Introduktion till kryptering

Robert Nyqvist

24 februari 2025

Innehåll

Kapitel

1

Termer och begrepp inom kryptologi

1.1 Terminologi

Ordet *krypto* härstammar från det grekiska ordet *kryptos* (κρυπτός) som betyder *gömd* eller *hemlig*. Läran om matematiska metoder för sekretess kallas *kryptologi*. Denna delas in i dels *kryptografi* som behandlar konstruktion av krypteringsmetoder och dels *kryptoanalys* som behandlar metoder för att knäcka krypton.

Avsändare: Den part i en kommunikation som vill sända ett meddelande på ett säkert sätt, d v s så att endast tänkt *mottagare* kan läsa det. Kallas oftast *Alice*.

Mottagare: Den andra parten i kommunikationen och den som är förtrogen med Alice. Mer känd under namnet *Bob*.

Antagonist: Boven i dramat och skälet till att Alice behöver kryptera sitt meddelande. Kallas oftast *Eve*. Namnet kommer från det engelska ordet för tjuvlyssna, *eavesdrop*. Alice vill inte att Eve ska kunna läsa hennes meddelande till Bob.

Klartext: Det okrypterade och därmed läsbara meddelandet som Alice vill skicka till Bob. Vi antar att Alice är tvungen att skicka sitt meddelande till Bob via en osäker kanal som lätt kan avlyssnas av Eve, som t ex post eller telefon. Därför måste Alice *kryptera* meddelandet innan hon sänder det till Bob.

Kryptogram eller **kryptotext:** Den *krypterade* versionen av Alice klartext, som är så mycket ”förvanskad” att Eve förhoppningsvis inte kan läsa det eller i bästa fall inte inom rimligt tid få ut någon information av det. Det är denna text som Alice sänder till Bob.

Kryptering: Den process som Alice använder för att omvandla sin klartext till ett kryptogram.

Dekryptering: Den process som Bob använder för att omvandla det mottagna kryptogrammet tillbaka till klartext

Nyckel: Parameter som i detalj styr en krypterings- eller en dekrypteringsalgoritm.

Forcering: Det arbete som Eve lägger ned för att *knäcka* Alice och Bobs krypto. Hennes mål kan vara att finna vad som står i ett specifikt meddelande eller vilka nycklar de använder.

Kryptosystem eller **krypto:** En uppsättning av algoritmer för kryptering och dekryptering samt beskrivning av mängden av alla tillåtna klartexter, mängden av alla möjliga kryptogram och mängden av alla möjliga nycklar. En annan benämning är *chiffer*.

Symmetriskt krypto: Kryptosystem där krypterings- och dekrypteringsnycklarna är lika eller där den ena nyckeln enkelt kan härledas vid kännedom om den andra. Alice och Bob måste hålla sina nycklar hemliga.

Asymmetriskt krypto: Ett kryptosystem som inte är symmetriskt, d v s nycklarna är så olika att även om man känner t ex krypteringsnyckeln är det tidskrävande att bestämma dekrypteringsnyckeln.

Blockkrypto: Kryptosystem där flera tecken i rad i klartexten, s k *block*, krypteras samtidigt. Antalet tecken i varje block är oftast fixerat.

Man kan även dela in kryptosystem i undergrupper. Bland de symmetriska krypton finns t ex *transpositionskrypton*, *flödeskryton* och *polyalfabetiska krypton*.

1.2 Kryptografi

Vid design av nya kryptosystem vill man att dess styrka gentemot forceringsförsök främst ska beror på att man håller nycklarna hemliga och **inte** själva krypteringsalgoritmen. Detta förhållningssätt kallas *Kerckhoffs princip*, efter upphovsmannen Auguste Kerckhoffs som formulerade den 1883.

Vid sidan om kryptering som lösning på problemet att uppnå sekretess vid kommunikation, har flera andra liknande problem uppstått. Några av dessa följer nedan.

- Autentisering: Hur avgör man om ett meddelande är från den person som står som avsändare?
- Nyckelutväxling: Hur utbyter man de nycklar som behövs vid kryptering utan att behöva träffas eller använda kurirer?
- Icke-förnekande: Hur ser Bob till att Alice i efterhand inte kan förneka att hon sänt ett meddelande, t ex en orderbekräftelse?
- Slantsingling: Hur går två personer tillväga för att singla slant på distans, utan någon av dem kan fuska?
- Hemlighetsdelning: Hur fördelar man fragment av en hemlighet till n personer, så minst k av dem måste träffas för att kunna återskapa hemligheten?
- Nollkoll-bevis: Hur övertygar du någon om att du har löst ett svårt problem, utan att avslöja själva lösningen?
- Elektroniska pengar: Hur ska ett system för digitala pengar vara konstruerat, så att man inte kan kopiera pengar eller återanvända dem? Hur uppnår man anonymitet och kopplar varje elektronisk peng till sin rättmätiga ägare?
- Digital röstning: Hur genomför man ett val så rättsäkert som möjligt? Man vill t ex säkerställa att varje röst räknas korrekt och att ingen kan rösta flera gånger.

1.3 Kryptoanalys

Det finns i princip fyra olika typer av forceringsattacker på ett krypto beroende på hur mycket information Eve har tillgång till.

Endast kryptogrammet känt: Eve har endast tillgång till ett eller flera kryptogram.

Känd klartext: Eve har tillgång till ett eller flera kryptogram och dess motsvarande klartext.

Valbar klartext: Eve kan välja egna klartexter, kryptera dessa, och försöka dra slutsatser om vilken nyckel som används utifrån de kryptogram som erhålls.

Valbart kryptogram: Eve kan välja egna kryptogram, dekryptera dessa, och försöka dra slutsatser om vilken nyckel som används utifrån de ”klartexter” som erhålls.

Vid design av ett kryptosystem studerar man hur stort motståndskraft det har emot de olika typerna av attacker. Eve har ofta en av följande mål med sitt forceringsförsök. Hon vill (a) läsa ett specifikt meddelande (b) veta Alice och Bobs nycklar och på så sätt kunna läsa all deras kommunikation (c) förändra ett meddelande från Alice och på så sätt lura Bob (d) låtsas vara Alice i avsikten att få Bob att tro att han kommunicera med Alice.

Kapitel

2 Symmetriska kryptosystem

2.1 Vernamchiffer

Detta krypto är uppkallat efter en av kryptots skapare, Gilbert S. Vernam. Kryptot kallas också *engångskrypto* (eng. *one-time pad*), *diplomatchiffer* och *blankettchiffer*. Egentligen var det Frank Miller som 1882 konstruerade kryptosystemet. Vernam uppfann kryptot igen 1917 och då i form av en kryptomaskin där nyckelflödet representerades av tecken på en pappersremsa, vars ändar var sammanfogade. På det sättet återanvänds samma nyckel när pappersremsan roterat ett varv. Joseph O. Mauborgne insåg att om man låter nyckeln vara slumpmässigt genererad så gör det kryptot näst intill omöjligt att forcera.

2.1.1 Flödeskrypton

Låt $x_1x_2x_3\ldots$ beteckna en följd av klartextsymboler, t ex bokstäver, siffror och skiljetecken, och låt $y_1y_2y_3\ldots$ beteckna motsvarande följd av kryptogramsymboler. Vidare låt E_k beteckna krypteringsfunktionen, där k är aktuell krypteringsnyckel. I ett *flödeskrypto* eller *strömkrypto* använde en följd av nycklar $k_1, k_2, k_3, \ldots$ för att kryptera respektive block enligt $y_i = E_{k_i}(x_i)$, d v s den i:te symbolen i kryptogrammet fås genom att man krypterar den i:te symbolen i klartexten med den i:te nyckeln enligt

$$y_1 = E_{k_1}(x_1), \quad y_2 = E_{k_2}(x_2), \quad y_3 = E_{k_3}(x_3)\ldots.$$

Ett praktiskt problem är hur detta flöde av nycklar ska genereras. En lösning är att använda en pseudoslumptalsgenerator.

2.1.2 Kryptering och dekryptering

Låt n vara antalet möjliga olika symboler, t ex bokstäver, siffror och skiljetecken, i det aktuella alfabetet. Varje symbol kodas som ett av heltalen i mängden

$$\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\},$$

så att varje heltal motsvarar exakt en symbol. Klartexten kan då kodas till en följd av element $(x_1, x_2, x_3, \ldots)$, där $x_i \in \mathbb{Z}_n$. Låt följden $(k_1, k_2, k_3, \ldots)$, där $k_i \in \mathbb{Z}_n$, beteckna krypteringsnyckeln. Motsvarande kryptogram $(y_1, .y_2, y_3, \ldots)$, där $y_i \in \mathbb{Z}_n$ fås med krypteringsfunktionen

$$y_i = E_{k_i}(x_i) \equiv x_i + k_i \pmod{n}.$$

Vid dekryptering är y_i och k_i kända medan x_i är obekant, d v s lös ut x_i ur kongruensen ovan.

Exempel 2.1. Låt oss koda bokstäverna i det svenska alfabetet enligt följande.

$$\texttt{a} \leftrightarrow 0, \quad \texttt{b} \leftrightarrow 1, \quad \texttt{c} \leftrightarrow 2, \quad \ldots, \quad \texttt{ö} \leftrightarrow 28.$$

Eftersom alfabetet innehåller 29 bokstäver är $n = 29$. Antag att Alice vill skicka meddelandet

”Hej Bob! Ditt uppdrag är. . . ”

till Bob. De har tidigare kommit överens om att används nyckelflödet

$$(k_1, k_2, \ldots) = (16, 5, 11, 22, 23, 14, 15, 5, 23, 22, 27, 1, 16, 13, 7, 3, 9, 2, 14, \ldots).$$

Bortser vi i klartexten från skiljetecken samt likställer versaler och gemener går krypteringen till enligt följande.

Klartext	h	e	j	b	o	b	d	i	t	t	u	p	p	d	r	a	g	ä	r	...
Kodning	7	4	9	1	14	1	3	8	19	19	20	15	15	3	17	0	6	27	17	...
Nyckel	16	5	11	22	23	14	15	5	23	22	27	1	16	13	7	3	9	2	14	...
Summa	23	9	20	23	37	15	18	13	42	41	47	16	31	16	24	3	15	29	31	...
Modulo	23	9	20	23	8	15	18	13	13	12	18	16	2	16	24	3	15	0	2	...
Kryptogram	X	J	U	X	I	P	S	N	N	M	S	Q	C	Q	Y	D	P	A	C	...

Vi har t ex att den femte bokstaven i klartexten är o, som kodas till heltalet $x_5 = 14$. Vidare är $k_5 = 23$ den femte nyckeln i nyckelflödet. Alltså är

$$y_5 \equiv x_5 + k_5 \equiv 14 + 23 \equiv 37 \equiv 8 \pmod{29},$$

d v s i detta fall krypteras o som I. Kryptogrammet lyder XJUXIPSNNMSQCQYDPAC.

Notera att vad en bokstav krypteras som beror på vilken position den har i klartexten, då resultatet beror på vilken nyckel den paras ihop med. För att kunna dekryptera måste Bob veta hela nyckelflödet. ◇

Exempel 2.2. Vid implementation av Vernamkryptot i en dator kan man utnyttja att en klartext lagras som en *bitström*, d v s en följd av ettor och nollor. Alltså har vi att

$$k_1, x_i, y_i \in \mathbb{Z}_2 = \{0, 1\} \quad \text{och} \quad y_i \equiv x_i + k_i \pmod{2}.$$

Antag att vi vill kryptera klartexten ”Troll!”. I åtta bitars ASCII, se avsnitt A.1, kodas klartexten decimalt till

$$84, 114, 111, 108, 108, 33$$

och binärt till

$$01010100\ 01110010\ 01101111\ 01101100\ 01101100\ 00100001.$$

Klartexten består av 48 bitar. Antag att nyckeln ges av

$$101101110001110111001100110110011111010000010110.$$

Kryptering ger oss följande resultat.

Klartext		010101000111001001101111011011000110110000100001
Nyckel	+	101101110001110111001100110110011111010000010110
Kryptogram		111000110110111110100011101101011000010000110111

Notera att

$$0 + 0 \equiv 1 + 1 \equiv 0 \pmod 2 \quad \text{och} \quad 0 + 1 \equiv 1 + 0 \equiv 1 \pmod 2.$$

Eftersom ASCII innehåller en del icke-utskivbara tecken kodar vi inte kryptogrammet till en textsträng. Eftersom $1 \equiv -1 \pmod 2$, så ges dekrypteringsfunktionen av

$$x_i \equiv y_i - k_i \equiv y_i + k_i \pmod 2,$$

d v s exakt samma nyckel och funktion används vid kryptering och dekryptering. ◊

2.1.3 *Perfekt sekretess*

Vernamchiffret är idag det enda kända krypto som är matematiskt bevisad att ha s k *perfekt sekretess*. Det innebär att kryptot är så säkert att även om man vet kryptogrammet, så ger inte det mer information om klartexten. Det bevisades första gången 1948 av Claude Shannon.

Beviset grundar sig på antagandet att varje nyckel k_i i nyckelflödet $k_1, k_2, k_3, \ldots$ är slumpmässigt valt. Det innebär att om vi genererar nycklarna på ett icke-slumpmässigt sätt, uppnår vi inte perfekt sekretess. Av samma skäl kan vi inte återanvända ett nyckelflöde, härav namnet engångkrypto. Det är alltså inte ett praktiskt krypto. Men just på grund av sin höga säkerhet har det använts, t ex under kalla kriget.

2.2 Data Encryption Standard (DES)

År 1973 efterlyste National Bureau of Standards (NBS) ett kryptosystem som skulle användas som amerikansk standard. IBM lämnade 1974 in sitt förslag som de kallade LUCIFER. Den amerikanska nationella säkerhetsmyndigheten National Security Agency (NSA) utvärderade LUCIFER och godkände det efter vissa modifikationer, då under namnet DES. NBS publicerade 1975 en fri licens för DES och 1977 blev den officiell standard. DES är ett blockkrypto och har använts mycket i den kommersiella sektorn, eftersom som den är snabb.

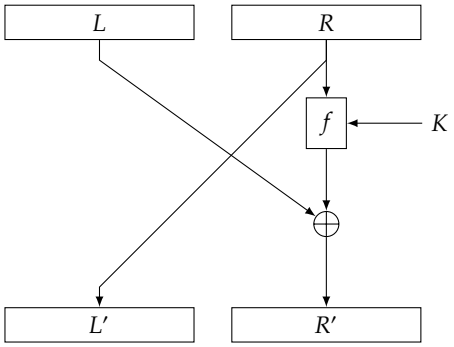
2.2.1 *Feistelsystem*

En central komponent i DES är det så kallade *Feistelsystemet*. Uppkallat efter Horst Feistel, som var med och utvecklade LUCIFER. Ett *Feistelsystem* beskriver hur varje block krypteras. I DES upprepas denna procedur 16 gånger. Utdata från föregående runda blir indata till nästa. Varje block delas upp i två lika delar, som betecknas L och R, där L är det vänstra och R det högra delblocket. Låt K representera krypteringsnyckeln samt låt L' och R' beteckna vänster och höger delblock av resutlatet från Feistelsystemet. Antag att L, R, L' och R' är bitsträngar av längd n och att K är en bisträng av längd m, d v s $L, R, L', R' \in \mathbb{Z}_2^n$ och $K \in \mathbb{Z}_2^m$. Ett Fiestelsystem ges algebraiskt av

$$L' = R \quad \text{och} \quad R' = L \oplus f(R, K),$$

där $f \colon \mathbb{Z}_2^n \times \mathbb{Z}_2^m \to \mathbb{Z}_2^n$ är en valfri funktion. Ett Feistelsystem kan illusterars enligt figur 2.1. Additionen $a \oplus b$ definieras på följande sätt. Låt x och y vara bitsträngar av samma längd, d v s

$$x = x_1 x_2 \ldots x_n \quad \text{och} \quad y = y_1 y_2 \ldots y_n,$$



Figur 2.1. Feistelsystem

där x_i och y_i är element i $\mathbb{Z}_2$, för $i = 1, 2, \ldots, n$. Då betecknar

$$z = x \oplus y = z_1 z_2 \ldots z_n$$

s k *bitvis addition modulo 2*, d v s

$$z_i \equiv x_i + y_i \pmod 2 \quad \text{där} \quad z_i \in \mathbb{Z}_2.$$

Andra vanliga skrivsätt är $a \veebar b$ och x XOR y, d v s *exklusivt eller*.

Exempel 2.3. Vi har t ex att $\mathtt{10010} \oplus \mathtt{01110} = \mathtt{11100}$, $\mathtt{0101} \oplus \mathtt{0101} = \mathtt{0000}$ och

$$\mathtt{00110011101011} \oplus \mathtt{00110111101011} = \mathtt{00000100000000}.$$

I det sista exemplet skiljer sig de två strångarna åt i endast en bit, nämligen den sjätte. Vi kan lika gärna betrakta $a \oplus b$ som skillnaden mellan a och b. Det finns många olika par av bitsträngar som har samma skillnad, som t ex

$$\mathtt{11011} \oplus \mathtt{10001} = \mathtt{01010} \quad \text{och} \quad \mathtt{00111} \oplus \mathtt{01001} = \mathtt{01010}.$$

Givet en bitsträng a och en önskad skillnad d, så är $b = a \oplus d$. ◊

2.2.2 *Kryptering*

Varje klartextblock m består av 64 bitar. Nyckeln K består även den av 64 bitar, men var åttonde bit är en paritetsbit, sådan att varje block om 8 bitar ska innehålla ett udda antal 1:or. Alltså finns det 2^{56} olika nycklar. Krypteringen sker i följande tre steg.

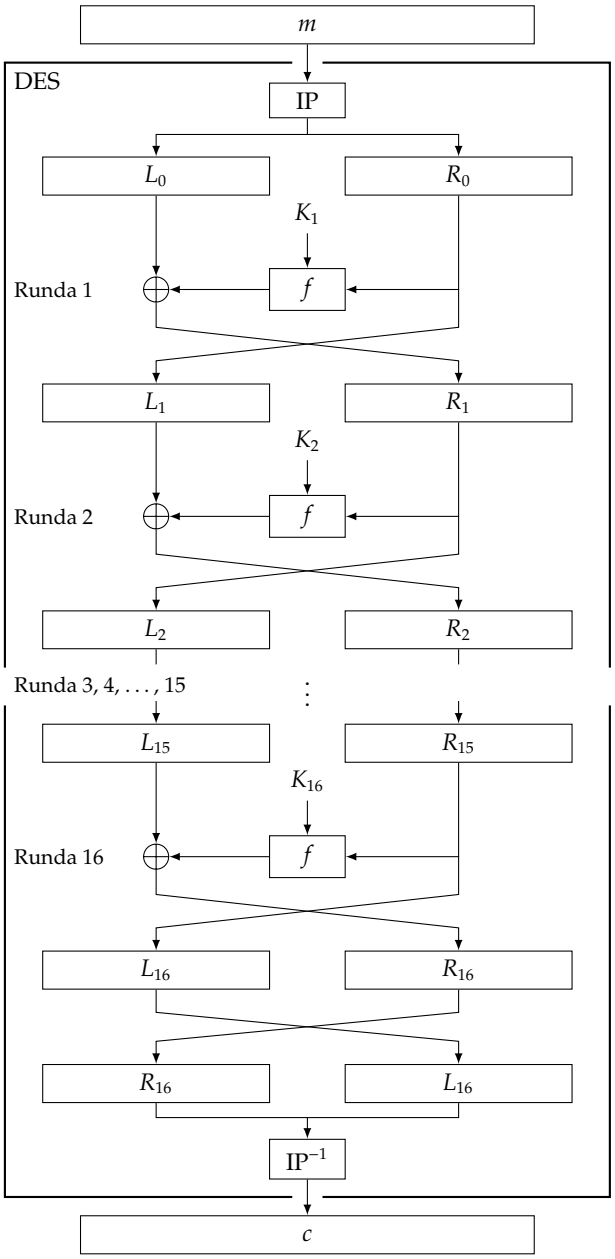
1. Permutera bitarna i m med funktionen IP och sätt $m_0 = \mathrm{IP}(m)$. Låt L_0 och R_0 vara de 32 första respektive sista bitarna i m_0, d v s $m_0 = L_0 R_0$. Definitionen av permutationen IP återkommer vi till längre fram.

2. För $i = 1, 2, \ldots, 16$ upprepa Feistelsystemet

$$L_i = R_{i-1} \text{ och } R_i = L_{i-1} \oplus f(R_{i-1}, K_i),$$

   där K_i är en bitsträng av längd 48, erhållen från nyckeln K, se avsnitt 2.2.5.

3. Kryptogrammet ges av $c = \mathrm{IP}^{-1}(R_{16} L_{16})$, där IP^{-1} är inversen till IP från första steget. Notera ordningen på blocken R_{16} och L_{16}, d v s de kastas efter sista rundan i Feistelsystemet.



Figur 2.2. DES

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

Tabell 2.1. Funktionen IP

Kryptogramblock består, precis som klartextblocket, av 64 bitar. Krypteringsalgoritmen för DES kan illustreras med ett flödesdiagram, se figur 2.2.

Beteckningen IP är en förkortning av *Initial Permutation*. Tilldelningen $m_0 = \mathrm{IP}(m)$ permuterar bitarna i klartextblocket m enligt tabell 2.1. Det innebär att 1:a biten i m_0 är den 58:e biten i m, den 2:a biten i m_0 är den 50:e biten i m, ..., den 64:e biten i m_0 är den 7:e biten i m. Eftersom IP är en permutation så existerar inversen IP^{-1}. Tabell 2.1 kan också användas för att bestämma $c = \mathrm{IP}^{-1}(R_{16}L_{16})$ på så sätt att 1:a biten i $R_{16}L_{16}$ är 58:e biten i c, o s v.

2.2.3 *Dekryptering*

Dekryptering utförs med samma algoritm som vid kryptering, fast man använder rundnycklarna $K_1, K_2, \ldots, K_{16}$ i omvänd ordning. Från $\mathrm{IP} \circ \mathrm{IP}^{-1} = \mathrm{id}$ följer att

$$\mathrm{IP}(c) = \mathrm{IP}(\mathrm{IP}^{-1}(R_{16}L_{16})) = \mathrm{id}(R_{16}L_{16}) = R_{16}L_{16}.$$

Under krypteringen erhåller vi

$$L_{16} = R_{15} \text{ och } R_{16} = L_{15} \oplus f(R_{15}, K_{16})$$

efter den sista iterationen av Feistelsystemet. Låt $L'_iR'_i$ beteckna blocket efter runda i under dekrypteringen. Vid dekryptering är $R_{16}L_{16}$ det block som är indata till första iterationen av Feistelsystemet, d v s $L'_0 = R_{16}$ och $R'_0 = L_{16}$. Efter runda 1 är

$$L'_1 = R'_0 = L_{16} = R_{15}$$

och

$$\begin{aligned} R'_1 &= L'_0 \oplus f(R'_0, K_{16}) = R_{16} \oplus f(R_{15}, K_{16}) \\ &= L_{15} \oplus f(R_{15}, K_{16}) \oplus f(R_{15}, K_{16}) = L_{15}. \end{aligned}$$

Med t ex induktion följer att

$$L'_i = R_{16-i} \text{ och } R'_i = L_{16-i}$$

där $i = 1, 2, \ldots, 16$, se övningsuppgift 12. Efter den sextonde och sista rundan har vi blocket $L'_{16}R'_{16} = R_0L_0$. Slutligen kastar vi om ordnigen mellan de två delblocken och applicerar IP^{-1}. Det ger att

$$\mathrm{IP}^{-1}(R'_{16}L'_{16}) = \mathrm{IP}^{-1}(L_0R_0) = \mathrm{IP}^{-1}(m_0) = \mathrm{IP}^{-1}(\mathrm{IP}(m)) = m.$$

Vi har visat att algoritmen fungerar även för dekrytpering, under förutsättning att vi använder rundnycklarna $K_1, K_2, \ldots, K_{16}$ i omvänd ordning. Notera att vi aldrig behöver kräva att funktionen f ska vara inverterbar, dekrypteringen fungerar ändå.

2.2.4 *Funktionen f*

Funktionen $f\colon \mathbb{Z}_2^{32} \times \mathbb{Z}_2^{48} \to \mathbb{Z}_2^{32}$ beskrivs enklast med en algoritm. Processen för att beräkna $f(R_{i-1}, K_i)$ kan delas i fyra steg.

1. Först expanderar vi de 32 bitarna i R_{i-1} till 48 bitar med funktionen $E(R_{i-1})$, vilken ges av tabell 2.2. Tabellen avläses på så sätt att 1:a biten i $E(R_{i-1})$ ges av den 32:a biten i R_{i-1} är, den 2:a biten i $E(R_{i-1})$ ges av 1:a biten i R_{i-1}, o s v. Med andra ord samma princip som hur tabell 2.1 definierar funktionen IP.

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

Tabell 2.2. Expansionsfunktionen E

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

Tabell 2.3. Permutationen efter S-box

2. Låt $B = E(R_{i-1}) \oplus K_i$ och dela sedan upp den 48 bitar långa bitsträngen B i block B_j av längd 6 bitar vardera, d v s $B = B_1B_2 \dots B_8$.

3. För varje $B_j = b_1b_2 \dots b_6$ låt C_j vara den binära representationen av heltalet på rad $(b_1b_6)_2$ och kolumn $(b_2b_3b_4b_5)_2$ i S-box j, se tabell 2.4. Numrering av rader och kolumner inleds med 0. Varje C_j repsenteras alltid med 4 bitar. Varje S-box kan betraktas som en funktion. Låt S_i beteckna funktionen som representera S-box i, d v s $C_j = S_j(B_j)$.

   Om t ex $B_1 = \texttt{101011}$, så ska vi i S-box 1 plocka ut elementet på rad $11_2 = 3$ och kolumn $0101_2 = 5$, d v s fjärde raden och sjätte kolumnen. Alltså är $C_1 = 9 = 1001_2$ och $S_1(\texttt{101011}) = \texttt{1001}$.

4. Bitsträngen $C_1C_2 \dots C_8$ har längden 32 och bitarna i denna permuteras enligt tabell 2.3 på sätt som tidigare. Resultatet returneras som $f(R_{i-1}, K_i)$.

I figur 2.3 illusteraras algoritmen som ett flödesdiagram.

Exempel 2.4. Om

$$R = \texttt{00110101001011110010001101011000}$$

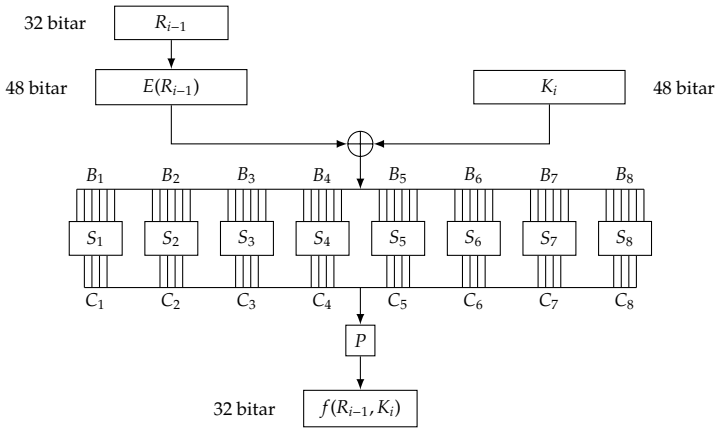
och

$$K = \texttt{011000010110001001100011011001000110010101100110},$$

så är

$$f(R, K) = \texttt{01111100010001101000011000101110}.$$

Detaljerna lämnas som övning till läsaren. ◊



Figur 2.3. Funktionen f i DES

S-box 1

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

S-box 2

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

S-box 3

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

S-box 4

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

S-box 5

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

S-box 6

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

S-box 7

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

S-box 8

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

Tabell 2.4. S-box (DES)

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

(a) Permutation av nyckeln K

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Skift	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

(b) Antal steg för LS_i

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

(c) Permutation för K_i

Tabell 2.5. Nyckelgenerering (DES)

2.2.5 *Nyckelgenererig*

Kvar är konstruktionen av rundnycklarna $K_1, K_2, \ldots, K_{16}$. Vi utgår från nyckeln K, som består av 64 bitar. Steg för steg går vi tillväga på följande sätt.

1. Bitarna i K permuteras enligt tabell 2.5 a. Notera att var åttonde bit ignoreras, då de är paritetsbitar och egentligen inte en del av själva nyckeln. Beteckna resultatet med C_0D_0, där C_0 och D_0 båda består av 28 bitar vardera.

2. För $i = 1, 2, \ldots, 16$ sätt $C_i = \mathrm{LS}_i(C_{i-1})$ och $D_i = \mathrm{LS}_i(D_{i-1})$, där funktionen LS_i betecknar ett cykliskt skift åt vänster. Antal steg i funktionen $\S_i$ ges av tabell 2.5 b. Vi har t ex att $\mathrm{LS}_2(\texttt{101100}) = \texttt{011001}$ och $\mathrm{LS}_8(\texttt{1011010}) = \texttt{1101011}$.

3. Varje bitsträng C_iD_i är av längd 56. Från C_iD_i väljs 48 bitar ut och permuteras enligt tabell 2.5 c. Resultatet är nyckeln K_i, där $i = 1, 2, \ldots, 16$.

Notera att nyckelkostruktionen är oberoende av krypteringsalgoritmen och därför kan därför implementeras som en separat funktion, vilken anropas innan själva krypteringen eller dekrypteringen.

2.3 Kryptografiska hackfunktioner

Låt X och Y vara icke-tomma mängder. Antag att $f : X \to Y$ är en funktion sådan att det är utförbart att bestämma bilden $y = f(x)$ för varje urbild $x \in X$. Med *utförbar* menas här att det i praktiken går att genomföra beräkningen, d v s det finns någon acceptabelt snabb algoritm för ändamålet. Om en sådan algoritm saknas säger man istället att beräkningen är *outförbar*.

Om det för varje $y \in Y$ är outförbart att hitta ett $x \in X$ sådant att $f(x) = y$, så säges f vara *urbildsresistent*. Motsvarande engelska term är *preimage resistant*. Är det outförbart att hitta x_1 och x_2 i X sådana att $f(x_1) = f(x_2)$, och $x_1 \neq x_2$ säger man att f är *kollisionsresistent*. Ibland används ”kollisionsfri” som namn på sådana funktioner, men det är missvisande, då definitionen tillåter att elementen x_1 och x_2 existerar, bara det är svårt att hitta dem. Om det för ett givet $x_1 \in X$ är outförbart att hitta $x_2 \in X$

för vilket $f(x_1) = f(x_2)$ och $x_1 \neq x_2$, så kallas f för *svagt urbildsresistent*. Motsvarande engelska termer är *weak preimage resistant* eller *second preimage resistant*.

Man kan bevisa att en kollisionsresitent funktion också är urbildsresistent. En funktion f som är kollisionsresistent kallas också för en *kryptologisk hackfunktion*. Den engelska termen är *cryptographic hash function*. Den svenska översättning antyder att hackfunktioner ”förstör” indata genom att hacka sönder det så mycket att det är svårt att hitta urbilden.[1]

Exempel 2.5. En möjlig tillämpning av hackfunktioner är vid autentisering. Antag att Bob administrerar ett datornätverk och har godkänt Alice som användare. Hon har av Bob fått en nyckel k och en hackfunktion f. Då Alice vill logga in på nätverket går de tillväga på följande sätt.

1. Bob väljer slumpmässigt ett $x \in X$ och skickar denna till Alice. Man kallar x för en *utmaning*.
2. Alice beräknar $y = f(k, x)$ och skickar y till Bob.
3. Bob beräknar $y' = f(k, x)$. Om $y' = y$, så är det troligen Alice han kommunicerar med och han släpper in henne i sitt nätverk.

Om Eve vill lura Bob genom att identifiera sig som Alice, måste hon hitta ett k' sådant att $f(k, x) = f(k', x)$, d v s en kollision. Eftersom Bob dessutom ändrar sin utmaning x vid varje autentisering ställs Eve inför ett svårt problem. ♢

2.3.1 *Merkle-Damgårds konstruktion*

Merkle-Damgårds konstruktion är en metod att skapa kollisionsresistenta hackfunktioner. Den har använts vid design av bl a MD5, SHA-1 och SHA-2. Ralph Merkle utvecklade metoden 1979. Båed Merkle och Ivan Damgård visade oberoende av varandra att konstruktionen ger en kollisionsresistent hackfunktion.

Sätt $\mathbb{B} = \{0, 1\}$. Låt $\mathbb{B}^*$ och $\mathbb{B}^n$ beteckna mängden av alla bitsträngar av godtycklig längd respektive längd n. Om $x, y \in \mathbb{B}^*$, så betecknar

$$x \,\|\, y$$

sammanfogningen (konkatenering) av bitsträngarna x och y. Vi ska i detta avsnitt studera en metod att definiera en hackfunktion på formen

$$h\colon \mathbb{B}^* \to \mathbb{B}^n,$$

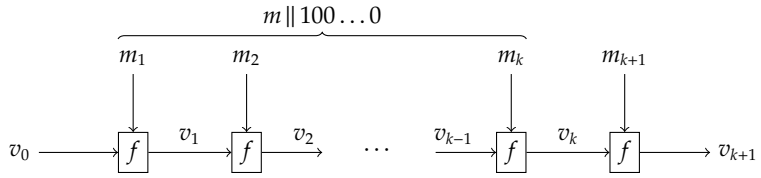
d v s en funktion som oavsett antal bitar i indata $m \in \mathbb{B}^*$ kommer utdata $h(m)$ vara en bitsträng av längd n för något fixt positivt heltal n.

Val av parametrar Välj positiva heltal n och r. Definiera en funktion $f\colon \mathbb{B}^{n+r} \to \mathbb{B}^n$. Eftersom f avbildar strängar av längd $n + r$ på strängar av längd n kallas f för en *komprimeringsfunktion med kompressionsgrad r*. Välj ett fixt $v_0 \in \mathbb{B}^n$.

Konstruktion av en hackfunktion Låt $m \in \mathbb{B}^*$ beteckna indata till h.

[1] Fyll på i slutet av m med en 1:a följt av ingen eller flera 0:or så att längden av nya bitsträngen m' är en multipel av r. Detta gör man även om längden av m redan är en multipel av r.

[1] Det engelska ordet ”hash” kan också översättas till ”pölsa”.



Figur 2.4. Merkle-Damgårds konstruktion av $h(m) = v_{k+1}$

[2] Dela upp m' i block av längd r, d v s

$$m' = m_1 \,\|\, m_2 \,\|\, \ldots \,\|\, m_k,$$

där $m_i \in \mathbb{B}^r$.

[3] Skapa ytterligare ett block $m_{k+1} \in \mathbb{B}^r$ där man lagrar längden av m givet i det binära talsystemet. Bitarna högerställs och man fyller på från vänster med tillräckligt många nollor.

[4] Bilda blocken $v_i \in \mathbb{B}^n$ rekursivt enligt

$$v_i = f(v_{i-1} \,\|\, m_i),$$

där $i = 1, 2, \ldots, k+1$.

[5] Utdata från h med m indata definieras som v_{k+1}, d v s $h(m) = v_{k+1}$.

Merkle-Damgårds konstruktion kan illustreras enligt figur 2.4.

Exempel 2.6. Låt $n = 3$ och $r = 6$. Komprimeringsfunktionen $f\colon \mathbb{B}^{3+6} \to \mathbb{B}^3$ definieras i detta exempel enligt

$$f(a) = x \oplus y \oplus z,$$

där biträngen $a = x \,\|\, y \,\|\, z$ delas upp i tre biträngar $x, y, z \in \mathbb{B}^3$ och där $x \oplus y$ betecknar bitvis addition modulo 2. Vidare sätt $v_0 = 101$.

Låt $m = 1100101110011$. Eftersom m består av 13 bitar måste vi lägga till 10000 i slutet av m för att få en bitsträng vars längd är en multipel av $r = 6$, d v s

$$m' = m \,\|\, 10000 = 110010111001110000.$$

Då är m' en bitsträng av längd 18. I nästa steg delar vi upp m' i delsträngar som var och är av längd $r = 6$. Vi får att

$$m' = m_1 \,\|\, m_2 \,\|\, m_3 = 110010 \,\|\, 111001 \,\|\, 110000.$$

Vidare är $k = 3$ och $m_{k+1} = m_4 = 001101$, eftersom $13_{\text{tio}} = 1101_{\text{två}}$. Det ger att

$$\begin{aligned}
v_1 &= f(v_0 \,\|\, m_1) = f(101110010) = 101 \oplus 110 \oplus 010 = 001 \\
v_2 &= f(v_1 \,\|\, m_2) = f(001111001) = 001 \oplus 111 \oplus 001 = 111 \\
v_3 &= f(v_2 \,\|\, m_3) = f(111110000) = 111 \oplus 110 \oplus 000 = 001 \\
v_4 &= f(v_3 \,\|\, m_4) = f(001001101) = 001 \oplus 001 \oplus 101 = 101.
\end{aligned}$$

Alltså är $h(m) = h(1100101110011) = 101$. ◊

2.4 Krypteringsoperationer för blockkrypton

Antag att en klartext är så lång att den måste delas upp i fler block. I denna framställning har varje sådant block krypterats oberoende av övriga block. Denna metod kallas för *Electronic Codebook* (ECB). Det finns flera andra krypteringsmodeller för blockkrypton och vi ska nedan studera några av dessa.

Låt m_i och c_i beteckna det i:te klartextblocket respektive motsvarande i:te kryptogramblock, där i är ett positivt heltal. Krypteringsfunktionen och dekrypteringsfunktionen i det aktuella blockkryptot betecknas E respektive D. Låt k betecknas krypteringsnyckeln och eftersom kryptot är symmetriskt så används k även för att beteckna dekrypteringsnyckeln. Då kan kryptering och dekrypteirng enligt ECB beskrivas algebraiskt som

$$c_i = E_k(m_i) \quad \text{respektive} \quad m_i = D_k(c_i).$$

Låt IV vara ett fixt block, vilket också kallas *initialvektor*. Precis som med krypteirngs- och dekrypteringsnycklarna måste Alice och Bob hålla IV hemlig. Vidare, låt ℓ vara kryptots blocklängd och r ett heltal sådant att $1 \leq r < \ell$. Funktionerna $F_n(b)$ och $L_n(b)$ retunerar de n första respektive de n sista komponenterna i blocket b.

2.4.1 *Cipher Block Chaining (CBC)*

Sätt $c_0 = \text{IV}$. För $i \geq 1$ ges det i:te kryptogramblocket av

$$c_i = E_k(m_i \oplus c_{i-1}).$$

Klartexten delas in i block av längd ℓ. Vid dekryptering återfår man klartexten enligt

$$m_i = D_k(c_i) \oplus c_{i-1},$$

eftersom

$$D_k(c_i) = D_k(E_k(m_k \oplus c_{i-1})) = m_k \oplus c_{i-1}$$

och $c_{i-1} \oplus c_{i-1} = 00\ldots0$.

2.4.2 *Cipher Feedback (CFB)*

Klartexten delas in i block m_i av längd r. Sätt $x_1 = \text{IV}$ och

$$y_i = E_k(x_i), \quad c_i = m_i \oplus F_r(y_i) \quad \text{och} \quad x_{i+1} = L_\ell(x_i \,\|\, c_i),$$

där $i \geq 1$.

2.4.3 *Output Feedback (OFB)*

Klartexten delas in i block av längd r. Sätt $x_1 = \text{IV}$ och

$$y_i = E_k(x_i), \quad c_i = m_i \oplus F_r(y_i) \quad \text{och} \quad x_{i+1} = L_\ell(x_i \,\|\, F_r(y_i)),$$

där $i \geq 1$.

2.4.4 *Counter (CTR)*

Klartexten delas in i block av längd r. Antag att blocket IV har längden $\ell - j$. Låt n vara ett heltal, vilken binärt representeras som ett vänsterställt block av längd j. Sätt

$$n \leftarrow n + 1, \quad x_i = \mathrm{IV} \,||\, n, \quad y_i = E_k(x_i) \quad \text{och} \quad c_i = m_i \oplus F_r(y_i),$$

där $i \geq 1$. Notera att Alice och Bob bör hålla n hemlig.

2.5 Övningsuppgifter

Vernamchiffer

I nedanstående uppgifter kodas bokstäver på samma sätt som i exempel 2.1.

1. Kryptera `reträtt` med Vernamkryptot och nyckeln `tchmawb`.
2. Kryptogrammet `JOÄVO` har erhållits med nyckeln `ukvrå`. Bestäm klartexten.
3. Klartexten `hjälte` krypteras till kryptogrammet `ÅDKXXJ`. Bestäm nyckeln.

I följande uppgifter kodas klartexten på samma sätt som i exempel 2.2.

4. Kryptera `Bra` med Vernamkryptot och nyckeln

   $$101001000010100001101011.$$

5. Kryptogrammet

   $$010101110011111001000110$$

   har erhållits med nyckeln

   $$000100010101101100101010.$$

   Bestäm klartexten.

6. Klartexten `Kul` krypteras till kryptogrammet

   $$101100000100011110100000.$$

   Bestäm nyckeln.

7. Vilket kryptogram får man om man använder klartexten som nyckel i ett Vernamchiffer?

Data Encryption Standard

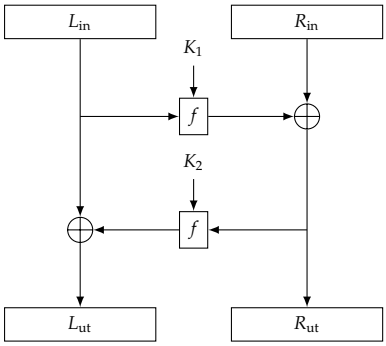
8. Låt x, y och z vara bitsträngar av samma längd. Visa följande likheter.

   (a) $x \oplus x = \mathtt{00}\ldots\mathtt{0}$ (b) $\mathtt{00}\ldots\mathtt{0} \oplus x = x$

   (c) $x \oplus y = y \oplus x$ (d) $x \oplus (y \oplus z) = (x \oplus y) \oplus z$

9. Låt x, y och z vara bitsträngar av samma längd. Förenkla

   $$x \oplus y \oplus x \oplus z \oplus y \oplus z \oplus x.$$



Figur 2.5. Variant på Feistelsystem

10. Låt a, b och c vara bitsträngar av samma längd. Visa att $a \oplus b = c$ om och endast om $a = b \oplus c$.

11. Låt $\Omega = \texttt{010110}$. Finn tre exempel på x och y sådana att $x \oplus y = \Omega$.

12. Visa att $L_i' = R_{16-i}$ och $R_i' = L_{16-i}$ för alla $i = 1, 2, \ldots, 16$, se avsnitt 2.2.3.

13. Bestäm C_1 då B_1 är bitsträngen

(a) $\texttt{100001}$ (b) $\texttt{110011}$ (c) $\texttt{001110}$ (d) $\texttt{111111}$.

14. Låt S_i beteckna den i:te S-boxen i DES. Bestäm följande.

(a) $S_3(\texttt{110011})$ (b) $S_1(\texttt{010100})$ (c) $S_7(\texttt{000010})$ (d) $S_6(\texttt{100111})$

15. Antag att $S_1(B_1) = \texttt{1011}$. Bestäm alla möjliga indata B_1 till S-box 1.

16. Vad blir resultatet för L_1R_1 efter den första rundan av Feistelsystem, om både klartextblocket och nyckeln endast utgörs av ettor?

17. Studera krypteringssystemet i figur 2.5.

(a) Uttryck krypteringen algebraiskt, d v s bestäm explicita uttryck för blocken L_{ut} och R_{ut} med avseende på L_{in} och R_{in} samt nycklarna K_1 och K_2.

(b) Härled uttryck för dekryptering och rita motsvarande diagram.

(c) Går det att byta ut funktionen f med två olika funktioner, f_1 och f_2?

(d) Låt $K_1 = \texttt{1100}$, $K_2 = \texttt{0101}$ och $f(x, y) = x \oplus y$. Kryptera den klartext som ges av blocken $L_{in} = \texttt{0011}$ och $R_{in} = \texttt{1011}$.

(e) Låt $f(x, y) = x \oplus y$. Beskriv hur man kan gå tillväga för att vid en valbar klartext-attack forcera nycklarna K_1 och K_2.

Kryptografiska hackfunktioner

18. Låt h vara hackfunktionen som definieras i exempel 2.6. Beräkna

(a) $h(0101)$ (b) $h(1111111)$ (c) $h(00110011001100)$.

19. Låt $f : \mathbb{Z}_n \to \mathbb{Z}_n$ enligt $f(x) = 12x + 1 \bmod n$. Bestäm samtliga kollisioner för f.

(a) $n = 15$ (b) $n = 18$ (c) $n = 19$

Krypteringsoperationer för blockkrypton

20. Härled formler för dekryptering för följande rypteringsoperationer.

(a) CFB (b) OFB (c) CRT

Kapitel

3 Asymmetriska kryptosystem

Grundidéen med asymmetriska kryptosystem är att även om Eve känner till krypteringsnycklen, ska hon inte på rimligt tid finna dekrytperingsnyckeln. Är detta uppfyll kan Bob göra sin krypteringsnyckel offentlig och hålla sin dekrypteringsnyckel hemlig. Hans nyckel är nu öppen för alla att använda. Alice behöver endast veta hur krypteringsnyckeln för att kunna skicka ett meddelande till Bob. Om Bob vill svara på Alice meddelande, så använder han hennes krypteringsnyckel.

3.1 Enkelriktade funktioner

Vi ger en något informell matematisk defintion av de begrepp som krävs för ett asymmetriska kryptosystem. Låt X och Y vara icke-tomma mängder. En inverterbar funktion $f\colon X \to Y$ är *enkelriktad* om det går snabbt att beräkna $f(x)$ för $x \in X$, men det tar lång tid att beräkna $f^{-1}(y)$, för $y \in Y$. Den engelska termen är *one-way function*.

Det räcker inte att en krypteringsfunktion är enkelriktad. Antag att $E\colon X \to Y$ är en enkelriktad funktion. Det andra villkoret på en enkelriktad funktion gör det svårt för Bob att dekryptera kryptogrammet, ty $D(y) = E^{-1}(y)$ är tidskrävande att beräkna. Alltså räcker det inte att E är enkelriktad. Om det går snabbt att beräkna $E(x)$ om man känner krypteringsnyckeln och om det också går snabbt att beräkna $E^{-1}(y)$, om man känner dekrypteringsnyckeln, så säges E ha en *lönndörr* (*trapddor* i engelsk litteratur). Eftersom Bob har tilgång till dekrypteringsnyckeln möjliggör detta för Bob att dekryptera meddelandet utan alltför stor ansträngning, medan forceringsarbetet för Eve inte har förenklats.

Ovanstående defintioner är vaga. Vad menar vi med att det ska ”ta lång tid att beräkna $E^{-1}(y)$”? Jo att det är fullt möjligt att beräkna $E^{-1}(y)$ även utan att känna till dekrypteringsnyckeln, men att det inte finns någon snabbare metod att göra det på än att antingen gissa eller att pröva sig fram genom att t ex testa många kombinationer av klartexter och krypteringsnycklar.

3.1.1 Privata och öppna nycklar

Bob väljer en krypteringsnyckel k, där $E = E_k$ är en enkelriktad funktion med en lönndörr. Låt d vara motsvarande dekrypteringsnyckel. För att kunna bestämma d och $D = E^{-1}$ på rimlig tid behövs någon form av extra information som inte snabbt går att härleda från k. Bob offentliggör k, men håller d hemlig och den extra information förstås. Eftersom E bestäms av k kommer även E att vara offentlig. Man säger att k är en *öppen*, *publik* eller *offentlig nyckel* och att d är en *privat nyckel*. Alice krypterar sitt meddelande x enligt $y = E(x)$ och sänder y till Bob. Med sin kunskap om d och E^{-1} kan Bob enkelt och snabbt dekryptera Alices kryptogram. Men Eve vet inte hur

man öppnar lönndörren och är då tvungen att utföra omfattande och tidskrävande beräkningar för att hitta d, E^{-1} och till slut x.

3.2 RSA

Kryptosystemet RSA namn består av initialerna i upphovsmännens efternamn, som är Ronald Rivest, Adi Shamir och Leonard Adleman. De presenterade kryptot 1978.

3.2.1 Nyckelkonstruktion

Välj två olika stora primtal p och q. Låt $n = pq$. Beräkningarna vid kryptering utförs modulo n, så klartexten delas upp i heltalsblock m, där $0 \leq m < n$. Sätt $N = (p-1)(q-1)$. Välj ett heltal e sådant att

$$0 < e < N \quad \text{och} \quad \gcd(e, N) = 1.$$

Därefter bestämmer man det heltal d sådant att $0 < d < N$ och

$$ed \equiv 1 \pmod{N}.$$

Den publika nyckel är paret (n, e) och den privata nyckel är trippeln (p, q, d).

3.2.2 Kryptering

Ett klartextblock kodas som ett heltal m, där $0 \leq m < n$. Motsvarande kryptogram c fås med krypteringsfunktionen

$$c = E(m) \equiv m^e \pmod{n},$$

där $0 \leq c < n$.

Exempel 3.1. Låt $p = 457$ och $q = 607$. Då är

$$n = pq = 457 \cdot 607 = 277\,399$$

och

$$N = (457 - 1)(607 - 1) = 276\,336.$$

Välj t ex $e = 1223$. Notera att

$$\gcd(e, N) = \gcd(1223, 276\,336) = 1.$$

Genom att lösa den linjära kongruensen

$$1223x \equiv 1 \pmod{276\,336},$$

t ex med hjälp av Euklides algoritm, finner vi att $d = x = 49\,031$. Alltså ges den publika respektive den privata nyckeln av

$$(n, e) = (277\,399, 1223) \quad \text{respektive} \quad (p, q, d) = (457, 607, 49\,031)$$

Efersom $2^{18} < n < 2^{19}$ så kan vi kryptera klartextblock m som uppfyller

$$0 \leq m < 2^{18}.$$

Om vi kodar varje tecken i en klartext enligt åtta bitars ASCII och konkatenera två bitsträngar representerar de ett positivt heltal vars binära representation består av som mest 16 bitar. Varje sådant heltal är mindre än n i detta exempel. Antag att vi vill kryptera klartexten `Top secret`. De två första bokstäverna, `T` och `o`, kodas som 84 respektive 111, vilket binärt ges av `01010100` och `01101111`. Slår vi samman dessa två bitsrängar får vi

`0101010001101111`

som decimalt motsvarar heltalet 21 615, ty

$$0 \cdot 2^{15} + 1 \cdot 2^{14} + 0 \cdot 2^{13} + 1 \cdot 2^{12} + 0 \cdot 2^{11} + 1 \cdot 2^{10} + 0 \cdot 2^9 + 0 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 1 = 21\,615.$$

Med andra ord kodas `To` som 21 615. Hela klartexten kodas till block enligt följande.

`To`	`0101010001101111`	21 615
`p␣`	`0111000000100000`	28 704
`se`	`0111001101100101`	29 541
`cr`	`0110001101110010`	25 458
`et`	`0110010101110100`	25 972

Det kryptogram som hör till första klartextblocket ges av

$$21\,615^{1223} \equiv 252\,613 \pmod{277\,399}.$$

Hela kryprogrammet blir

$$252\,613,\ 214\,301,\ 123\,634,\ 92\,904,\ 152\,075.$$

Vi skicker dessa heltal till tänkt mottagare utan ”översätta” dem till bokstäver. ◊

3.2.3 *Dekryptering*

Man dekrypterar ett kryptogram c med funktionen

$$m = D(c) \equiv c^d \pmod{n}.$$

Beviset för att dekrypteringsfunktionen verkligen fungerar, d v s att man får tillbaka den ursprungliga klartexten, faller utanför ramen för denna framställning.

Exempel 3.2. Kryptogramblocken

$$200\,228,\ 272\,958,\ 46\,956$$

är krypterade med samma publika nyckel som i föregående exempel. Dekryptering av första blocket ger

$$200\,228^{49\,031} \equiv 19\,684 \pmod{277\,399}$$

På samma sätt dekrypterar man de två andra blocken, vilket till slut ger oss

$$19\,684,\ 29\,538,\ 24\,946$$

vilket binärt motsvaras av

`0100110011100100`, `0111001101100010`, `0110000101110010`.

Kodning ger oss klartexten `Läsbar`. ◊

3.2.4 Sårbarhet

Om det slumpar sig så att $\gcd(m, n) \neq 1$, så kan man faktorisera m. Det innebär att man tar en liten risk när man väljer att använda RSA. För stora primtal p och q är den risken mycket liten, se övningsupppgift 8. Man känner till flera brister hos RSA, vilka man kan förebygga genom att t ex välja bra nycklar. Man bör bl a välja p och q så att skillnaden $p - q$ är stor och så att $p - 1$ och $q - 1$ inte ara har små primtalsdelare. Viss attacker grundar sig på tidtagning av beräkningarna under dekryptering, då det ger insyn i hur många ettor den binära representationen av d innehåller. Det kan man motverka genom att lägga in en fördröjning i funktionen.

3.3 Diffie-Hellmans nyckelutväxling

Asymmetriska kryptosystem i all ära, men tyvärr har de den gemensam nackdelen att kryptering och dekryptering tar lång tid, bl a på grund av beräkningarna involverar mycket stora heltal. Därför föredrar man snabbare symmetriska kryptosystem. Men det gör att man är tillbaka på ruta ett och återigen måste träffas för att utbyta nycklar. En lösning på problemet är att använda ett asymmetriskt krypto för att på varsitt håll bestämma sin del av en nyckel till ett symmetrisk krypto. Om Eve avlyssnar kommunikationen mellan Alice och Bob, ska hon inte kunna luska ut vilken nyckel de kommit överens om.

Algoritm 3.1. Alice och Bob vill bestämma vilken nyckel k som de ska använda till ett symmetriskt krypto. De väljer ett stort primtal p och ett heltal g sådant $\gcd(g, p) = 1$. Både p och g gör de publika, d v s även Eve får känna till p och g. Eventuellt kan de låta en betrodd tredje part välja p och g.

[1] Alice väljer ett heltal a, där $1 \leq a \leq p-2$ och beräknar $c \equiv g^a \pmod{p}$. Hon sänder c till Bob.

[2] Oberoende av Alice väljer Bob slumpmässigt ett heltal b, där $1 \leq b \leq p - 2$. Han beräknar $d \equiv g^b \pmod{p}$ och sänder d till Alice.

[3] Nyckeln k bestämmer Alice genom att beräkna $k \equiv d^a \pmod{p}$.

[4] Samma nyckel erhåller Bob genom att beräkna $k \equiv c^b \pmod{p}$.

För att heltalet k ska fungerar som en krypteringsnyckel till ett symmetriskt kryptoystem, som t ex DES, kan Alice och Bob använda den binära representationen av k. Att Alice och Bob erhåller samma nyckel k följer från

$$d^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv c^b \pmod{p}.$$

De två heltal a och b som Alice och Bob väljer håller de hemliga även för varandra. Om vi antar att Eve har avlyssnat Alice och Bob, så vet hon c och d, tillsammans med de publika parametrarna p och g. För att Eve ska kunna lista ut k utifrån det hon vet, måste hon bestämma a och b, för vilket det idag inte finns någon känd snabb metod.

Exempel 3.3. Låt p vara primtalet 127 och låt $g = 3$. Alice väljer $a = 41$ och skickar

$$c \equiv 3^{41} \equiv 78 \pmod{127}$$

till Bob. Han i sin tur väljer $b = 38$ och sänder

$$d \equiv 3^{38} \equiv 17 \pmod{127}$$

till Alice. Hon bestämmer nyckeln till

$$k \equiv d^a \equiv 17^{41} \equiv 31 \pmod{127}.$$

Samma värde på nyckeln k får Bob då han beräkna c^b modulo 127. ◊

3.4 Digitala signaturer

Hur ska Bob kunna veta att ett meddelande som han mottagit verkligen är från Alice? En lösning är att använda det som kallas *digitala signaturer*. Tekniken kan också användas för att förhindrar Alice att i efterhand försöka förneka att hon signerat ett dokument, som t ex ett kontrakt.

Algoritm 3.2 (Digitala signaturer i RSA)**.** Låt (n, e) och (p, q, d) vara Alice publika respektive privata nyckel. Vidare låt h vara en kryptografisk hackfunktion. Alice vill signera meddelandet m till Bob. För att signera meddelandet gör Alice följande.

[1] Bestämmer $x = h(m)$.

[2] Beräknar $s \equiv x^d \pmod{n}$.

[3] Skickar (m, s) till Bob, som sitt signerade meddelande.

Bob i sin tur verifiera att det är som Alice som signerat meddelandet genom att utföra nedanstående steg.

[1] Bestämmer $y = h(m)$.

[2] Beräknar $z \equiv s^e \pmod{n}$.

[3] Om $y = z$, accepterar Bob signaturen vara äkta.

Bob kan konstatera att Alice är avsändaren, eftersom endast hon vet d och kan därför utföra beräkningen x^d modulo n_1. Detta förhindrar av samma skäl Alice att senare försöka förneka att hon sänt meddelandet till Bob. Givetvis kan Alice kryptera m med Bobs publika nyckel och sedan skicka kryptogrammet istället tillsammans med s.

Exempel 3.4. Låt

$$(n, e) = (589, 73) \quad \text{och} \quad (p, q, d) = (19, 31, 37)$$

vara Alice publika respektive privata nyckel. Antag att Alice ska signera meddelandet `Avtal`. Med åttar bitars ASCII kodas hennes meddelande som

$$m = \texttt{01000001011101100111010001100001011011100}.$$

Med hackfunktionen som definieras i exempel 2.5 får Alice att

$$x = h(m) = \texttt{100}.$$

Decimalt är $x = 4$. Det ger Alice signaturen

$$s \equiv x^d \equiv 4^{37} \equiv 574 \pmod{589}.$$

Hon skickar

$$(\texttt{Avtal}, 574)$$

till Bob som sitt signerade meddelande. Bob i sin tur beräknar

$$y = h(m) = \texttt{100} = 4_{\text{tio}} \quad \text{och} \quad z \equiv s^e \equiv 574^{73} \equiv 4 \pmod{589}.$$

Han ser att $y = z$ och ser därför att han har belägg för att utgå från att meddelandet är signerat av Alice. ◊

Det finns flera fördelar med använda en kryptografisk hackfunktion vid digital signering. En hackfunktion är känslig på så sätt att en liten ändring av indata ger en stor skillnad i utdata. Det motverkar för Alice att i efterhand påstå att hon inte signerade just det meddelandet. Att signera ett långt meddelande tar tid och minne. En hackfunktion avbildar ett långt meddelande till ett litet värde, t ex en bitsträng av fix längd.

3.5 Optimal Asymmetric Encryption Padding

RSA är i grunden deterministiskt på så sätt att en klartext alltid krypteras på samma sätt. Bellare och Rogaway utvecklade 1995 metoden *Optimal Asymmetric Encryption Padding* (OAEP) med vilken man kan få till en probabilistisk kryptering med en deterministisk krypteringsfunktion. Metoden gör det också svårare att genomföra en partiell forcering av kryptogram, d v s att återskapa delar av ett klartextblock utan att bestämma den privata nyckeln.

3.5.1 Kodning av klartext

Låt $f\colon \mathbb{Z}_m \to \mathbb{Z}_m$ beteckna krypteringsfunktionen för ett deterministisk asymmetriskt kryptosystem. Vidare, låt k vara det heltal som uppfyller

$$2^k \leq m < 2^{k+1},$$

d v s talet k betecknar det maximala antal bitar som binärt ett block kan bestå av vid kryptering med f. Välj positiva heltal k_0 och k_1 sådana att

$$n = k - k_0 - k_1 > 0.$$

En klartext delas upp i block som var och en binärt består av som mest n bitar. Välj två funktioner

$$G\colon \mathbb{B}^{k_0} \to \mathbb{B}^{n+k_1} \quad \text{och} \quad H\colon \mathbb{B}^{n+k_1} \to \mathbb{B}^{k_0}.$$

Om $n + k_1 > k_0$, så kan man välja en kryptografisk hackfunktion som H.

Låt $x \in \mathbb{B}^n$ vara ett klartextblock. För att koda x till ett block $z \in \mathbb{B}^k$ går man tillväga på följande sätt. Bilda $y \in \mathbb{B}^{n+k_1}$ enligt

$$y = x \,||\, 00\ldots0,$$

d v s lägg till k_1 stycken 0:or i slutet av x. Välj $r \in \mathbb{B}^{k_0}$ slumpmässigt. Sätt därefter

$$L = y \oplus G(r) \quad \text{och} \quad R = r \oplus H(L),$$

där $a \oplus b$ betecknar bitvis addition modulo 2. Slutligen sätt

$$z = L \,||\, R.$$

Eftersom

$$n + k_1 + k_0 = (k - k_0 - k_1) + k_1 + k_0 = k$$

samt $L \in \mathbb{B}^{n+k_1}$ och $R \in \mathbb{B}^{k_0}$ så följer det att $z \in \mathbb{B}^k$.

3.5.2 Kryptering och dekryptering

Vid kryptering kodas först ett klartextblock $x \in \mathbb{B}^n$ till $z \in \mathbb{B}^k$ enligt metoden i föregående avsnitt. Bitsträngen z motsvarar binärt ett heltal $a \in \mathbb{Z}_m$. Kryptogrammet ges av $c = f(a)$.

Dekryptering ger $a = f^{-1}(c)$ och därmed fås automatiskt också bitsträngen z genom att konvertera a till det binära talsystemet. I nästa avsnitt studera vi hur man bestämmer klartexten x givet z.

3.5.3 Avkodning till klartext

Låt $z \in \mathbb{B}^k$. Bestäm L och R genom att dela upp z i två delblock så att L är de $n + k_1$ första bitarna i z och R är de resterande k_0 bitarna i z. Bestäm därefter i tur och ordning

$$s = R \oplus H(L) \quad \text{och} \quad t = L \oplus G(s).$$

Då är x de n första bitarna i t eftersom

$$s = R \oplus H(L) = (r \oplus H(L)) \oplus H(L) = r \oplus 00\ldots0 = r$$

och

$$t = L \oplus G(s) = y \oplus G(r) \oplus G(r) = y \oplus 00\ldots0 = y$$

samt $y = x \,||\, 00\ldots0$. I princip är OAEP ett Feistelsystem.

Exempel 3.5. Låt $m = 25\,468\,447$. Då är $2^{24} \leq m < 2^{25}$, d v s $k = 24$. Med valen $k_0 = 8$ och $k_1 = 4$ är $n = k - k_0 - k_1 = 12$. Med andra ord kan vi kryptera klartextblock av längd 12. Vidare definierar vi funktionerna

$$G\colon \mathbb{B}^8 \to \mathbb{B}^{12+4} \quad \text{och} \quad H\colon \mathbb{B}^{12+4} \to \mathbb{B}^8$$

enligt

$$G(a) = \mathrm{ls}_3(a) \,||\, \mathrm{ls}_5(a) \quad \text{och} \quad H(a) = b \oplus c,$$

där $\mathrm{ls}_n(a)$ betecknar cykliskt vänsterskift n steg av a och där b och c är vänster- respektive högerblock av a, d v s $a = b \,||\, c$ och $b, c \in \mathbb{B}^8$.

Låt $x = 995$ vara ett klartextblock. Binärt med $n = 12$ bitar är $x = 001111100011$. Eftersom $k_1 = 4$ ska vi lägga till fyra 0:or i slutet av x när vi bestämmer y, d v s

$$y = x \,||\, 0000 = 0011111000110000$$

Låt $r = 10111001$. Då är

$$\begin{aligned} G(r) &= G(10111001) \\ &= \mathrm{ls}_3(10111001) \,||\, \mathrm{ls}_5(10111001) \\ &= 11001101 \,||\, 00110111 \\ &= 1100110100110111 \end{aligned}$$

och

$$\begin{aligned} L &= y \oplus G(r) \\ &= 0011111000110000 \oplus 1100110100110111 \\ &= 1111001100000111. \end{aligned}$$

Det ger att

$$\begin{aligned} H(L) &= H(1111001100000111) \\ &= 11110011 \oplus 00000111 \\ &= 11110100 \end{aligned}$$

och därmed är

$$\begin{aligned} R &= r \oplus H(L) \\ &= 10111001 \oplus 11110100 \\ &= 01001101. \end{aligned}$$

Alltså är

$$z = L \,||\, R = 1111001100000111 \,||\, 01001101 = 111100110000011101001101,$$

vilket är en bitsträng av längd 24 som decimalt motsvarar heltalet 15 927 117. Detta heltal kan nu krypteras t ex med RSA och $m = 25\,468\,447$ som modulus. Att avkoda z till klartexten x lämnas som övning. ♢

3.6 ElGamals kryptosystem baserat på elliptiska kurvor

I detta avsnitt ska vi studera ett asymmetriskt kryptosystem i vilket man utför beräkningarna vid kryptering och dekryptering över en elliptisk kurva. Det är ett probabilistiskt kryptosystem som påminner om ElGamals kryptosystem baserat på primitiva rötter. Kryptot utvecklades omkring 1986 av Victor Miller och av Neal Koblitz, oberoende av varandra.

3.6.1 *En mycket kort introduktion till elliptiska kurvor*

I denna korta sammanfattning av teorin för elliptiska kurvor begränsar vi oss till det som man behöver för att kunna använda elliptiska kurvor i kryptologi.

Låt p vara ett primtal som är större än 3. Vidare, låt A och B vara element i $\mathbb{Z}_p$ sådana att $4A^3 + 27B^2 \not\equiv 0 \pmod{p}$. Då definieras en *elliptisk kurva över* $\mathbb{Z}_p$ som mängden

$$E = \{(x, y) \in \mathbb{Z}_p^2 : y^2 \equiv x^3 + Ax + B \pmod{p}\} \cup \{\mathcal{O}\}, \tag{3.1}$$

där $\mathcal{O}$ är en s k *oändlighetspunkt*. Vi ska längre fram definiera en räkneoperation på mängden E och för att få den att fungera behövs oändlighetspunkten. Denna punkt saknar koordinater på formen (x, y).

Exempel 3.6. Låt $p = 23$ och antag att E ges av

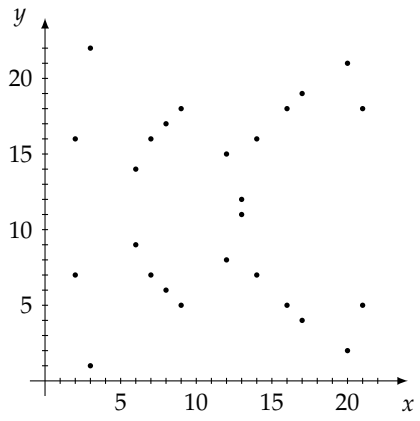
$$y^2 \equiv x^3 + 2x + 14 \pmod{23}, \tag{3.2}$$

d v s $A = 2$ och $B = 14$. Vi noterar att

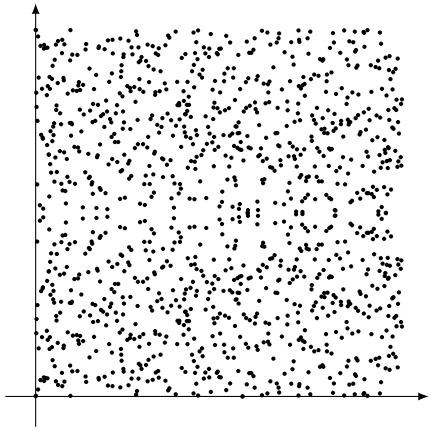
$$4A^3 + 27B^2 \equiv 11 \not\equiv 0 \pmod{23}.$$

Om $x = 0$, så är (3.2) ekvivalent med

$$y^2 \equiv 0^3 + 2 \cdot 0 + 14 \equiv 14 \pmod{23},$$



Figur 3.1. $y^2 \equiv x^3+2x+14 \pmod{23}$



Figur 3.2. $y^2 \equiv x^3+2x+1 \pmod{1213}$

som dock saknar lösning. Alltså innehåller E inte någon punkt vars x-koordinat är lika med 0, d v s någon punkt på formen $(0, y)$. Om istället $x = 2$ får vi kongruensen

$$y^2 \equiv 2^3 + 2 \cdot 2 + 14 \equiv 3 \pmod{23},$$

som har lösningarna

$$y = 7 \quad \text{och} \quad y = 16.$$

Det betyder att punkterna $(2,7) \in E$ och $(2,16) \in E$. På samma sätt studerar vi kongruensen (3.2) för samtliga 23 möjliga värden på $x \in \mathbb{Z}_{23}$. Det ger till slut att

$$\begin{aligned} E = \big\{&\mathcal{O}, (2,7), (2,16), (3,1), (3,22), (6,9), (6,14), (7,7), (7,16), (8,6), (8,17), \\ &(9,5), (9,18), (12,8), (12,15), (13,11), (13,12), (14,7), (14,16), (16,5), \\ &(16,18), (17,4), (17,19), (20,2), (20,21), (21,5), (21,18)\big\}. \end{aligned}$$

Antal element i E är 27. Det kan jämföras med det totala antalet punkter som den kartetiska produkten $\mathbb{Z}_{23}^2 = \mathbb{Z}_{23} \times \mathbb{Z}_{23}$ innehåller, nämligen $23^2 = 529$. I figur 3.1 är alla punkter i E, utom oändlighetspunkten $\mathcal{O}$, utritade. ◊

Låt $P \in E$ och antag att $P \neq \mathcal{O}$, d v s $P = (x, y)$ för några $x, y \in \mathbb{Z}_p$. Då definieras *speglingen av* P som punkten

$$P' = (x, p - y).$$

Notera att $p - y \equiv -y \pmod{p}$. Sätt $\mathcal{O}' = \mathcal{O}$. Härnäst ska vi definiera en addition på E. Till att börja med måste additionen fungera så att oavsett vilka två punkter P och Q på kurvan E som vi addera ska summan $P + Q$ också vara en punkt på E. Additionen kommer att har följande egenskaper.

(a) $(P + Q) + R = P + (Q + R)$ för alla $P, Q, R \in E$.

(b) $P + Q = Q + P$ för alla $P, Q \in E$.

(c) $P + \mathcal{O} = \mathcal{O} + P = P$ för alla $P \in E$.

(d) $P + P' = \mathcal{O}$ för alla $P \in E$.

Enligt (c) kan vi tolka oändligthetspunkten som ”nollan” med avseende på additionen. Med den tolkningen ger (d) i sin tur oss att speglingen P' av en punkt P är den additiva inversen till punkten P. Istället för att skriva P' och $P + Q'$ inför vi beteckninen

$$-P \quad \text{respektive} \quad P - Q.$$

Definitionen av den sökta additionen beskrivs i algoritm 3.1.

Algoritm 3.1: Addition i en elliptisk kurva

Indata: En elliptisk kurva E enligt (3.1) och två punkter P och Q på E.
Utdata: Summan $S = P + Q$.

operation $P + Q$
  om $P = \mathcal{O}$ **då**
    $S \leftarrow Q$
  annars om $Q = \mathcal{O}$ **då**
    $S \leftarrow P$
  annars om $P = -Q$ **då** ▷ Från och med nu är $P = (x_1, y_1)$ och $Q = (x_2, y_2)$.
    $S \leftarrow \mathcal{O}$ ▷ Villkoret $P = -Q$ är detsamma som $x_1 = x_2$ och $y_1 = p - y_2$.
  annars
    om $P = Q$ **då**
      $\lambda \leftarrow (3x_1^2 + A)(2y_1)^{-1} \bmod p$
    annars
      $\lambda \leftarrow (y_2 - y_1)(x_2 - x_1)^{-1} \bmod p$
    $x_3 \leftarrow \lambda^2 - x_1 - x_2 \bmod p$
    $y_3 \leftarrow \lambda(x_1 - x_3) - y_1 \bmod p$
    $S \leftarrow (x_3, y_3)$
  returnera S

Exempel 3.7. Låt E vara den elliptiska kurva som vi studerade i exempel 3.6. Vi fann då att punkterna $P = (9, 5)$ och $Q = (17, 19)$ tillhör E. Då är

$$P' = -P = (9, 23 - 5) = (9, 18)$$

och

$$P - P = P + P' = \mathcal{O},$$

d v s

$$(9, 5) + (9, 18) = \mathcal{O}.$$

För att beräkna $P + Q = (9, 5) + (17, 19)$ bestämmer vi först

$$\lambda \equiv (19 - 5)(17 - 9)^{-1} \equiv 14 \cdot 8^{-1} \equiv 19 \pmod{23}.$$

Det ger att

$$x_3 \equiv 19^2 - 9 - 17 \equiv 13 \pmod{23}$$

och

$$y_3 \equiv 19 \cdot (9 - 13) - 5 \equiv 11 \pmod{23}.$$

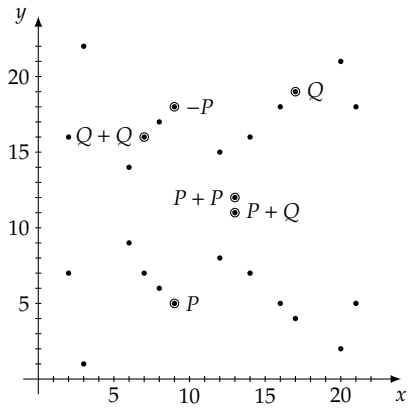
Alltså är

$$P + Q = (9, 5) + (17, 19) = (13, 11).$$

På liknande sätt finner vi att

$$P + P = (9, 5) + (9, 5) = (13, 12) \quad \text{och} \quad Q + Q = (17, 19) + (17, 19) = (7, 16).$$

Notera att i samtliga fall blir resultatet en punkt som tillhör E, se figur 3.3. ◊



Figur 3.3. Illustration av exempel 3.7.

Låt E vara en elliptisk kurva, P en punkt på E och n ett positivt heltal. Då definieras *potensen* nP som summan

$$nP = \underbrace{P + P + \cdots + P}_{n \text{ stycken}}.$$

Vidare sätt

$$0P = \mathcal{O} \quad \text{och} \quad (-n)P = \underbrace{(-P) + (-P) + \cdots + (-P)}_{n \text{ stycken}}.$$

Vi inser att

$$1P = P \quad \text{och} \quad (-1)P = -P.$$

Låt $m, n \in \mathbb{Z}$ och $P, Q \in E$. Då gäller följande.

(a) $\mathcal{O} = -\mathcal{O}$

(b) $n\mathcal{O} = \mathcal{O}$

(c) $m(nP) = (mn)P = n(mP)$

(d) $(m + n)P = mP + nP$

(e) $n(P + Q) = nP + nQ$

Det minsta positiva heltal s sådant att

$$sP = \mathcal{O}$$

kallas för *ordningen av* P. Låt ordningen av $\mathcal{O}$ vara 1. Man kan visa att s alltid delar $|E|$. Antag att P och Q är givna punkter på E. Det minsta positiva heltal n som är en lösning till ekvationen

$$nP = Q$$

kallas för den *elliptisk-diskreta logaritmen av* Q *i basen* P och betecknas

$$n = \text{edlog}_P(Q).$$

Om $nP = Q$ saknar lösning, så är $\text{edlog}_P(Q)$ inte definierad. Det har visat sig vara tidskrävande att hitta en acceptabelt snabb algoritm för att beräkna $\text{edlog}_P(Q)$. Däremot finns det snabba algoritmer för att beräkna potensen nP, se t ex algoritm 3.2.

Algoritm 3.2: Dubblera och addera

Indata: En elliptisk kurva E över $\mathbb{Z}_p$, en punkt P på E och ett heltal n som ges binärt av $n = 2^r n_r + \cdots + 2^2 n_2 + 2n_1 + n_0$, där $n_r = 1$ och $n_i \in \{0, 1\}$ då $i = 0, 1, \ldots, r-1$.
Utdata: Potensen $Q = nP$.

operation nP
  om $n_0 = 1$ **då** ▷ För $i = 0, 1, \ldots, r$ gör $n_i \leftarrow n \bmod 2$ och $n \leftarrow \lfloor n/2 \rfloor$.
    $Q \leftarrow P$
  annars
    $Q \leftarrow \mathcal{O}$
  $R \leftarrow P$
  för $i = 1, 2, \ldots, r$ **gör**
    $R \leftarrow 2R$ ▷ Notera att $2R = R + R$, d v s addition enligt algoritm 3.1.
    om $n_i = 1$ **då**
      $Q \leftarrow Q + R$
  returnera Q

3.6.2 *Nyckelkonstruktion*

Välj först ett primtal p, en elliptisk kurva E över $\mathbb{Z}_p$, en punkt $P \in E$ och ett heltal n. Bestäm därefter den punkt $Q \in E$ sådan att

$$Q = nP.$$

Privat nyckel är n och publik nyckel är (p, E, P, Q).

3.6.3 *Kryptering*

Låt klartexten representeras som en punkt M på kurvan. Det är dock inte självklart hur man koda en klartext till en punkt, se avsnitt **??** för en möjlig lösning. Välj ett slumpheltal k. och bestäm sedan

$$C = kP \quad \text{och} \quad D = M + kQ.$$

Kryptogrammet är då paret (C, D). Notera att C och P är punkter på E.

3.6.4 *Dekryptering*

För att bestämma klartexten M utför man beräkningen

$$D - nC.$$

Att det ger det önskade resultatet följer av

$$D - nC = (M + kQ) - n(kP) = M + k(nP) - k(nP) = M + \mathcal{O} = M.$$

Likheterna med ElGamals kryptosystem baserat på primitiva rötter är slående.

3.6.5 *Koblitz metod att koda klartext till punkt*

Att koda en klartext till ett heltal m är enkelt. Försöker man använda m som x- eller y-koordinat i en punkt på aktuell elliptisk kurva, så finns det risk att vi misslyckas. Det är nämligen inte garanterat att det finns någon punkt i E på formen (x, m) eller (m, y).

Låt E vara en elliptisk kurva över $\mathbb{Z}_p$ med koefficienterna A och B. Välj två positiva heltal $\mathcal{M}$ och κ sådana att $\mathcal{M}\kappa < p$. De två talen är publika parametrar – man behöver dem vid kryptering. Talet $\mathcal{M}$ ska väljas så att samtliga klartexter kodade som ett heltal m uppfyller $0 \leq m < \mathcal{M}$. Med andra ord är inte alla element i $\mathbb{Z}_p$ möjliga klartexter. Talet κ ska i sin tur väljas tillräckligt stor så att proceduren nedan har en chans att lyckas. För varje $j = 1, 2, \ldots, \kappa$ sätt

$$x = m\kappa + j$$

och testa om det existerar ett $y \in \mathbb{Z}_p$ sådant om

$$y^2 \equiv x^3 + Ax + B \pmod{p},$$

d v s om punkten (x, y) tillhör E. Om så är fallet avbryter vi och låter vi $M = (x, y)$ representera klartexten m. Tanken är med andra ord att låta x-koordinaten i en punkt innehålla m. Efter dekryptering avkodar man meddelandet enligt

$$\left\lfloor \frac{x-1}{\kappa} \right\rfloor.$$

Notera att κ är den grekiska bokstaven kappa.

3.6.6 *Hur säkert är kryptosystemet?*

Hur stora primtal måste man välja? National Security Agency (NSA) har uppskattat att om primtalet p består av 512 bitar då den representeras binärt, motsvarar det samma säkerhet som AES-256, d v s AES med nyckellängden 256 bitar. Ska man uppnå samma säkerhet med RSA krävs 15 360 bitar.

3.7 Övningsuppgifter

RSA

1. Bestäm e och n i den publika nyckeln till RSA, om den privata nyckeln är

(a) $p = 5, q = 3, d = 3$ (b) $p = 13, q = 17, d = 35$

(c) $p = 23, q = 11, d = 19$ (d) $p = 31, q = 41, d = 403$.

2. Bestäm p, q och d i den privata nyckeln till RSA, om den publika nyckeln är

(a) $n = 55, e = 3$ (b) $n = 143, e = 13$

(c) $n = 26, e = 7$ (d) $n = 95, e = 65$.

3. Låt $p = 13$ och $q = 41$. Konstruera nycklarna till ett RSA-krypto med utgångspunkt från valen av primtalen p och q. Som e ska du välja det näst minsta positiva heltal som uppfyller kraven på en korrekt nyckel till RSA. Kryptera därefter $m = 111$.

4. I denna uppgift krypterar vi en bokstav i taget. Bokstäverna kodas enligt åtta bitars ASCII, se avsnitt A.1. Låt p och q vara det 11:e respektive 17:e primtalet och låt $e = 701$.

(a) Bestäm den publika nyckeln. (b) Bestäm den privata nyckeln.

(c) Kryptera B. (d) Dekryptera $c = 1592$.

5. Varför är det inte möjligt att välja $e = 2$ i den publika nyckeln (n, e) till RSA?

6. Låt (n, e) vara Alices publika nyckel för RSA. Antag att Alice skickar till dig kryptogrammet y, vars motsvarande klartext x är okänd för dig. Alice har gått med på att dekryptera ett block $z \neq y$ som du sänder till henne, d v s hon krypterar z med sin privata nyckel d samt skickar tillbaka resultatet. Hur ska du välja z så att du kan bestämma x?

7. Höjer man säkerheten i RSA, om man krypterar en klartext i två omgångar, först med hjälp av nyckeln (n, e_1), därefter med (n, e_2)?

8. Låt p och q vara primtal. Sätt $n = pq$.

(a) Låt $m \in \mathbb{Z}$. Om $\gcd(m, n) \neq 1$, vad kan $\gcd(m, n)$ vara lika med?

(b) Hur många heltal i $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ är delbara med p?

(c) Hur många heltal i $\mathbb{Z}_n$ är delbara med minst en av p och q?

(d) Låt $m \in \mathbb{Z}_n$. Hur stor är sannolikheten att $\gcd(m, n) \neq 1$?

Digitala signaturer

9. Med samma förutsättningar som i exempel 3.4.

(a) Signera meddelandet `Pakt`.

(b) Vilket av meddelandena $(\texttt{ett}, 574)$ och $(\texttt{två}, 121)$ har Alice **inte** signerat?

ElGamals kryptosystem baserat på elliptiska kurvor

10. Låt E vara den elliptiska kurva som ges av ekvationen

$$y^2 \equiv x^3 + 4x + 16 \pmod{37}.$$

Visa att $P = (6, 21)$ och $Q = (33, 11)$ är punkter på kurvan.

11. Låt $p = 17$ och E en elliptisk kurva given av

$$y^2 \equiv x^3 + 2x + 15 \pmod{17}.$$

Låt $P = (1, 1)$ och $Q = (12, 4)$. Beräkna följande summor.

(a) $2P$ (b) $2Q$ (c) $P + Q$

(d) $P - Q$ (e) $Q - P$ (f) $2P - 2Q$

12. Låt

$$P = (16, 27) \quad \text{och} \quad Q = (38, 49)$$

vara punkter på en elliptisk kurva

$$y^2 \equiv x^3 + Ax + B \pmod{53}.$$

(a) Bestäm A och B.

(b) Lös ekvationen $P + R = \mathcal{O}$.

(c) Beräkna $P + Q$.

Kapitel

A

Tabeller

A.1 ASCII

Ett urval ur åtta bitars ASCII (respektive teckens kodning decimalt, binärt och hexadecimalt). Mellanslag betecknas ␣.

␣	32	00100000	20	B	66	01000010	42	d	100	01100100	64
!	33	00100001	21	C	67	01000011	43	e	101	01100101	65
"	34	00100010	22	D	68	01000100	44	f	102	01100110	66
#	35	00100011	23	E	69	01000101	45	g	103	01100111	67
$	36	00100100	24	F	70	01000110	46	h	104	01101000	68
%	37	00100101	25	G	71	01000111	47	i	105	01101001	69
&	38	00100110	26	H	72	01001000	48	j	106	01101010	6A
’	39	00100111	27	I	73	01001001	49	k	107	01101011	6B
(	40	00101000	28	J	74	01001010	4A	l	108	01101100	6C
)	41	00101001	29	K	75	01001011	4B	m	109	01101101	6D
*	42	00101010	2A	L	76	01001100	4C	n	110	01101110	6E
+	43	00101011	2B	M	77	01001101	4D	o	111	01101111	6F
,	44	00101100	2C	N	78	01001110	4E	p	112	01110000	70
-	45	00101101	2D	O	79	01001111	4F	q	113	01110001	71
.	46	00101110	2E	P	80	01010000	50	r	114	01110010	72
/	47	00101111	2F	Q	81	01010001	51	s	115	01110011	73
0	48	00110000	30	R	82	01010010	52	t	116	01110100	74
1	49	00110001	31	S	83	01010011	53	u	117	01110101	75
2	50	00110010	32	T	84	01010100	54	v	118	01110110	76
3	51	00110011	33	U	85	01010101	55	w	119	01110111	77
4	52	00110100	34	V	86	01010110	56	x	120	01111000	78
5	53	00110101	35	W	87	01010111	57	y	121	01111001	79
6	54	00110110	36	X	88	01011000	58	z	122	01111010	7A
7	55	00110111	37	Y	89	01011001	59	{	123	01111011	7B
8	56	00111000	38	Z	90	01011010	5A	|	124	01111100	7C
9	57	00111001	39	[	91	01011011	5B	}	125	01111101	7D
:	58	00111010	3A	\	92	01011100	5C	~	126	01111110	7E
;	59	00111011	3B	]	93	01011101	5D	Ä	196	11000100	C4
<	60	00111100	3C	^	94	01011110	5E	Å	197	11000101	C5
=	61	00111101	3D	_	95	01011111	5F	Ö	214	11010110	D6
>	62	00111110	3E	‘	96	01100000	60	ä	228	11100100	E4
?	63	00111111	3F	a	97	01100001	61	å	229	11100101	E5
@	64	01000000	40	b	98	01100010	62	ö	246	11110110	F6
A	65	01000001	41	c	99	01100011	63				

Facit

2 Symmetriska kryptosystem

1. `HGÅAÄMU`

2. `seger`

3. `txmmef`

4. 111001100101101000001010

5. Fel

6. 111110110011001011001100

7. 000 . . . 0

9. x

13. (a) `1111` (b) `1011` (c) `1000` (d) `1101`

14. (a) `1111` (b) `0110` (c) `1011` (d) `1100`

15. `001100`, `010111`, `101110` eller `110011`

16. $L_1 = 111\ldots1$ (32 bitar), R_1 = `00100111001001110010010001000011`

17. (a) $L_{\text{ut}} = L_{\text{in}} \oplus f(R_{\text{in}} \oplus f(L_{\text{in}}, K_1), K_2)$ och $R_{\text{ut}} = R_{\text{in}} \oplus f(L_{\text{in}}, K_1)$

(b) $L_{\text{in}} = L_{\text{ut}} \oplus f(R_{\text{ut}}, K_2)$ och $R_{\text{in}} = R_{\text{ut}} \oplus f(L_{\text{ut}} \oplus f(R_{\text{ut}}, K_2), K_1)$

(c) Ja

18. (a) 101 (b) 100 (c) 011

19. (a) $f(0) = f(5) = f(10) = 1$
$f(1) = f(6) = f(11) = 13$
$f(2) = f(7) = f(12) = 10$
$f(3) = f(8) = f(13) = 7$
$f(4) = f(9) = f(14) = 4$

(b) $f(0) = f(3) = f(6) = f(9) = f(12) = f(15) = 1$
$f(1) = f(4) = f(7) = f(10) = f(13) = f(16) = 13$
$f(2) = f(5) = f(8) = f(11) = f(14) = f(17) = 7$

(c) Kollision saknas.

20. (a) $y_i = E_k(x_i),\ m_i = c_i \oplus F_r(y_i)$ och $x_{i+1} = L_\ell(x_i \,\|\, c_i)$
(b) $y_i = E_k(x_i),\ c_i = m_i \oplus F_r(y_i)$ och $x_{i+1} = L_\ell(x_i \,\|\, F_r(y_i))$
(c) $n \leftarrow n + 1,\ x_i = \mathrm{IV} \,\|\, n,\ y_i = E_k(x_i)$ och $m_i \leftarrow c_i \oplus F_r(y_i)$

3 Asymmetriska kryptosystem

1. (a) $n = 15, e = 3$ (b) $n = 221, e = 11$
(c) $n = 253, e = 139$ (d) $n = 1271, e = 667$

2. (a) $p = 5, q = 11, d = 27$ (b) $p = 11, q = 13, d = 37$
(c) $p = 2, q = 13, d = 7$ (d) $p = 5, q = 19, d = 41$

3. Publik och privat nyckel är $(n, e) = (533, 11)$ respektive $(p, q, d) = (13, 41, 131)$. Kryptogrammet är $c = 15$.

4. (a) $(1829, 701)$ (b) $(31, 59, 1601)$
(c) 1585 (d) `u`

6. Välj $z = y^{e+1} \bmod p$. Multiplicera sedan det heltal Alice skickar tillbaka med y^{-1} modulo n.

7. Nej

8. (a) p eller q
(b) q, nämligen $0, p, 2p, 3p, \ldots, (q-1)q$
(c) $n - N = pq - (p-1)(q-1) = p + q - 1$
(d) $\dfrac{n-N}{n} = \dfrac{1}{p} + \dfrac{1}{q} - \dfrac{1}{pq}$

9. (a) $(\texttt{Pakt}, 5)$ (b) `två`

11. (a) $(0, 10)$ (b) $(11, 12)$ (c) $(0, 7)$
(d) $(8, 13)$ (e) $(8, 4)$ (f) $(10, 10)$

12. (a) $A = 47$ och $B = 15$ (b) $(16, 26)$ (c) $(0, 42)$