmål om det er muligt at håndhæve en given lov i praksis, pga. f.eks. internettet.

Et af de mest aktive lande når det handler om lovgivning mod kryptering er USA – det førnævnte eksempel med PGP viser også dette, Phil Zimmerman skaffede sig faktisk en retssag på halsen [Back, 2001] som dog senere blev opgivet.

6.3.1 Wassenaar-aftalen

Wassenaar-aftalen er et forsøg fra USA på at få andre lande til at indføre samme lovgivning som dem [Koops, 2001]. Aftalen forsøger at lave eksportrestriktioner på stærke krypteringsalgoritmer, dvs. over 56 bit ved symmetrisk kryptering og 512 bit ved asymmetrisk kryptering.

Mange lande, inklusiv Danmark, har underskrevet aftalen, men landene skal selv indarbejde aftalen i deres lovgivning, og 4 danske ministre har senere, i et brev til IT-sikkerhedsudvalget, offentliggjort regeringens holdning på området. De 4 ministre siger at kryptering skal kunne bruges og spredes frit inden for EU og Wassenaar-aftalens lande, men man skal stadig beholde kontrollen over spredning af “følsomme produkter til følsomme brugere” [Weiss et al., 2000].

Men hvordan dette skulle kunne lade sig gøre, og hvem der skal gøre det nævner brevet ikke noget om, og det lader ikke til at der er sket noget konkret. Måske er dette en erkendelse af at den aftale som de har skrevet under på ikke kan bruges i praksis. USA har ikke selv været i stand til at forhindre distribution af de algoritmer der er blevet lavet i landet.

Kapitel 7

Konklusion

Kryptering går ud på at gøre data uforståelig så man kan undgå at uvedkommende kan drage nytte af dem.

Historisk set har kryptologi specielt under krige spillet en stor rolle fordi det kan betyde forskellen mellem liv og død om man kan holde kommunikationen hemmelig – især med indførelsen af letopsnappelige radiosignaler.

Eksempler på klassiske kryptosystemer er ROT_{13}, XOR og DES som bygger på private nøgler, hvilket betyder at man bruger samme nøgle til at kryptere og dekryptere med.

RSA er et eksempel på et moderne kryptosystem der bygger på offentlige nøgler. Offentlige nøgler er velegnede til internettet fordi man ikke løber nogen risiko ved udveksling af nøglerne idet der bruges forskellige nøgler til at kryptere og dekryptere med.

DES, RSA eller varianter af de to ser ikke ud til at ville dominere i fremtiden hver for sig, snarere vil man benytte en kombination. DES udmærker sig ved en sikker og forholdsvis hurtig kryptering og dens svaghed ved nøgleudveksling kan så løses med RSA's tilsvarende styrke på netop dette punkt.

Hvad angår politikkernes anstrengelser for at begrænse krypteringen, må vi slå fast at den bærer præg af uvidenhed og inkompetence: Det kan i praksis ikke lade sig gøre at forhindre udbredelsen af algoritmer eller forhindre forbrydere i at kryptere data så sikkert at politiet ikke kan bryde det. De forsøg på eksport-restriktioner der har været, er en total fiasko som ikke har haft nogen reel virkning. PGP har på trods af eksportrestriktioner under hele programmets eksistens været lettilgængelig for hele verdenen.

Vi mener at man bør søge nye veje for at sikre sig bevismateriale når den vej der har været trådt de seneste år nu ikke har virket. Måske kunne man forestille sig en løsning hvor man efter en dommerkendelse havde pligt til at udlevere sine private nøgler, på samme måde som ved en ransagelse.

Litteratur

Steve Acheson. *Secure Shell FAQ*. 2001. http://www.employees.org/~satch/ssh/faq/ssh-faq-1.html#ss1.1.

Mike Ashley. *The GNU Privacy Handbook*. 1999. http://www.gnupg.org/gph/en/manual.html#AEN385.

Adam Back. *PGP Timeline*. 2001. http://www.cypherspace.org/~adam/timeline/.

Daniel J. Barrett og Richard E. Silverman. *SSH, The Secure Shell: The Definitive Guide*. O'Reilly, 2001. http://safari2.oreilly.com/main.asp?bookname=sshtdg&snode=19.

Friedrich L. Bauer. *Decrypted secrects: methods and maxims of cryptology*. Springer-Verlag, 1997.

Albrecht Beutelspacher. *Cryptology*. The Mathematical Association of America, 1994.

Anna Bjørk. *"Microsoft" sender ny orm. IT-Avisen*, 2001. IT-Avisen, 31/8-2001, http://www.IT-Avisen.dk/nyheder.asp?ID=2582.

DES. *Data Encryption Standard (DES)*. U.S. Department of Commerce, National Institute of Standards and Technology, fips pub 46-2 udgave, 1993. http://www.itl.nist.gov/fipspubs/fip46-2.htm.

D. R. Hankerson, D. G. Hoffman, D. A. Leonard, C. C. Lindner, K. T. Phelps, C. A. Rodger og J. R. Wall. *Coding Theory and Cryptography: The Essentials*. Marcel Dekker, anden udgave, 2000.

Bert-Jaap Koops. *Wassenaar Arrangement*. 2001. http://cwis.kub.nl/~frw/people/koops/cls2.htm#Wassenaar.

Peter Landrock og Knud Nissen. *Kryptologi – fra viden til videnskab*. ABACUS, 1997.

Jacob Langvad. *USA-spionage koster Europa hundreder af milliarder. Information*, 2001. http://www.information.dk/Indgang/VisArtikel.dna?pArtNo=107928.

Alfred J. Menezes, Paul C. van Oorschot og Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996. En online kopi kan ses på http://www.cacr.math.uwaterloo.ca/hac/.

Menneskerettighederne. *FN's Verdenserklæring om Menneskerettighederne*. 1948. http://www.amnesty.dk/index.htm?content=sider/bibliotek/traktat_verdens%erkl.htm.

Eric Myers. *Please use Secure Shell (SSH) instead of Telnet or rsh/rcp/rlogin*. 2000. http://feynman.physics.lsa.umich.edu/~myers/help/SecureShell.html.

Claus Thorhauge. *Politiet har åbnet fem Tvind-computere. Computer World*, 2001a. Computer World Online, 9/8-2001, http://www.cw.dk/vis_artikel.asp?ArticleID=11594.

Claus Thorhauge. *Umuligt at knække Tvind-kryptering. Computer World*, 2001b. Computer World Online, 1/7-2001, http://www.cw.dk/vis_artikel.asp?ArticleID=10920.

Birte Weiss, Pia Gjellerup, Frank Jensen og Hans Hækkerup. *Danish Encryption Policy - Letter from the 4 ministers to the IT-security Council*. 2000. http://www.fsk.dk/cgi-bin/doc-show.cgi?doc_id=25649&doc_type=831&topmen%u=4.

Bilag A

Brev fra “Microsoft”

From: "Microsoft Support" support@microsoft.com

Subject: Invalid SSL Certificate

Hello,

Microsoft Corporation announced that an invalid SSL certificate that web sites use is required to be installed on the user computer to use the https protocol. During the installation, the certificate causes a buffer overrun in Microsoft Internet Explorer and by that allows attackers to get access to your computer. The SSL protocol is used by many companies that require credit card or personal information so, there is a high possibility that you have this certificate installed. To avoid of being attacked by hackers, please download and install the attached patch. It is strongly recommended to install it because almost all users have this certificate installed without their knowledge.

Have a nice day,
Microsoft Corporation

Attachment: sslpatch.exe

Bilag B

Lidt talteori

Talteori beskæftiger sig med egenskaber ved de hele tal $\mathbb{Z}$. Af speciel interesse er divisioner med hele tal fordi resultatet ikke nødvendigvis selv er et helt tal. Hele denne sektion bygger på kap. 4 i [Landrock og Nissen, 1997] (dog er der for at forsimple tingene ikke gjort brug af kongruenser) – hvor beviserne for de sætninger der ikke vises her, også kan ses.

B.1 Division og rester

Først skal nogle grundlæggende begreber og egenskaber slåes fast:

DEFINITION B.1.1 *Et helt tal $a \neq 0$ siges at være divisor i et helt tal b hvis der findes et helt tal q så*

$$b = qa$$

og man siger at a går op i b, hvilket betegnes $a|b$.

SÆTNING B.1.1 (DIVISION MED REST) *For alle $m \in \mathbb{Z}$ og $n \in \mathbb{N}$ findes entydigt bestemte hele tal q og r så*

$$m = qn + r, \quad 0 \leq r < n$$

B.2 Modulo

Tallet r i den ovenstående sætning benævnes den *principale rest* – læg mærke til at $0 \leq r < n$. Man benytter en symbolsk notation til at betegne den („mod“ udtales modulo):

DEFINITION B.2.1 *For vilkårlige hele tal m og n hvor $n > 0$, defineres*

$$m \pmod n = \textit{den principale rest ved division af } m \textit{ med } n$$

F.eks. er $13 \pmod{10} = 3$ og $10 \pmod 3 = 1$. På grund af definitionen er desuden $-17 \pmod 8 = 7$, måske lidt overraskende, fordi $-17 = -3 \cdot 8 + 7$. Nogle basale egenskaber fremgår af følgende sætning:

SÆTNING B.2.1 *For alle $m, k \in \mathbb{Z}$ og $n \in \mathbb{N}$ gælder*

1. $n|m \Leftrightarrow m \pmod n = n$
2. $0 \leq m < n \Rightarrow m \pmod n = m$
3. $(m \pmod n) \pmod n = m \pmod n$
4. $(m + kn) \pmod n = m \pmod n$

Endelig er der er en række regneregler for omformning af udtryk med modulo:

SÆTNING B.2.2 *For alle $a, b \in \mathbb{Z}$ og $n, t \in \mathbb{N}$ gælder*

1. $(a + b) \pmod n = (a \pmod n + b \pmod n) \pmod n$
2. $(a \cdot b) \pmod n = (a \pmod n \cdot b \pmod n) \pmod n$
3. $a^t \pmod n = (a \pmod n)^t \pmod n$

B.3 Primtal

For at klarlægge hvad vi mener med et primtal, fastlægges det i følgende definition:

DEFINITION B.3.1 *Et helt tal $p > 1$ kaldes et* primtal *hvis p kun har de trivielle divisorer 1 og p.*

Positive hele tal der ikke er primtal, kaldes *sammensatte tal*. Ifølge en sætning ved navn aritmetikkens fundamentalsætning kan de kan alle *primfaktoriseres* entydigt, dvs. skrives som et entydigt produkt af primtal; f.eks. er $21 = 3 \cdot 7$. Sætningen er ikke svær at vise, men vil ikke blive taget op her – det vigtige er blot at primfaktorisering er entydig.

Hvis $d|a$ og $d|b$, siges d at være fælles divisor i a og b, og den største fælles divisor for a og b benævnes (a, b). Hvis $(a, b) = 1$, svarende til at kun 1 går op i både a og b, siges a og b at være *indbyrdes primiske*.

B.4 Den kinesiske restsætning

Den kinesiske restsætning dækker over et problem mht. at finde løsninger til et ligningssystem med modulo-udtryk:

SÆTNING B.4.1 *Lad $a, b \in \mathbb{Z}$ og $n, m \in \mathbb{N}, (n, m) = 1$. Så har*

$$x \pmod n = a \pmod n \wedge x \pmod m = b \pmod m$$

en entydig bestemt løsning modulo nm givet ved

$$x \pmod{nm} = (bsn + atm) \pmod{nm}$$

hvor s og t er valgt så $sn + tm = 1$.

B.5 Reducering med $\phi(n)$

Man kan vise at hvis $(a, n) = 1$ så er $a^{\phi(n)} \pmod n = 1 \pmod n$ (sætningen kaldes Eulers sætning). En konsekvens af dette er følgende sætning:

SÆTNING B.5.1 *For alle $a \in \mathbb{Z}$ og $n, t \in \mathbb{N}$ med $(a, n) = 1$ gælder*

$$a^t \pmod n = a^{t \pmod{\phi(n)}} \pmod n$$

Bevis: Der findes ifølge (B.1.1) hele tal q og r sådan at $s = q\phi(n) + r$ hvor $r = s \pmod{\phi(n)}$. Af regnereglerne for modulo-udtryk og Eulers sætning fås så

$$\begin{aligned} a^t \pmod n &= a^{q\phi(n)+r} \pmod n = (a^{q\phi(n)} a^r) \pmod n \\ &= \left(\left((a^{\phi(n)} \pmod n)^q \pmod n \right) \left(a^r \pmod n \right) \right) \pmod n \\ &= \left((1^q \pmod n)(a^r \pmod n) \right) \pmod n \\ &= a^r \pmod n = a^{s \pmod{\phi(n)}} \pmod n \end{aligned}$$

Hvis p er et primtal, er $\phi(n) = p - 1$ da alle tal fra 1 til p er indbyrdes primiske med p. Som et specialtilfælde af Eulers sætning fås da

SÆTNING B.5.2 (FERMATS LILLE SÆTNING) *Lad $a \in \mathbb{Z}$ og p være et primtal med $(p, a) = 1$. Så er*

$$a^{p-1} \pmod p = 1 \pmod p$$