# Kapitel 6 Introduktion till kryptering

## 6.1 Allmänt om kryptering

Kryptering är ett sätt att förvanska eller förställa ett meddelande så att meddelandet i förvanskad form inte går att läsa. Det är bara den avsedda mottagaren som ska kunna återställa meddelandet från dess förvanskade form och ta del av dess innehåll.

Att förvanska brukar i krypteringssammanhang kallas för att kryptera, att återställa brukar kallas att dekryptera. (Engelska: *encrypt* & *decrypt*.)

En krypteringsmetod brukar kallas för ett krypto.

Ett meddelande som förekommer okrypterat sägs förekomma i klartext och vi kommer att beteckna en sådan förekomst med *T* ( = *Text*.) Ett meddelande som förekommer krypterat sägs förekomma i kryptotext och vi kommer att beteckna en sådan förekomst med *C* ( = *Ciphertext*.)

## 6.2 Caesars krypto

En mycket enkel form av kryptering är den så kallade *Caesarrullningen* där varje bokstav ersätts med en annan ett visst fixt antal steg framåt i alfabetet. Om antalet steg är två så ersätts de två sista bokstäverna, *Ä* och *Ö* med de två första *A* och *B*.

**Exempel:**

> Klartexten T = ”JAG ÄTER GRÖT” krypteras enligt Caesarrullning med två steg till kryptotexten C = ”LCI AVGT ITBV”. Här har J ersatts med L (två steg framåt), A har ersatts med C, G med I, Ä med A (Ä är ju den näst sista bokstaven, två steg framåt från Ä finns inte så Ä får ersättas med första boksatven som är A.), och så vidare.
>
> Om vi tolkar bokstäverna som tal, A som 0, B som 1, … Ä som 27 och Ö som 28 så kan vi beskriva aktiviteten att ”flytta fram två steg” som att vi adderar 2 till varje tecken. Ä har numret 27 och om vi adderar 2 till 27 får vi 29 vilket inte representerar någon bokstav. Men här har vi valt att låta Ä ersättas med A som är den första bokstaven som har nummer 0. Detta blir i själva verket addition *modulo* 29 (29 = antalet bokstäver i alfabetet.) På samma sätt får vi B’s ordningsnummer som är 1 om vi adderar 2 till Ö’s ordningsnummer och tar *modulo* 29. Ö har ordningsnumret 28 och $28 + 2 = 30 \equiv 1 (\mathrm{mod}\, 29)$ och 1 är som sagt B’s ordningsnummer.

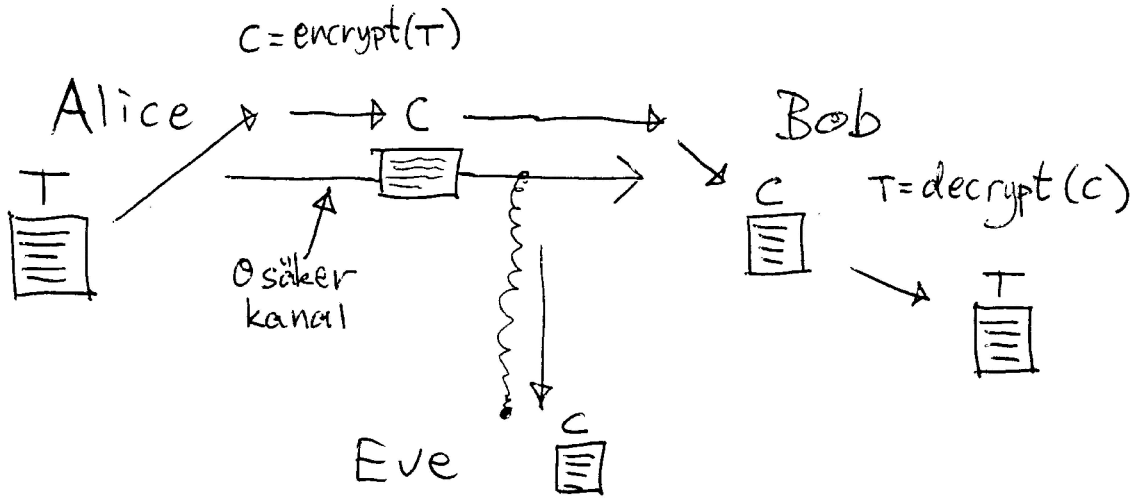
Formen av kryptering som gavs i detta exempel är emellertid alldeles för svag. Om en person som tjuvlyssnar fångar upp tillräckligt många meddelanden kan han knäcka koden genom använda procentuella förekomster av de olika tecknen. I vanlig svenska är e, s, och t de vanligaste bokstäverna. Bokstaven q är mycket ovanlig och om en tjuvlyssnare finner att till exempel bokstäverna g, u, v har samma procentuella förekomster som e, s, och t kan han dra slutsatsen att förmodligen är kryptotexten framställd med hjälp av en Caesarrullning med två steg. Det hjälper inte ens utöka antalet Caesarrullningar och till exempel använda en återkommande cykel med första bokstaven krypteras med 2 steg, andra med säg 5 steg, och tredje med säg 1 steg och resten av bokstäverna i klartexten krypteras enligt den

återkommande cykeln, 2, 5, 1, 2, 5, 1, 2, 5, 1, osv. Detta blir också känsligt för lite mer sofistikerad frekvensanalys.

Att byta rullningssteg i Caesarkryptot kallas Vigenére-kryptot och det är som sagt även det känsligt för en frekvensanalys som dock måste genomföras i ett antal steg. Det enda riktigt säkra kryptot (som man känner till idag) är att ha en lång slumpsekvens av tal, 0, 3, 8, 1, 9, 21, 4, … och krypteringen kan då vara en Caesarrullning med varierande antal steg som föreskrivs av slumpsekvensen. Första tecknet i kryptotexten krypteras med Caesarrullning i 0 steg, andra tecknet i kryptotexten krypteras med Caesarrullning i 3 steg, tredje tecknet i kryptotexten krypteras med Caesarrullning i 8 steg och så vidare. Detta kallas engångskrypto och även om det är fullständigt säkert har det ett annat problem: För att inga mönster ska uppstå måste slumpsekvensen vara minst lika lång som meddelandet. Och när en slumpsekvens är använd kan man inte återanvända den, den får kastas och en ny måste framställas. Detta gör det hela oerhört dyrt. Både sändare och mottagare måste ha en gemensam hemlighet: den långa slumpsekvensen. Hur ska de komma överens om den? Sändare och mottagare måste träffas för att överlämna den i hemlighet och så kan de ju lika gärna träffas för att överlämna det hemliga meddelandet på en gång och aldrig använda slumpsekvensen. En lösning kan då vara att sändare och mottagare möts en gång och då byter slumpsekvenser tillräckligt för att kunna skicka säg 10 eller 100 meddelanden till varandra. Men detta är fortfarande dyrt, det involverar resor och det finns billigare metoder som vi ska se. Dessa metoder blir emellertid inte helt säkra som engångskryptot.

Kryptering används bland annat vid överföring av känslig (hemlig) information. I våra modeller ska vi anta att vi har ett sändare-mottagarepar som vi kommer att benämna *Alice* och *Bob*. Vi ska anta att *Alice* och *Bob* vill kommunicera via en osäker avlyssningsbar kanal utan att *Eve* (som tjuvlyssnar) ska förstå vad som kommuniceras.

Vi illustrerar förhållandena i figur 3.1




*Figur 3.1 De tre aktörerna i kryptering, Alice, Bob (som vill kommunicera ostört) och Eve (som tjuvlyssnar.)*

Baserat på figur 3.1 tänker vi oss följande scenario: Alice vill skicka ett meddelande till Bob. Hon börjar med att framställa meddelandet i klartext, T. Därefter använder hon ett krypto och kalkylerar alltså kryptotexten $C = encrypt(T)$. Därefter sänder hon meddelandet i krypterad form till Bob som tar emot det. Samtidigt som Bob tagit emot meddelandet har Eve avlyssnat sändningen så även hon har tillgång till kryptotexten. Både Bob och Eve har nu tillgång till *C*.

Om Alice och Bob valt ett starkt krypto kommer bara Bob att kunna dekryptera meddelande och extrahera meddelandet i sin oförvanskade form. Bob tar nu fram dekrypteringsfunktionen och beräknar $T = decrypt(C)$.

Krypteringsfunktionen *encrypt* och *decrypt* kan ses som vanliga matematiska funktioner som vi ofta programmerar i en dator. Vi ska nämna en del kring teorin om dessa funktioner.

För det första måste *decrypt* göra det som *encrypt* gör fast *baklänges*. I matematiska termer brukar man kalla det här för att *decrypt* ska vara *encrypt*s invers. Vi kan uttrycka det tecken för tecken. I Caesarkryptot med två steg (som behandlades tidigare) gäller att *encrypt*(A) = C, *encrypt*(B) = D, *encrypt*(E) = G och så vidare. För att *decrypt* ska vara *encrypt*s invers här måste alltså *decrypt* (C) = A, *decrypt*(D) = B och *decrypt*(G) = E och så vidare. Vi kan också formulera detta som att $T = decrypt(encrypt(T))$ för alla meddelanden $T$ och här låter vi alltså funktionerna lite oegentligt verka på hela följder av tecken, men det bereder oss egentligen inga svårigheter.

Att funktionen *decrypt* är funktionen *encrypt*s invers kan ibland formuleras så här: $decrypt^{-1} = encrypt$ och är helt i analogi med andra inverser i matematiken, till exempel är funktionen $f(x) = \arcsin(x)$ inversen till restriktionen av funktionen $g(x) = \sin(x)$ till intervallet $\frac{\pi}{2} \le x \le \frac{\pi}{2}$. I definitionsintervallet gäller $\arcsin(\sin(x)) = x$ vilket är helt i analogi med identiteten $T = decrypt(encrypt(T))$ för alla meddelanden $T$. Ett annat exempel är inversen till exponentialfunktionen $f(x) = e^x$ som är den naturliga logaritmen $g(x) = \ln(x)$. För positiva $x$ gäller identiteten $f(g(x)) = e^{\ln(x)} = x$.

Problemet med kryptering är tvådelat. För de som vill använda kryptering för att nå avskildhet (Alice och Bob) gäller det att använda eller konstruera ett tillräckligt stark och praktiskt användbart krypto. Det finns som sagt helt oknäckbara krypton (engångskryptot) men det är som vi tidigare sett behäftat med vissa svårigheter. För de som tjuvlyssnar (Eve) är problemet: ”Kan jag bryta igenom det krypto som Alice och Bob har valt?”

*Övning:* Betrakta följande kryptotext: ”VWHOEHQW VWKQV”. Den är framtagen med en Caesarrullning. Kan du knäcka vad klartexten är? (Ledning: Tre av de vanligaste bokstäverna i svenska är E, S och T.)

### 6.3 Nycklar

I princip har alla krypton så kallade *nycklar*. En nyckel är en bit information som används i krypteringen och dekrypteringen av ett meddelande. När vi fastställer en nyckel hörande till ett visst krypto har vi bestämt vilka krypterings och dekrypteringsfunktioner vi ska använda. I denna text kommer vi att säga att vi då fastställt en instans av det aktuella kryptot. Nyckeln hörande till instansen av Caesarrullningen i det tidigare exemplet är 2 och denna lilla bit information blir alltså Alice och Bobs gemensamma hemlighet som inte Eve får reda på. Om Eve får tag i nyckeln 2 så är det lätt för Eve att dekryptera kryptotexten och få tag på meddelandet i sin klartext. Nyckeln till instansen av Vigenérekryptot (också exemplifierat ovan) är sekvensen 2, 5, 1. Nyckeln till engångskryptot är den långa slumpsekvens som Alice och Bob kommit överens om i förväg.

Många krypton har hemliga nycklar men vi ska studera en typ av krypton som arbetar med öppna nycklar.

Vi beskriver det hela i lite mer matematiska termer: Om vi identifierar varje bokstav med ett tal mellan 0 och 28 kan krypteringsfunktionen hörande till den tidigare instansen av Caesarrullningen (med nyckeln 2) kan beskrivas så här

$$encrypt(T) = T + 2 (\mathrm{mod}\, 29)$$

Tecknet A, med ordningsnummer 0, krypteras då till 2 som är Tecknet C. Tecknet B, med ordningsnummer 1, krypteras då till 3 som är Tecknet D… Tecknet Å med ordningsnummer 26 krypteras då till 26+2(mod29) = 28(mod29) = 29 som är Tecknet Ö. Tecknet Ä, med ordningsnummer 27 krypteras då till 27+2(mod29) = 29(mod29) = 0 som är Tecknet A och slutligen Tecknet Ö, med ordningsnummer 28 krypteras till 28+2(mod29) = 30(mod29) = 1 som är Tecknet B.

Inversen till denna funktion är $T = decrypt(C)$ och den funktionen ges av att vi subtraherar 2 mod(29) från alla tecken/tal i kryptotexten. Alltså:

$$encrypt(T) = T + 2 (\mathrm{mod}\, 29) \text{ och } decrypt(C) = C - 2 (\mathrm{mod}\, 29).$$

### 6.4 Assymmetriska krypton och RSA-kryptot

Nyckeln är 2 och om man vet om att den är 2 så kan man alltså lätt dekryptera. Alla krypton med hemliga nycklar lider av *nyckeldistributionsproblemet*: Hur ska Alice och Bob komma överens om vilken nyckel de ska använda på ett säkert sätt? Denna överenskommelse *kräver* att de har en säker kanal för kommunikation om vilken nyckel som ska användas. Men om de redan har en säker kanal varför inte bara sända det hemliga meddelandet via den kanalen? Vi har en moment-22 situation. Lösningen är att använda ett så kallat *assymmetriskt* krypto. Ett symmetriskt krypto har samma nyckel som används vid kryptering och dekryptering. Ett assymmetriskt krypto har *olika nycklar* för kryptering och dekryptering. Fördelen är att dekrypteringsnyckeln kan hållas hemlig och krypteringsnyckeln kan publiceras fritt. RSA är ett sådant krypto och skapades på 70-talet och har fått sitt namn av dess uppfinnare som är två dataloger och en matematiker (Rivest, Shamir och Adleman.) Funktionerna i RSA är inte helt olika de funktioner som finns i Caesarrullningen:

**Exempel:**

En liten instans av RSA-kryptot:

$$encrypt(T) = T^7 (\mathrm{mod}\, 10) \text{ och } decrypt(C) = C^3 (\mathrm{mod}\, 10).$$

Nycklarna i denna instans av RSA är $e = 7$ för kryptering och $d = 3$ för dekryptering. Man måste också komma överens om vilket modulo man arbetar med och i den här instansen av RSA är det $N = 10$. Fördelen med skilda nycklar är som sagt att Alice kan skicka krypteringsnyckeln e = 7 till Bob som krypterar sitt meddelande genom att beräkna $C = encrypt(T) = T^7 (\mathrm{mod}\, 10)$. När det är klart skickar han kryptotexten $C$ till Alice och samtidigt snappas kryptotexten $C$ upp av Eve. Alice tar fram den hemliga krypteringsnyckeln $d = 3$ och beräknar $T = decrypt(C) = C^3 (\mathrm{mod}\, 10)$ och får fram

meddelandet i sin okrypterade form. Eve kan inte göra detta eftersom hon inte vet vad dekrypteringsnyckeln $d = 3$ är. *Inte ens Bob kan dekryptera meddelandet när han väl en gång krypterat det.*

Vi ska se på instansen av RSA som gavs i exemplet mera i detalj:

| Klartext Text | Kryptotext encrypt(T) | Klartext (dekrypterad) decrypt(C) | |
|---|---|---|---|
| T | C = T^7(mod 10) | T = C^3(mod 10) | |
| 0 | 0 | 0 | (0^7=0, 0(mod10)=0) |
| 1 | 1 | 1 | (1^7=1, 1(mod 10)=1) |
| 2 | 8 | 2 | (2^7=128, 128(mod 10)=8) |
| 3 | 7 | 3 | (3^7=2187, 2187(mod 10)=7) |
| 4 | 4 | 4 | ... |
| 5 | 5 | 5 | ... |
| 6 | 6 | 6 | ... |
| 7 | 3 | 7 | ... |
| 8 | 2 | 8 | ... |
| 9 | 9 | 9 | ... |

En uppenbar svaghet med denna instans av RSA är att den bara kan kryptera 10 olika tecken, här symboliserade med siffrorna 0, 1, 2, ..., 9. Vi ser också att krypteringen inte är särskilt stark, 0, 1, 4, 5, 6 och 9 byts inte ut alls, det är bara 2, 3, 7 och 8 som blandas. Svagheten beror på att detta är ett överskådligt undervisningsexempel. I verkligheten används RSA för att blanda (kryptera) mycket större, ohyggligt mycket större tal än så här. Storleken på talen innebär att det är mycket svårt att invertera krypteringsfunktionen även om den anges helt öppet eftersom krypteringsnyckeln ju anges öppet så att både Bob och Eve får reda på den.

**Vi tar ett annat exempel:**

E = 17
D = 2753
N = 3233

Denna variant kan kryptera talen 0 till och med N − 1 = 3232. Till exempel gäller

encrypt(123) = (123^17) mod 3233 =
337587917446653715596592958817679803 mod 3233 = 855.

För att dekryptera 855 beräknar vi

decrypt(855) = (855^2753) mod 3233 =


504328889584160687344228991273944666314538783600355093155549675645 01
055628612082559978744245428110054383498654289336384930246451441507 85
172091796654782635307099638035387326500896686074771829745822950342 95
040790358184594095637793858659893688380836028401325097686207669773 96
675332505428260934757351379880632564826393344530925943855624292330 17
519771900169249169128091505960191787601713497254392792156967017899 02
134307146468971279610277181378394586967728986934236524031169321708 92
696176437265213156658331587124597598030425031440068378832461017848 30
717585474547252069688925995892544366701432205469543174002285500923 86
369424448559733330630516073853028632193029135037454719467577767135 79

549652029197905057815328715583920703031595859374936632835486020908306355070445565889631931801193412201782692334410133011648069633402407504695258866987658669006224024102088466507530263953870526631933584734810948761562271260373275973603752373883641480889484380961577570453800810794698006673487779588375828998513279307035335512750904399481789790548993381217329458535447413268056981087263348285463816885048824346588978393334662544540066196452187666947955280230884124659482392751057704911332902568430650522925614273038983208900705151105525061899417123177795157979429711795475296301837843862913977877661298207389072796767202350113992715819642730764074189891904868607481245493157953743771244160143876506914586819640227602776686953090395131496831909732450545234594477256587887692693353918692354818518542420923064996406822184490119135710885424428521120773712238311054554312653073940759278908226060431711333957522660344516452597631618427745904320191345289329932161307440532227470572894812143586831978415597276496357090901215131304157569209798518321041155969357848833665315951327344675243940875769777890849012691532284208094963079297247130442219424390659030814289393029158483087368745078977086921845296741146321155667865528338164806795455941891006950919658990854567980723923708463025535456869192355462995715735879062274586195721721110788286575638597094190776320509783239571346411902500470208485604082175094910771655311765297473803176765820587673140288910328834318508844721164427193903740413155649869959137365162108451137402243351859957665775396936281254253900685526245456141925880943740212888666974410972184534221817198089911953707545542033911964539366461792968165342652234639936742330970183533904623677697670380534264482173582384219251590438148524738896864244370318665419961537791396964900303958760654915244945043600135939277133952101251928572092597887511601959629615690271164318946373426500236310045557180036935865524910000907245183786689564417164907278356281009708545241354696608448116133878065485451517616730860510806578293652410872326366722805400378941086434822675009077826512101372819583165313969830908873174174745359886842985598071851922159700465081060684455953648089224944054276632967459230889848486843586547985051154284401646235269693179937784430217857019197098751629654665130278009966580052178208139317232379013232494682609200819981037684847167874989193694997914824716345060937125650122501953795166897601855087599313367797793952782227323337529580263122665358948205566515289466369032083287680432390611549350954590934066764022586708483376053699867941026204709057156744705653111242862907354888492989983560999636092141128497745861469604028702967070147817949024828290748416008368045866685507604619225209434980471574526881813185085915019485276359650345815364165649316013061330407434457965108380306402240278898042825189094716292266898016684480963645198090510905796513075703792459580744797523712667610114738787421441491548135917439279949695641565386688389171544630561180536972834347021920634899953191764016110392490439179803398975491765395923608511807653184706473318015782074127647875927390874929557168536651859126663738312359458912678709583800022451509424457564874484086877530845395521730636693891702394037184780362774643171470855830491959895146776294392143100245613061114299370005577513397172825491100560089408984196713197091181655429087610900832499783133824078696157849234198629916800867749593407759306602207814943807854996798945399364063685722697422361858411425048372451244655802708591797955910865230997565198382779529457569965742455786883835444236857223681399021261363744082131478483203563615611346287019851423901842909741638620232051039712184983355286308685184282634615027441873586395040422815123995059959836537922272858474220716778366794513436380708657977421985359539316627998878972169596345534633649794922113017661316207477266113107012321403713882270221723233085472679533015079980622538354589480248200431447261915961905260340690619309392907241028494870016717296951770346790997944097506376492963567555800711621827727603182921790350290486090976266285396627024392536890256337101471683274045045830602286763142158159900791642627700054612322919219299719969076901690259464681041412142044724026616582756805241668614733933226595912700645630447416085291672187007045144649793226668732146346749041185886760836840306190695786990096521390675205019744076776510438851519416193184799191349243881528220384647292694460849152999588185988551951490663073117772381322675169458825936387861072430256598091490140178384821401136556784934102431512482864529170314100400120163648299853251663490560537945850894244038552524554777922401046148907527451634251399216373835681414904793203742633730198782540569961916352019389698254478631309773749154478427634532593998741700138163198116645377208944002854850002696859826445621837941167021518477219093392321850877757909593326763114131296193984959261389879016697108810276638623167694057295932538078643444100512138025081797622723797210352196773268441946486164029610598990277105325704570163326134310764177000432371524746263939901189972784536294930363691490088106053123163000901015083933188011668215163893104666659513782749892374556051100401647771682271626727078370122424655126487845492350418521674263831897333234346744490397800178468972640546214802412412583384350170488532060147568786231809409001263241969092252022679880113408073012216264404133887392600523096072386158554965158001034746119792130767224543803671883253708606713311325819922797552277184864847532612430280417794309093899237093805365204646299227955227184864847532612430280417794309093899237093805365204646255147267884961527773274119265709116613580084145421487687310394441054

```
79639308530896880365608504772144592172500126500717068969428154627563
70458838904219177398190648731908014828739058159462227867277418610111
02763247972904122211994117388204526335701759090678628159281519982214
57652796853892517218720090070389138562840007332258507590485348046564
54349837073287625935891427854318266587294608072389652291599021738887
95773647738726574610400822551124182720096168188828493894678810468847
31265541726209789056784581096517975300873063154649030211213352818084
76122990409576427857316364124880930949770739567588422963171158464569
84202455109029882398517953684125891446352791897307683834073696131409
74522985638668272691043357517677128894527881368623965066654089894394
95161912002160777898876864736481837825324846699168307281220310791935
64666840159148582699993374427677252275403853322196852298590851548110
40229657916338257385513314823459591633281445819843614596306024993617
53097925561238039014690665163673718859582772525683119989984646027216
46279764077057074816406450769779869955106180046471937808223250148934
07851137833251073753823403466269553292608813843895784099804170410417
77608463062862610614059615207066695243018438575031762939543026312673
77406936404705896083462601885911184367532529845888040849710922999195
65539701911191919188327308603766775339607722455632113506572191067587
51186812786344197572392195263333856538382400571901025649493339445 19
65959203992392217400247234147190970964562108299547746193228981181286
05556588093851898811812905614274085809168765711911224763288658712755
38928438126611991937924624112632990739867854558756652453056197509891
14578114735771283607554001774268660965093305172102723066635739462334
13638045914237759965220309418558880039496755829711258361621890140359
54234930424749053693992776114261796407100127643280428706083531594582
30594632682786127020335698034614324569702148437 5 mod 3233 = 123
```

Talet 123 krypteras alltså till 855. (Och 855 dekrypteras alltså tillbaka till 123.) Beräkningarna involverar ett tal med cirka 4200 siffror, det blir svårt att hantera men det är det som är kruxet med RSA. Det är bara den hemliga dekrypteringsnyckeln som kan nysta upp sifferhärvan på ett korrekt sätt.

Vi ska beskriva hur man finner nycklar och beräkningstal hörande till RSA-kryptot, det vill säga hur man skapar instanser av RSA. Vi väver in de två exemplen vi stött på i beskrivningen

*Beskrivning av konstruktion av en RSA-instans*

1. Välj två primtal, till exempel $p = 2$ och $q = 5$. Bilda produkten $pq = 10$, detta är *N*, (det som vi sen tar modulo med). Det anger också hur många olika symboler vi kan kryptera. (Bara 10 här alltså.)

2. Välj krypteringstalet $e > 1$, så att *e* och $(p-1)(q-1)$ är relativt prima. I vårt exempel är $(p-1)(q-1) = (2-1)(5-1) = 4$. Talet 7 är ett primtal och 7 och 4 är relativt prima. Vi väljer alltså $e = 7$.

3. Välj dekrypteringstalet *d* så att $de \equiv 1(\mathrm{mod}(p-1)(q-1))$. Det är lätt att göra med en dator. I vårt exempel gäller att $(p-1)(q-1) = 4$, som sagt, och vi behöver bara finna ett tal *x* som har egenskapen att $(x(p-1)(q-1)+1)/e = (4x+1)/7$ blir ett heltal. Detta heltal duger sedan som *d*. I exemplet ser vi att d = 3 och vad det svarar mot för *x* kan vi se genom att lösa ut *x* ur $(4x+1)/7 = 3 \Leftrightarrow 4x+1 = 21 \Leftrightarrow x = 5$. En dator kan programmeras för att successivt finna *x* genom testning. Ger $x = 1$ att $(4x+1)/7$ blir heltal? Om inte så fortsätt testa $x = 2$ osv. Vi beräknar produkten *de* för att kontrollera: $de = 3 \cdot 7 = 21 = 20 + 1 = 5 \cdot 4 + 1 \equiv 1(\mathrm{mod}\,4)$. Alltså gäller $de \equiv 1(\mathrm{mod}(p-1)(q-1))$.

4. Krypteringsfunktionen blir nu $C = \mathrm{encrypt}(T) = T\,\hat{}\,e(\mathrm{mod}\,N)$, i vårt fall $C = \mathrm{encrypt}(T) = T\,\hat{}\,7(\mathrm{mod}10)$.

5. Dekrypteringsfunktionen blir nu $T = \text{decrypt}(C) = C\text{^}d \pmod N$, i vårt fall $T = \text{decrypt}(C) = C\text{^}3 \pmod{10}$.

Detta är det första exemplet på RSA som gav svag kryptering där bara 2, 3, 7 och 8 krypterades. Den nyckel som kan publiceras är (*e*, *pq*), i exemplet (7, 10). Den hemliga nyckeln är *d*, i exemplet 3.

Vi ser på de bakomliggande delarna av det andra exemplet (det där med 4197 siffror):

1. De två primtalen är här p = 61 och q = 53. Produkten är $N = pq = 61 \cdot 53 = 3233$.

2. Krypteringstalet $e > 1$ väljs så att *e* och $(p-1)(q-1)$ är relativt prima. I vårt exempel är $(p-1)(q-1) = (61-1)(53-1) = 60 \cdot 52$. Talet 17 är ett primtal och 17 och 17 delar inte 60 eller 52 således är talen 17 och $60 \cdot 52$ relativt prima. Vi väljer alltså $e = 17$.

3. Välj dekrypteringstalet *d* så att $de \equiv 1 (\text{mod}(p-1)(q-1))$. I vårt exempel gäller att $(p-1)(q-1) = (61-1)(53-1) = 3120$. Talet *x* i det här fallet får vi av beräkningen $(x(p-1)(q-1)+1)/e = (3120x+1)/17 = 2753 \Leftrightarrow 3120x + 1 = 17 \cdot 2753$ $\Leftrightarrow 3120x = 46801 - 1 \Leftrightarrow x = 46800/3120 = 15$. Vi beräknar produkten *de* för att kontrollera: $de = 17 \cdot 2753 = 46801 = 46800 + 1 = 15 \cdot 3120 + 1 \equiv 1 (\text{mod}\, 3120)$. Alltså gäller $de \equiv 1 (\text{mod}(p-1)(q-1))$.

4. Krypteringsfunktionen blir nu $C = \text{encrypt}(T) = T\text{^}e \pmod N$, i vårt fall $C = \text{encrypt}(T) = T\text{^}17 \pmod{3233}$.

5. Dekrypteringsfunktionen blir nu $T = \text{decrypt}(C) = C\text{^}d \pmod N$, i vårt fall $T = \text{decrypt}(C) = C\text{^}2753 \pmod{3233}$.

Den offentliga krypteringsnyckeln blir (*e*, *pq*) = (17, 3233). Den hemliga dekrypteringsnyckeln blir *d* = 2753.

Detta exempel är hämtat från webben: http://world.std.com/~franl/crypto/rsa-example.html, (*Copyright © Francis Litterio*)

*Övning 3.1.* Verifiera att talen $p = 83$, $q = 89$, $N = 7387$, $e = 17$ och $d = 849$ uppfyller kraven som ställs i RSA-kryptot. (Dessa tal kommer att ligga till grund för laboration 2.)

*Övning 3.2.* Välj primtalen $p = 5$ och $q = 7$. Då blir $N = 5 \cdot 7 = 35$. Finn ett *e* och ett *d* som fungerar tillsammans med detta val av *p* och *q* och gör en tabell liknande den som finns i exemplet på RSA-kryptot ovan (där *p* var 2 och *q* var 5.) Hur många symboler kan krypteras med denna instans av RSA-kryptot?