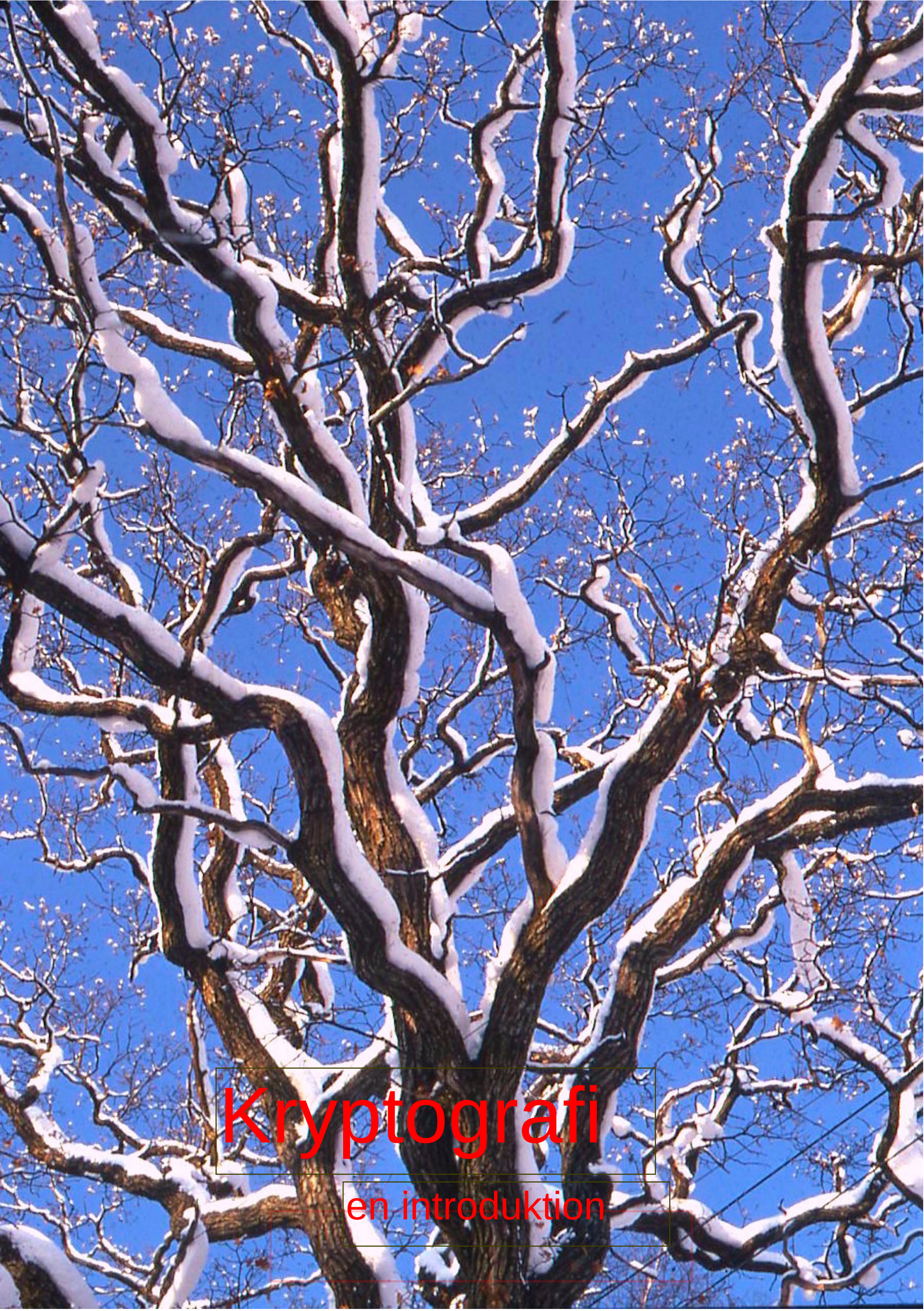


Kryptografi

en introduktion

Kryptografi

en introduktion

ISBN 978-91-983027-3-8

Omslagsbild från trakterna omkring Stora Mellansjö och

Ågestasjön, vintern 1969

Originalbild tagen med Kodachrome (diapositiv)

15 DIN i halvformat (18x24 mm)

Kamera Olympus Pen FT, objektiv Zuiko 150 mm med UV-filter

Boken har förut utgivits som pappersbok av
Studentlitteratur med ISBN 978-91-44-07238-8

Denna utgåva har samma innehåll som Studentlitteraturs

Innehåll

Förord

Syftet med denna bok är att vara en introduktion till de grundläggande principerna för kryptografi. Den är primärt avsedd att vara en kursbok alternativt fördjupningsbok för dem som studerar datasäkerhet och/eller datakommunikation på högskolenivå.

Innehållet förutsätter grundläggande kunskaper i sannolikhetslära, speciellt stokastisk variabel. Andra bakgrundskunskaper ges i boken, speciellt i kapitel 3 om Algebra för kryptografi.

Innehållet i korta drag är:

Kapitel 1 Introduktion till kryptografi

beskriver krypteringens plats i ett digitalt punkt till punkt kommunikationssystem och vilka uppgifter den avses lösa.

Kapitel 2 Kryptografi

2.1 – 2.3

Börjar med att gå igenom grundläggande begrepp. Därefter följer en beskrivning av en del historiskt intressanta chiffer.

2.4 – 2.6

Vidare följer begreppet perfekt sekretess och the one time pad. Krav på ett modernt krypteringssystem.

2.8

Begreppen entropi och betingad entropi, språktakt och redundans, entydighetsdistans och ideal sekretess.

2.9 – 2.10

Praktisk sekretess med block- och strömchiffer. Principer för de viktigaste blockchiffren DES och AES.

2.11

Hashfunktioner eller digitala fingeravtryck för skydd av bl.a. lösenord.

2.12

Publika nyckelkryptosystem. Här ingår ett appendix för bevis av RSA-algoritmen. Full förståelse av detta bevis är inte nödvändig för att förstå huvuddragen i RSA-algoritmen.

2.12 – 2.13

Elgamals kryptosystem, diskret logaritm och kryptografi med elliptiska kurvor. För att helt kunna tillgodogöra sig innehållet i detta avsnitt krävs kunskap om grupper och talkroppar som förmedlas i kapitel 3.

2.15

Dataintegritet och autenticitet. I detta avsnitt beskrivs protokollen PGP och Kerberos för att illustrera hur olika algoritmer används som byggstenar för att åstadkomma önskade säkerhetslösningar.

2.16

Kvantkryptografi. Detta något futuristiska avsnitt beskriver kortfattat de kvantmekaniska effekter på vilka eventuella framtida krypteringsmetoder kan komma att baseras på.

2.17

I detta Appendix finns en mer detaljerad beskrivning av de olika stegen i de två blockchiffren DES och AES.

3 Algebra för kryptografi

Här ges grundläggande kunskaper inom det som också kallas abstrakt algebra, begränsat till framförallt grupper och talkroppar. Avsnittet om ringar är inte nödvändigt för att kunna följa framställningen i avsnitten 2.13 och 2.14 om Elgamals kryptosystem, diskret logaritm och kryptografi med elliptiska kurvor.

För många värdefulla påpekanden och synpunkter tackar jag universitetsadjunkt Göran Andersson, KTH Kista och fil kand Pär Höglund Uppsala.

Knivsta i november 2010

Christer Frank

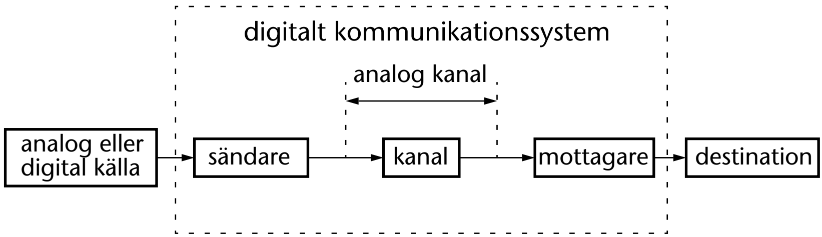
1 Introduktion till kryptografi

1.1 Krypteringens plats i ett kommunikationssystem

I ämnet digital kommunikation studeras hur information överförs via en störd och/eller avlyssnad och/eller manipulerad kanal.

I principschemat för ett digitalt kommunikationssystem enligt Figur 1.1 visas hur information, som genereras av en analog eller digital källa, överförs till en destination eller slutanvändare via en kanal.

Det digitala kommunikationssystemet består av en sändare som anpassar meddelandet, gör det mer robust, för överföring via kanalen. I kanalen påverkas nämligen de överförda signalerna av brus, störningar, variationer i kanalens överföringsegenskaper och andra signalförsämrande effekter av mer eller mindre allvarlig art. En viktig anpassning av det digitala meddelandet till kanalen är att omvandla det till analog form eftersom de flesta fysiska kanaler till sin natur är analoga och endast kan överföra vågformsignaler. Den omvandling som görs i sändaren i detta syfte kallas digital modulation och är den viktigaste operation signalen undergår.



Figur 1.1 Principschema för ett digitalt kommunikationssystem.

I mottagaren, efter vågformsignalens passage av kanalen, utförs den omvända operationen, demodulation eller detektering.

För det fall att källan genererar en analog signal omvandlas den i sändaren till digital form av en analogdigitalomvandlare. På mottagarutgången digitalanalogomvandlas därför den mottagna och demodulerade signalen innan den når destinationen.

Fördelen med digital kommunikation jämfört med analog är bland annat att:

- Alla meddelandesignaler omvandlas till en enhetlig digital form och att vågformsignalerna oberoende av källsignalens natur kan optimeras med avseende på att undertryck inverkan av olika former av försämrande påverkningar som signalen utsätts för i överföringskanalen.
- Oberoende av meddelandekällans natur som tal, data, video, m.fl., kan flera meddelandesignaler multiplexas och sammanföras till en gemensam ”bitström”. Detta är vid analog kommunikation endast delvis möjligt.
- Vid långa överföringssträckor, då signalnivån sjunkit, kan den återställas till sin ursprungliga form i en repeater. Vid motsvarande operation utförd med analog kommunikation minskar signalens signalbrusförhållande i varje repeater (överdrag), något som sätter en övre gräns för antalet repeatrar och därmed den totala överföringssträckans längd.
- Bandbredden med hjälp av datakomprimering/källkodning och flernivåsignalering kan göras mindre än för motsvarande meddelandeöverföring utförd med analog kommunikation.

1.2 Digital punkt till punkt kommunikation

Principschemat för ett digitalt kommunikationssystem enligt Figur 1.1 är mycket generellt. Det är tillämpligt för såväl minneslagringssystem som trådlösa eller trådbundna informationsöverföringssystem. För överföring av information mellan två punkter är det för en konstruktör av ett digitalt kommunikationssystem viktigt att förstå de olika blockens inverkan och betydelse för kommunikationsprocessen. Det är endast blocken ”sändare” och ”mottagare” som kan påverkas direkt av konstruktören. Men för att denna ”påverkan” skall vara meningsfull är det viktigt att den görs med kännedom om och anpassning till de andra blocken: Analog eller digital källa, kanal och destination.

Blocket sändare kan delas upp i ett antal funktionsblock svarande mot olika operationer som meddelandesignalen undergår från källa till kanal, se Figur 1.2 Källan kan vara analog eller digital. Analoga källsignaler analogdigitalomvandlas innan de ansluts till den egentliga sändaren.

I en sändare kan det finnas en källkodare som tar bort redundansen d.v.s. eliminerar överflödiga databitar. Operationen kallas också datakomprimering och avsikten är bl.a. att reducera meddelandets överföringstid på kanalen genom att ta bort de databitar i meddelandet som inte innehåller någon information.

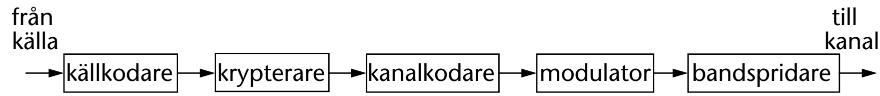
I kanalen utsätts meddelandesignalen för allehanda störningar men innehållet i meddelandet kan också utsättas för oönskad avlyssning. Om förbindelsen är av den arten att avlyssning inte får ske innehåller sändaren en krypterare.

Innan dataströmmen når modulatorn passerar den en kanalkodare som har till uppgift att förse databitarna med checkbitar, som gör det möjligt att i mottagaren korrigera eventuella bitfel som kan uppstå i kanalen. Exempelvis kan p.g.a. störningar i kanalen en sänd nolla detekteras som en etta i mottagaren. Kanalkodare ingår numera i de flesta digitala sändare.

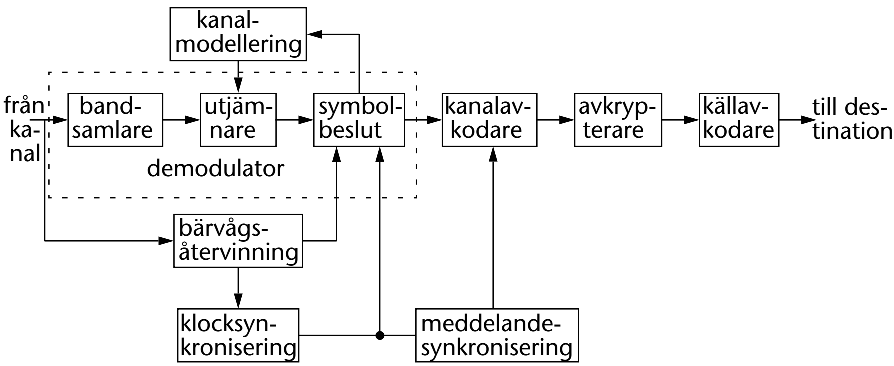
I blocket modulator omvandlas den digitala dataströmmen till analoga vågformer. Vilken typ av vågform som väljs beror på kanalens egenskaper.

I nära anknytning till modulatorn, ibland som en naturlig del av den, kan en bandspridare finnas som har till uppgift att expandera signalens bandbredd. Historiskt sett användes bandspridning först i militära sammanhang för att minska fiendens såväl störnings- som avlyssningsförmåga. Numera används bandspridning även i civila tillämpningar för att minska inverkan av ”dåliga frekvenser” och fädning. En tillämpning är WCDMA (Wide Band Code Division Multiple Access) i 3G-systemet.

Om den sända signalen har undergått bandspridning i sändaren kommer mottagaren först att samla den bandspridda signalen. Därefter omvandlar demodulatorn vågformsignalerna till en ström av diskreta, ofta binära, symboler i en symbolbeslutskrets. Men för att beslutet skall bli så bra som möjligt får vågformsignalerna dessförinnan passera ett filter (utjämnare)



Figur 1.2 Sändare uppdelad i funktionsblock.



Figur 1.3 Mottagare uppdelad i funktionsblock.

som på bästa sätt strävar efter att kompensera för olinjäriteter och tidsvariationer i kanalen.

De redundanta bitar som tillfogats i sändarens kanalkodare används av mottagarens kanalavkodare för att korrigera de bitfel som kan uppstå vid överföringen. I de två sista blocken: avkrypterare och källavkodare utförs de omvända operationerna till de som utförts i sändarblocken: krypterare och källkodare.

1.3 Krypteringens uppgift

Denna bok behandlar den del av informationsöverföringen som har att göra med kryptering i sändaren och dekryptering i mottagaren eller hur datasäkerhet åstadkoms. Om man närmare analyserar de krav som bör ställas på datasäkerhet kan de delas upp i ett antal delkrav enligt:

Konfidentialitet

Privata samtal eller annat informationsutbyte mellan två parter skall ej kunna avlyssnas, d.v.s. meddelandehemligheten måste garanteras.

Dataintegritet

Informationsöverföring måste garanterat inte utsättas för manipulation av obehörig.

Autenticering

Mottagaren måste få sändarens autenticitet bekräftad och säkerställd, d.v.s. att han är den han uppger sig vara.

Dataursprungsautenticering

Bekräftelse om att sänd information är identisk med mottagen.

Digital signatur

Ett sätt att koppla en ansvarig till ett meddelande.

Dessutom finns ett antal vanliga datakommunikationsprotokoll såsom:

Tidsstämpling som anger när ett meddelande sänts eller mottagits.

Kvitto som anger att informationen mottagits.

2 Kryptografi

2.1 Inledning

I dagens elektroniska värld har andelen meddelanden och finansiella transaktioner i form av t.ex. brevförsändelser och manuellt bokförda banktransaktioner gradvis minskat till förmån för betydligt snabbare elektroniska motsvarigheter via internet. Eftersom det vid meddelandeöverföring såväl via tråd som trådlöst alltid är möjligt för en obehörig att avlyssna trafiken är det nödvändigt att se till att denne inte kan tillgodogöra sig meddelandets innehåll. Ett vanligt sätt att göra meddelandet obegripligt är att kryptera det.

Vetenskapen om krypteringssystem (eller chiffersystem) kallas kryptografi. Kryptografi är en väletablerad vetenskap som haft ett betydande inflytande på historiens gång under flera tusen år. Förr i världen – före c:a 1970 – användes kryptografi främst av regeringar och i militära sammanhang men det tidigast dokumenterade exemplet på ett krypterat meddelande utgörs av en gravinskription utförd 1900 år före vår tideräkning. I inskriptionen som hittades i Egypten hade en del ovanliga hieroglyfer infogats här och där i texten. Även om detta inte gjorde texten obegriplig är det dock det tidigast kända exemplet på an avsiktlig ändring av skriften. En ändring eller transformation av texten är ett väsentligt element i kryptografi.

Många exempel på kryptografi och dess betydelse i historien finns dokumenterade i litteraturen och på film och i många filmer om andra världskriget har vikten av att kunna knäcka (eller forcera) krypterade meddelanden (eller chiffer) visats. Exempelvis har sådana viktiga händelser som hur tyskarnas Enigmachiffer knäcktes och också knäckandet av japanska kodade meddelanden strax innan attacken mot Pearl Harbour visats på film och TV.

Numera har kryptografi blivit en väletablerad disciplin som lärs ut vid många universitet och högskolor och kryptografi används av företag och enskilda människor. Den snabba ökningen av antalet användare har flera orsaker men den främsta är expansionen av internettrafiken. Vid överföring via internet, vare sig det gäller ekonomiska transaktioner eller textmeddelanden, är det inte bara viktigt att

1. meddelandehemligheten bevaras

utan även att

2. identiteten hos de kommunicerande kan garanteras.

Dessutom kan det förekomma situationer vid vilka det är viktigt att

3. den mottagande parten är försäkrad om att den sändande inte vid ett senare tillfälle kan förneka att han sänt det mottagna meddelandet och påstå att han sänt något annat.

De nämnda problemen är viktiga men inte så lätta att lösa. Vid vanliga icke-elektroniska affärstransaktioner är handskrivna underskrifter vanliga för att försäkra sig om alla tre punkterna, även om en underskrift naturligtvis inte är någon absolut garanti.

Därför har det blivit en stor utmaning att utforma ”elektroniska motsvarigheter” till den igenkänning som uppstår vid ett möte ansikte mot ansikte och eventuellt ett bekräftande handslag eller den verifikation som i ett brev fås genom underskrift med en handskriven signatur. Sådana mellanmänskliga signaler går ju helt förlorade vid övergång till digitala transaktioner.

1976 publicerade Diffie och Hellman en viktig uppsats (Directions in Cryptography) i vilken de föreslog ett sätt att använda kryptografi för att åstadkomma elektroniska ekvivalenter till handskrivna signaturer. Före Diffies och Hellmans uppsats hade kryptografi huvudsakligen nyttjats för att säkerställa att meddelanden inte ändrades under transmissionen.

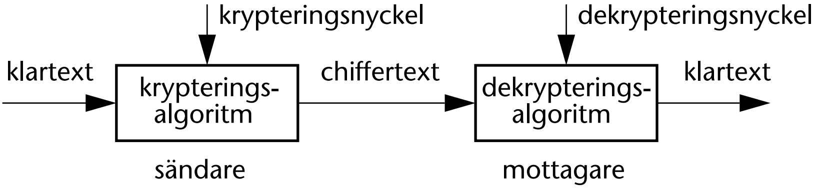
Detta byggde på de kommunicerande parternas ömsesidiga förtroende, vilket inte utgjorde något större problem när det gällde t.ex. transaktioner mellan banker eller andra finansiella institutioner som var de huvudsakliga användarna före 60- och 70-talet.

Modern kryptografi har utvecklats avsevärt sedan dess. Inte bara tekniken utan även tillämpningarna. Dessutom har praktiskt taget varje människa blivit påverkad indirekt eller direkt som användare. Därför är det viktigt att förstå hur kryptografi används och fungerar.

2.2 Grundläggande begrepp

Avsikten med ett krypteringssystem (eller chiffersystem) är att maskera ett meddelande på ett sådant sätt att innehållet blir obegripligt för obehöriga. Vanliga tillämpningar är vid lagring av data och överföring via osäkra kanaler som i internet. I alla dessa fall är det möjligt för obehöriga att avlyssna meddelandet men eftersom det är krypterat är det för den obehörige obegripligt.

Det meddelande som skall krypteras kallas *klartext* medan den operation som används för att maskera innehållet benämns *kryptering*. Den krypterade klartexten kallas *chiffertext*, *kryptotext* eller *kryptogram* medan de regler som används för att kryptera klartextmeddelandet benämns krypteringsalgoritmen. Vanligen är en kryptotext en funktion av inte bara en mer eller mindre allmän krypteringsalgoritm och klartexten utan också en speciell *krypteringsnyckel* som är en för varje användare speciell kod. För att i mottagaren återskapa den ursprungliga klartexten krävs en *dekrypteringsalgoritm* och en mot den speciella krypteringsnyckeln svarande *dekrypteringsnyckel*. Figur 2.1 visar schematiskt ett krypteringssystem.



Figur 2.1 *Modell av krypteringsprocessen.*

En obehörig som uppsnappar chiffertexten vid överföringen och som utan dekrypteringsnyckel (men eventuellt med tillgång av dekrypteringsalgoritmen) försöker ta reda på klartexten kallas *forcör.*

Medan kryptografi är benämningen på den vetenskap som behandlar konstruktionen av krypteringssystem, är *kryptoanalys* den process som används för att ta reda på innehållet i chiffertexten (klartexten) utan att äga kännedom om den riktiga dekrypteringsnyckeln.

Kryptologi är den kollektiva termen för kryptografi och kryptoanalys. Man bör dock notera att kryptoanalys inte är den enda metoden för att forcera en chiffertext. Utskrivna klartextmeddelanden eller dekrypteringsnycklar som kvarglömts på platser där en forcör kan få tillgång till dem visar att det är många säkerhetsdetaljer som förutom en bra krypteringsalgoritm är viktiga för att säkra meddelandehemligheten.

Viktigast för säkerheten hos ett krypteringssystem är nycklarnas säkerhet. I praktiken koncentreras de flesta kryptoanalytiska attacker på att försöka bestämma dekrypteringsnyckeln och vanligtvis innebär meningen med att ett krypteringssystem har forcerats (eller knäckts) att man hittat en metod att bestämma dekrypteringsnyckeln. Av ovanstående bör det framgå att kännedom om krypteringsnyckeln inte är nödvändig för att bestämma klartexten från chiffertexten. Denna observation utgör kärnan i Diffie-Hellmans uppsats och har starkt påverkat modern kryptologi och lett till en indelning av krypteringssystem i *symmetriska* och *asymmetriska*. Vanliga eller symmetriska krypteringssystem har ett så enkelt samband mellan krypterings- och dekrypteringsnyckeln att den senare kan bestämmas ur den förra. I själva verket är dessa två nycklar identiska. Av den anledningen kallas sådana system *hemlig nyckel*- eller *enhetsnyckelsystem*.

Om det däremot är så gott som omöjligt att bestämma dekrypteringsnyckeln med hjälp av krypteringsnyckeln kallas systemet asymmetriskt. Under alla omständigheter är det, för att förhindra att en obehörig, med tillgång till krypterings- och/eller dekrypteringsalgo- ritmen får tillgång till motsvarande klartext med hjälp av uppsnappad chiffertext, viktigt att dekrypteringsnyckeln hålls hemlig. I själva verket kan i ett asymmetriskt krypteringssystem krypteringsnyckeln göras offentlig. En konsekvens av detta är att sändaren och mottagaren inte behöver dela några hemligheter med varandra.

Generering, distribution och hemlighållande av nycklar hör till de svåraste problemen att lösa när säkerheten hos praktiska krypteringssystem skall garanteras. När det gäller asymmetriska krypteringssystem, inskränker sig problemet till att distribuera och hemlighålla dekrypteringsnycklarna. En viktig princip inom kryptografi är att systemets säkerhet inte skall bero på att man lyckas hålla krypteringsalgoritmen hemlig utan bara på att dekrypteringsnyckeln hålls hemlig.

Ett av syftena med att studera kryptografi är att kunna bedöma om ett krypteringssystem är tillräckligt säkert för den avsedda tillämpningen. För att fastställa systemets säkerhet bedömer man säkerheten under förutsättning av vart och ett av följande ”worst-case”:

1. kryptoanalytikern har fullständig kännedom om krypteringssystemet.
2. kryptoanalytikern har lyckats få tag i en avsevärd mängd krypterad text.
3. kryptoanalytikern känner till klartexten för en viss mängd kryptotext.

Vad som skall menas med de svävande begreppen ”avsevärd mängd” och ”viss mängd” får bestämmas från fall till fall.

Fallet 1 medför att det inte är någon idé att försöka hemlighålla olika delar i krypteringssystemet. Å andra sidan bör systemet inte avsiktligt göras offentligt. Om kryptoanalytikern inte känner till systemet har han naturligtvis en betydligt svårare uppgift, även om han skulle ha tillgång till hårdvaran. Fallet 1 utgör en väsentlig förutsättning för att minska ansvaret för systemets hemlighållande.

Fallet 2 utgör en rimlig förutsättning. Om avlyssning är möjlig måste man i värsta fall utgå från att all kommunikation kan avlyssnas.

Fallet 3 utgör också en realistisk förutsättning som kan delas upp i följande tre delar:

a. Attack på känd klartext, vilket innebär att kryptoanalytikern använder sig av en känd klartext tillsammans med motsvarande chiffertext.

b. Attack på vald klartext, vilket innebär att kryptoanalytikern haft möjlighet att påverka valet av klartext. Som exempel på detta kan nämnas ett fall från andra världskriget då en lysboj bombades i syfte att försäkra sig om att det tyska ordet Leuchttonne skulle ingå i den klartext som krypterades i den s.k. Enigmakrypteringsmaskinen.

c. Attack på endast chiffertext innebär att kryptoanalytikern endast har kännedom om chiffertexten.

En förutsättning för att dessa worst-case scenarier kan ställas upp är att det enda som skiljer den avsedda meddelandemottagaren från kryptoanalytikern är den förres kännedom om dekrypteringsnyckeln d.v.s. systemets sekretess är helt och hållet avhängig sekretessen hos dekrypteringsnyckeln. Slutsatsen blir att ett gott handhavande av dekrypteringsnyckeln är av yttersta vikt. Dock bygger alla förutsättningar på antaganden om kryptoanalytikern och vilka texter han har tillgång till. Vid tveksamheter bör man förutsätta värsta fallet och den relevanta fråga man bör ställa sig är inte om ”systemet är exceptionellt säkert” utan hellre om ”systemet är tillräckligt säkert för den aktuella tillämpningen”.

Eftersom krypteringssystem är komplicerade och dyra och ofta medför reducerad överföringshastighet är det viktigt att systemets säkerhet minimeras. Vanligen försöker man uppskatta *täckningstiden* (eng. *cover time*) d.v.s. hur länge meddelandet behöver hemlig-hållas. Täckningstiden för meddelanden i militärtaktiska situationer kan vara några minuter medan den för regeringshemligheter och medicinska dokument kan uppgå till årtionden.

Om kryptoanalytikern känner till dekrypteringsalgoritmen skulle han i princip kunna pröva varje dekrypteringsnyckel med hopp om att hitta den riktiga. En sådan attack kallas *uttömmande nyckelsökning* (eng. *exhaustive key search* eller *brute force attack*). Denna metod förutsätter att kryptoanalytikern har något sätt att identifiera den riktiga nyckeln eller att åtminstone förkasta de felaktiga.

I en attack med vald klartext kan alla dekrypteringsnycklar, som inte ger den riktiga klartexten, genast elimineras. Oturligt nog för kryptoanalytikern kan det finnas många dekrypteringsnycklar som ger riktiga klartexter för motsvarande chiffertexter om antalet klartext/chiffertextpar inte är tillräckligt stort. Ett sätt att eliminera en del felaktiga dekrypteringsnycklar sammanhänger med den statistiska fördelningen av bokstäverna i det i klartexten använda språket.

För att specificera ett krypteringssystem för en viss tillämpning skall systemanvändaren ange en täckningstid. Konstruktören å sin sida måste veta hur många dekrypteringsnycklar systemet har. Han måste med ledning av detta söka uppskatta hur lång tid en kryptoanalytiker behöver för att testa varje dekrypteringsnyckel. Detta ger en möjlighet att bedöma hur lång tid en uttömmande nyckelsökning tar. Om denna tid är kortare än den specificerade täckningstiden måste systemet förbättras. Ett grundläggande krav på ett krypteringssystem är att den uppskattade tiden för en uttömmande nyckelsökning är väsentligt längre än den specificerade täckningstiden.

Innan vi går vidare ska vi i nästa kapitel titta på några historiska exempel på kryptering för att illustrera teorin.

2.3 Chiffer i historien

2.3.1 Inledning

I detta kapitel beskrivs några ”historiskt tidiga” exempel för att illustrera de grundläggande begreppen i avsnitt 2.2 och för att visa några attacker som kryptoanalytiker kan göra. Alla dessa historiska algoritmer är symmetriska.

2.3.2 Caesars chiffer

Ett av de äldsta kända chiffren härstammar från Julius Caesar[1]. Chiffret använde han under de galliska krigen[2]. Chiffret fungerar så att varje bokstav i alfabetet byts mot den som fås genom ett cykliskt skift tre steg framåt i alfabetet. Bokstäverna Å, Ä och Ö representeras därvid av A, B och C. Fastän Caesar använde ett skift på tre kan naturligtvis en liknande effekt uppstå med ett godtyckligt skift mellan 1 och 27.

Caesars chiffer är ett exempel på ett monoalfabetiskt substitutionschiffer i det att varje enskild bokstav substitueras med en annan bokstav.

En uttömmande sökning på chiffret DQIDOO kan åskådliggöras enligt Tabell 2.1.

Av Tabell 2.1 framgår att Caesars chiffer är ganska enkelt att knäcka.

---

1 Gaius Julius Caesar (100 - 44 f. Kr.).
2 (58 - 49 f. Kr.).

skift	meddelande	skift	meddelande	skift	meddelande	skift	meddelande
0	DQIDOO	7	KYPKVV	14	RCXRAA	21	ZJBZHH
1	ERJEPP	8	LZQLXX	15	SDYSBB	22	ÅKCÅII
2	FSKFQQ	9	MÅRMYY	16	TEZTCC	23	ÄLDÄJJ
3	GTLGRR	10	NÄSNZZ	17	UFÅUDD	24	ÖMEÖKK
4	HUMHSS	11	OÖTOÅÅ	18	VGÄVEE	25	ANFALL
5	IVNITT	12	PAUPÄÄ	19	XHÖXFF	26	BOGBMM
6	JXOJUU	13	QBVQÖÖ	20	YIAYGG	27	CPHCNN

Tabell 2.1 Exempel på uttömmande nyckelsökning: kryptogrammet DQIDOO.

2.3.3 Polybius kvadrat

Med Polybius[3] kvadrat arrangeras bokstäverna i en kvadrat enligt Figur 2.2. Först kombineras bokstäverna I och J och behandlas som en enda bokstav eftersom det efter dekryptering är lätt att av sammanhanget förstå vilken av dem som avses. Kryptering av klartexten utförs genom att för varje bokstav ange kolumn och radnummer i kvadraten enligt nedanstående exempel:

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	IJ	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Figur 2.2 Polybius kvadrat.

3 Polybius, grekisk historiker (c:a 200 - 118 f. Kr.).

klartext:	N	U	K	O	M	M	E	R	B	I	L	E	N
chiffertext:	33	54	52	43	23	23	51	24	21	42	13	51	33

Koden kan ändras genom att bokstäverna i kvadraten placeras i en annan ordning.

2.3.4 Trithemius progressiva nyckel

Trithemius[4] progressiva nyckel – se Figur 2.3 – är ett exempel på ett polyalfabetiskt chiffer. Raden med nummer 0 motsvarar den vanliga placeringen av bokstäverna i alfabetet. De därpå följande raderna är cykliskt skiftade lika många steg som anges i vänsterkolumnen. Ett sätt att använda detta polyalfabet är att för den första chiffertextbokstaven använda rad 1, den andra chifferbokstaven rad 2 o.s.v. enligt nedanstående exempel:

klartext:	h	u	r	s	t	å	r	d	e	t	t	i	l	l
chiffertext:	I	X	U	X	Z	D	Z	L	N	B	C	U	Z	Å

4 Trithemius, tysk abbot, magiker och alkemist (1462 - 1516).

klartext:		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	x	y	z	å	ä	ö
skift:	0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö
	1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A
	2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B
	3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C
	4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D
	5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E
	6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F
	7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G
	8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H
	9	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I
	10	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J
	11	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K
	12	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L
	13	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M
	14	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	15	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	16	Q	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	17	R	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
	18	S	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
	19	T	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
	20	U	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
	21	V	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
	22	X	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
	23	Y	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X
	24	Z	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y
	25	Å	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z
	26	Ä	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å
	27	Ö	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z	Å	Ä

Figur 2.3 *Trithemius´ progressiva nyckel.*

2.3.5 Vigenères nyckelmetod

1585 publicerade Vigenère[5] verket Tracte des Chiffres i vilket han beskrev ett nytt sätt att använda Trithemius polyalfabet. På *ett* sätt som beskrivs används en startbokstav följt av klartexten som nyckel, d.v.s. nyckeln anger vilken rad som skall användas vid kryptering och dekryptering – se nedanstående exempel där f valts som startbokstav:

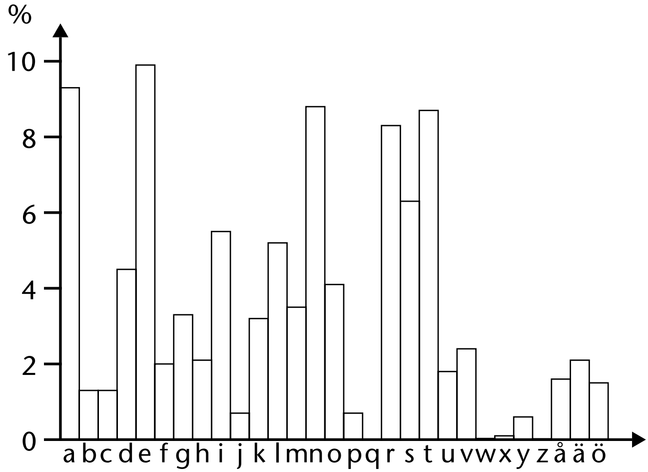
nyckel:	f	h	u	r	s	t	å	r	d	e	t	t	i	l
klartext:	h	u	r	s	t	å	r	d	e	t	t	i	l	l
chiffertext:	M	Ö	J	H	J	Q	O	U	H	Y	K	Ö	T	X

Av exemplet framgår att varje bokstav kan substitueras med olika bokstäver.

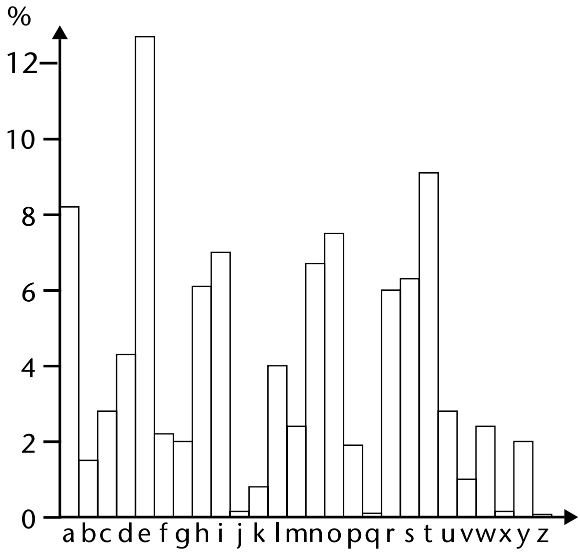
Det finns många varianter på Vigenère-chiffer. En variant som används består av ett nyckelord som upprepas periodiskt. Antalet bokstäver i nyckeln är nyckelns period.

Substitutionsmetoder som denna kallas polyalfabetiska. Med polyalfabetisk substitution förändras tecknens relativa frekvenser i ett chiffer och därmed försvåras sådana attacker som kan användas vid monoalfabetiska chiffer och som bygger på att bokstäverna förekommer med ganska bestämda relativa frekvenser – se Figur 2.4 och Figur 2.5.

5 Blaise de Vigenère (1523 - 1596) fransk kryptograf.



Figur 2.4 Relativ förekomst av bokstäver i svensk text.



Figur 2.5 Relativ förekomst av bokstäver i engelsk text.

De metoder som angivits av Vigenère utgör idag en väsentlig del av ett standardiserat krypteringssystem som heter DES (Data Encryption Standard).

2.4 Moderna krypteringsmetoder

2.4.1 Inledning

Exemplen i avsnitt 2.3 är enkla. De flesta kan lätt knäckas även om detta inte gällde vid den tid då de konstruerades. Kryptoanalys består av en hel del ”trial and error” och datorer har förenklat denna metod. En sådan trial and error-metod är uttömmande nyckelsökning (omnämnd i avsnitt 2.2). En uttömmande nyckelsökning av ett Vigenère-chiffer som har en sexbokstavsnyckel (28^6 möjliga kombinationer) tar mindre än en dag med en dator som kan testa 10.000 sexbokstavsnycklar i sekunden. På 1500-talet hade det varit praktiskt omöjligt.

I samband med att man diskuterar moderna krypteringsmetoder kommer frågan om obrytbara chiffer upp. I historisk tid har många konstruktörer av krypteringsalgoritmer hävdat att deras chiffer var obrytbara. Sådana påstående ledde ofta till fruktansvärda konsekvenser. Detta gällde t.ex. den skotska drottning Mary som fängslats av drottning Elisabeth av England. Från fängelset korresponderade hon via krypterade meddelanden med sina sammansvurna om hur hon skulle mörda drottning Elisabeth och överta den engelska tronen. Även om någon skulle få tag i korrespondensen var drottning Mary övertygad om att det inte skulle vara möjligt att dekryptera innehållet. Men drottning Elisabeths huvudsekreterare Francis Walsingham som fått tag i några av de utsmugglade breven var också chef för vad som skulle kunna kallas den tidens kontraspionage. Han dechiffrerade innehållet och vid rättegången mot drottning Mary användes breven som bevis mot henne. Hon avrättades genom halshuggning 1587.

Under andra världskriget använde sig tyskarna av en krypteringsmaskin som gick under namnet Enigma. Krypteringsmekanismen i Enigma var mycket komplicerad och hade 10^{30} möjliga nycklar som faktiskt är mer än en del moderna algoritmer har. Användarna var övertygade om att Enigmas chiffer var obrytbara. De allierade lyckades dock vid flera tillfällen knäcka Enigmas chiffer delvis tack vare operatörernas handhavandefel. Arbetet med att knäcka Enigmas chiffer koncentrerades till Bletchley Park i England – numera museum. Det uppskattas att arbetet i Bletchley Park förkortade kriget med åtminstone två år.

2.5 Sekretess hos chiffersystem

2.5.1 Perfekt sekretess

Vi utgår från ett system med de N meddelandena: $M_0, M_1,\ldots, M_{N-1}$ och de U chiffertexterna: $C_0, C_1,\ldots, C_{U-1}$. $P(M_i)$ är apriorisannolikheten att M_i sänts. $P(M_i|C_j)$ är aposteriorisannolikheten att M_i sänts med villkoret att C_j mottagits.

Ett kryptosystem sägs ha *perfekt sekretess* om det gäller

$$P(M_i|C_j) = P(M_i) \tag{2.1}$$

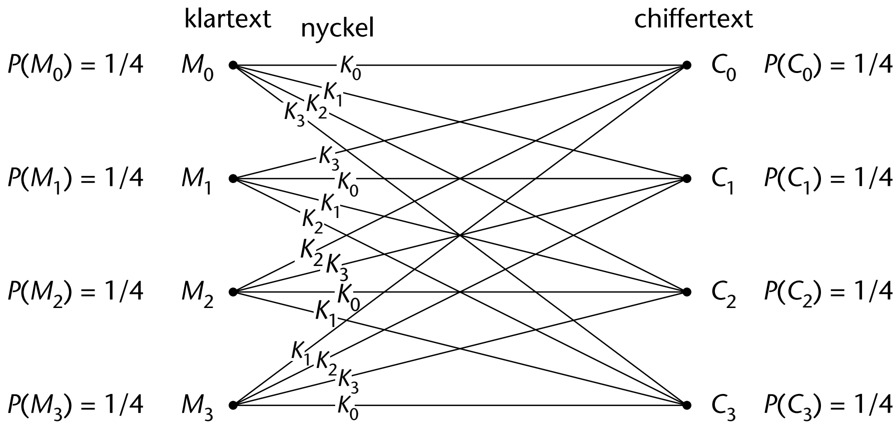
för alla meddelande och chiffertexter. Detta medför att en forcör som fått tag i en chiffertext, C_j, inte kan lista ut vilket meddelandet är eftersom han ju inte får någon annan ytterligare information.

För att perfekt sekretess skall råda är det nödvändiga och tillräckliga villkoret:

$$P(C_j|M_i) = P(C_j) \tag{2.2}$$

I Figur 2.6 visas ett exempel på överföring med perfekt sekretess. De fyra klartextmeddelandena är M_0, M_1, M_2 och M_3 medan chiffertexterna är C_0, C_1, C_2 och C_3. Nycklarna K_0, K_1, K_2 och K_3 genererar C_0, C_1, C_2 och C_3. Dessutom gäller att sannolikheterna $P(M_i) = P(C_j)$ är lika stora, d.v.s. 1/4.

Uttrycket för klartext-chiffertext-transformation (eller klartext-chiffertext-kryptering) kan skrivas (T_K betecknar transformation, eller kryptering, med nyckeln K_j)



Figur 2.6 Ett exempel på ett system med perfekt sekretess och fyra klartextmeddelanden.

$$C_s = T_{K_j}(M_i) \tag{2.3}$$

och $s = (i + j)$ modulo-N (d.v.s. $s = 0,1,2$ och 3).

Eftersom en forcör som snappar upp någon av chiffertexterna C_0, C_1, C_2 eller C_3 inte har någon möjlighet att avgöra vilken av nycklarna K_0, K_1, K_2 eller K_3 som använts, kan han heller inte avgöra vilket av klartextmeddelandena M_0, M_1, M_2 eller M_3 som sänts.

Ett krypteringssystem som har lika många klartextmeddelanden, som nycklar och som chiffertexter har perfekt sekretess om det bara finns en enda nyckel som transformerar varje klartextmeddelande till varje chiffertext och dessutom alla nycklar är lika sannolika.

Om dessa villkor inte är uppfyllda kan det ju faktiskt finnas ett klartextmeddelande, kalla det M_i, så att en chiffertext, C_j, inte med någon nyckel kan dechiffreras till M_i. Forcören drar då slutsatsen att $P(M_i|C_j) = 0$ för denna kombination av i och j, d.v.s. detta klartextmeddelande kan uteslutas och eventuellt kan han hitta fler kombinationer som på samma sätt kan uteslutas. Varje sådant uteslutande reducerar antalet möjliga kandidater och gör att forcören kommer närmare lösningen. Ett kryptosystem med perfekt sekretess är det som mest av allt eftersträvas eftersom ett sådant system är ovillkorligt säkert, men ovillkorligt säkra system är opraktiska för oftast vill man ju inte ha någon begränsning av hur många meddelanden som kan skickas och eftersom varje meddelande kräver en egen nyckel blir antalet nycklar i princip hur stort som helst.

2.5.2 One time pad

En viktig konsekvens av diskussionen om perfekt sekretess är att den går att uppnå men endast till priset av oerhörda mängder nycklar som måste administreras och överföras till mottagaren via en säker kanal. Det klassiska exemplet på ett perfekt chiffersystem är ”the one time pad”. Krypteringsprincipen är densamma som för Vigenères nyckelmetod – se 2.3.5 – som med en mer generell variant har en slumpmässigt vald nyckel av samma längd som meddelandet. Att nyckellängden är lika stor som meddelandelängden garanterar att perfekt sekretess uppnås. Vigenères metod i binär form patenterades av Vernam som gav den namnet ”one time pad[6]”.

Vernam patenterade sin idé i hopp om att den skulle få stor kommersiell spridning. Som vi redan sett är den största nackdelen med ett ovillkorligt säkert kryptosystem att den mängd nycklar, som måste hanteras och överföras säkert, åtminstone är lika stor som antalet meddelanden. Så kräver t.ex. ett meddelande om n bitars klartext en nyckellängd på n bitar. Detta skulle inte vara något större problem om samma nyckel kunde användas för att kryptera olika meddelanden. Men säkerheten i ovillkorligt säkra kryptosystem är direkt avhängig av att varje nyckel används endast en gång, därav namnet *one time* pad.

Exempel: Vid kryptoanalys av det tyska Enigma kryptosystemet utbyttjades operatörsslarv i form av att en del one-time chiffer användes mer än en gång.

Svårigheten att administrera de många nycklarna har begränsat one time paden vid kommersiella användningar, men i militära sam-

6 One time pad (U.S. Patent number 1310719, 22 juli 1919) uppfanns 1917 av Gilbert Sandford Vernam (1890 - 1960).

manhang och inom diplomatin där ovillkorlig säkerhet är av största vikt har one time paden funnit tillämpningar.

Historiskt sett har utvecklingen av kryptografin gått mot att konstruera system där *en* nyckel kan användas för att kryptera många meddelanden och ändå bibehålla ”tillräcklig säkerhet”. Ett sätt att definiera ”tillräcklig säkerhet” är att säga att systemet är ”beräkningsmässigt säkert” under x år. Med detta menas att chiffertexten inte kan forceras på kortare tid än x år med hjälp av dagens mest avancerade datorer.

Kryptosystem av sådant slag innefattar DES (Data Encryption Standard) och AES (Advanced Encryption Standard), system som bahandlas längre fram.

2.6 Moderna algoritmer

Kryptering kan indelas i *blockkryptering* och *strömkryptering*. Vid blockkryptering indelas klartexten i block med konstant längd och varje sådant block krypteras oberoende av de andra.

Exempel: Blockkryptering kan användas för att kryptera ett hemligt meddelande genom att bokstäver, siffror, mellanslag och andra tecken först koda till binära siffror (noll och ett). En vanlig kod är ASCII-koden (American Standard Code for Information Exchange). Den uppkomna bitsekvensen krypteras därefter till chiffertexten som utgörs av en annan bitsekvens. ASCII-koden använder 8 bitar för varje tecken och för ett blockchiffer med 64-bits block krypterar krypteringsalgoritmen 8 tecken i taget.

Om en och samma nyckel används kommer därför ett stycke klartext alltid att transformeras till samma chiffertext närhelst det dyker upp.

Vid strömkryptering finns ingen fix blocklängd utan varje klartextbit, m_i, krypteras med det i:te elementet, k_i, i en symbolsekvens som genereras av en nyckel. Krypteringen är *periodisk* om nyckelsekvensen alltid upprepas efter ett visst antal symboler. Om nyckelsekvensen inte är periodisk är den *aperiodisk*.

2.7 Krav på ett krypteringssystem

Ett kryptosystem skall förse alla behöriga användare med ett enkelt och billigt system för kryptering och dekryptering och göra det svårt och dyrt för en forcör att utan nyckel kunna dechiffrera en kryptotext.

Kryptosystem är antingen *ovillkorligt säkra* eller *beräkningsmässigt säkra*. Ett ovillkorligt säkert system innehåller inte tillräcklig mängd information för att forcören skall kunna bestämma krypterings- och dekrypteringstransformationen. Ett exempel på ett ovillkorligt säkert kryptosystem är en *"one time pad"*. Med en one-time pad krypteras ett meddelande med en *slumpnyckel* som skall användas endast *en* gång. Om den används fler gånger kan forcören utvinna tillräckligt med information för att kunna forcera chiffret. Eftersom ett ovillkorligt säkert system kräver en ny nyckel för varje nytt meddelande åtgår det också en omfattande apparat (dyr och långsam) för administration och distribution av nycklar. I de flesta kryptosystem nyttjas istället system, som är beräkningsmässigt säkra under x år – se även näst sista stycket i avsnitt 2.5.2.

2.8 Teoretiska grunder

2.8.1 Entropi

I föregående avsnitt behandlades begreppet perfekt sekretess, som endast kan uppnås med en nyckel, som används för bara ett enda meddelande. Hur troligt är det då att en forcör lyckas knäcka en chiffertext om en och samma nyckel används till fler meddelanden? För att lösa detta problem använder vi oss av ett ”verktyg” som kallas informationsteori introducerat av Claude Shannon 1948[7].

Centralt begrepp i informationsteorin är entropi. Entropi är ett matematiskt mått på information eller osäkerhet och beror av en sannolikhetsfunktion (eller fördelningsfunktion). Betrakta en diskret stokastisk variabel X med ändligt många värden $x_1, x_2, \ldots, x_M$ och en given (eller specificerad) fördelningsfunktion. Hur mycket information erhålls genom att man får kännedom om utfallet av ett experiment, som kan modelleras med den givna fördelningsfunktionen?

Frågan är ekvivalent med att fråga hur stor osäkerheten är i utfallet innan experimentet ägt rum. Svaret på frågan kallas informationen alternativt entropin för X och betecknas $H(X)$.

Exempel 2.1

Låt den stokastiska variabeln X representeras av experimentet ”slantsingling”. För ett välgjort mynt är sannolikheten för de två möjliga utfallen krona och klave lika stora d.v.s. P(krona) = P(klave) =1/2. Utfallen kan representeras av 1 och 0. Informationen om utfallet krona respektive klave kan alltså representeras av ett enbits binärt tal och informationen eller entropin för utfallet kallas därför 1 bit (plural: bitar)

Exempel 2.2

Om myntet kastas n gånger efter varandra, kan utfallet för experimentet ”n av varandra oberoende kast” representeras av ett binärt tal med n bitar och informationen eller entropin, som erhålls genom utfallet av experimentet är n bitar.

◊

Ovanstående exempel illustrerar hur information definieras för experiment där utfallen är lika sannolika, d.v.s. den stokastiska variabelns fördelningsfunktion är likformig.

---

7 C. E. Shannon, ”Communication Theory of Secrecy Systems”, *Bell Systems Technical Journal*, 28 (1949), pp. 656-715.

Exempel 2.3

Betrakta ett exempel som kan representeras av en stokastisk variabel med de möjliga utfallen x_1, x_2 och x_3 med sannolikheterna 1/2, 1/ 4 och 1/4.

x_1, x_2 och x_3 anges med de binära talen 0, 10 och 11 och följaktligen är

$P(x_1) = P(0) = 1/2$

$P(x_2) = P(10) = 1/4$

$P(x_3) = P(11) = 1/4$

Det är tydligen så att informationen (eller entropin) för utfallet är 1 eller 2 bitar men det genomsnittliga antalet är

$P(x_1) \cdot 1$ (bit) $+ P(x_2) \cdot 2$ (bitar) $+ P(x_3) \cdot 2$ (bitar) $=$

$$= \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 = \frac{3}{2}$$

◊

En matematisk funktion som modellerar antalet bitar i ett utfall x, som inträffar med sannolikheten $P(x)$ är $1/\log_2 P(x)$ och med ovanstående exempel får vi

$$P(x_1) \cdot \frac{1}{\log_2 P(x_1)} + P(x_2) \cdot \frac{1}{\log_2 P(x_2)} + P(x_3) \cdot \frac{1}{\log_2 P(x_3)} =$$

$$= \frac{1}{2} \cdot \frac{1}{\log_2 \frac{1}{2}} + \frac{1}{4} \cdot \frac{1}{\log_2 \frac{1}{4}} + \frac{1}{4} \cdot \frac{1}{\log_2 \frac{1}{4}} =$$

$$= \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 = \frac{3}{2}$$

vilket kan generaliseras till följande definition för entropin:

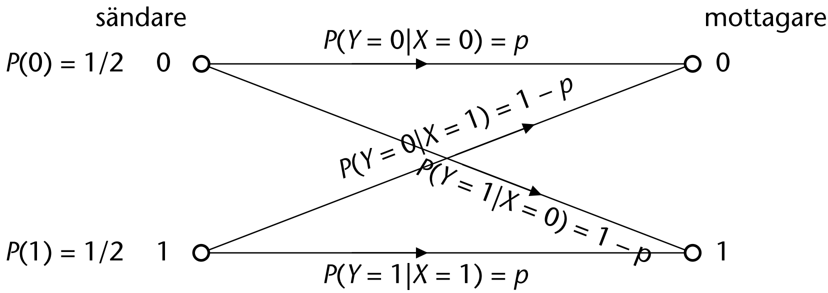
$$H(X) = \sum_{i=1}^{M} P(x_i) \log_2 \frac{1}{P(x_i)} \tag{2.4}$$

2.8.2 Betingad entropi

Exempel 2.4

Antag att vi överför information som kan representeras av binära siffror eller bitar med hastigheten 1000 bitar/s via en binär symmetrisk kanal enligt Figur 2.7. Sannolikheten för fel i överföringen är p och apriorisannolikheten (infördelningen) $P(x = 0) = P(x = 1) = 1/2$. Om brusnivån i överföringen är så stor att sannolikheten att mottaga en etta är 1/2 oberoende av vad som sänts och på samma sätt 1/2 att mottaga en nolla så skulle i ett sådant fall hälften av de mottagna symbolerna vara riktiga, d.v.s. informationsöverföringshastigheten skulle förefalla vara 500 bitar/s. Men eftersom man inte kan veta vilka bitar som är riktiga och vilka som är felaktiga så överförs ingen information alls.

Man kan lika gärna singla slant för att ange mottagarens utsignal som alltså är helt oberoende av vad som händer på sändarsidan och i kanalen.



Figur 2.7 Binär symmetrisk kanal (BSC).

◊

För att bestämma hur mycket information, som kommer fram till en mottagare via en störd kanal måste vi från den sända informationen $H(X)$ subtrahera den information som förloras i kanalen. Shannon beräknade hur mycket som skall subtraheras i form av en korrektionsterm[8] H(X|Y), som är den betingade entropin (eng. equivocation):

$$H(X|Y) = \sum_{i=1}^{M} \sum_{j=1}^{M} P(x_i,y_j) \log_2 \frac{1}{P(x_i|y_j)} =$$
$$= \sum_{j=1}^{M} P(y_j) \sum_{j=1}^{M} P(x_i|y_j) \log_2 \frac{1}{P(x_i|y_j)} \tag{2.5}$$

där $P(x_i,y_j)$ är den simultana sannolikheten för x_i och y_j.

Mottagen medelinformation (eller ömsesidig information) är

$$I(X,Y) = H(X) - H(X|Y) \tag{2.6}$$

Den betingade entropin kan uppfattas som osäkerheten i att meddelandet X sänts när Y mottagits. En forcör vill att $H(X|Y)$ skall minska allteftersom han samlar på sig mer och mer chiffertext Y.

Exempel 2.5

a. Bestäm entropin för en meddelandekälla som en gång i minuten genererar ett av följande lika sannolika meddelanden: x_1, x_2, …, x_8.

b. En mottagare mottar ett av de två lika sannolika meddelandena y_1 och y_2. Meddelandena y_j närmar sig de möjliga sända x_i på följande sätt:

Om y_1 mottagits vet man att endast x_1, x_2, x_3 eller x_4 är möjliga.

Om y_2 mottagits vet man att endast x_5, x_6, x_7 eller x_8 är möjliga.

Bestäm den betingade entropin $H(X|Y)$.

Lösning:

a. $P(X) = \frac{1}{8}$; $H(X) = 8 \cdot \frac{1}{8} \log_2 8 = 3$ [bitar/meddelande]

b. $P(Y) = \frac{1}{2}$; För varje y är $P(X|Y) = \frac{1}{4}$ för fyra av de åtta x:en och noll för de övriga. Med användande av uttrycket (2.5)

8 C. E. Shannon, Mathematical Theory of Communication, University of Illinois Press, 1963 (ursprungligen publicerad i *The Bell System Technical Journal*, vol. 27, pp. 379 and pp. 623-656, 1948).

$$H(X|Y) = \sum_{i=1}^{M} \sum_{j=1}^{M} P(x_i, y_j) \log_2 \frac{1}{P(x_i|y_j)} = \sum_{j=1}^{M} P(y_j) \sum_{j=1}^{M} P(x_i|y_j) \log_2 \frac{1}{P(x_i|y_j)}$$

får vi:

$$H(X|Y) = 2 \left[\frac{1}{2} 4 \cdot \frac{1}{4} \log_2 4 \right] = 2 \text{ [bitar/meddelande]}$$

Kännedom om Y har alltså reducerat osäkerheten om X från 3 bitar/meddelande till 2 bitar/meddelande.

◊

2.8.3 Ett språks takt och redundans

För ett språk definieras: *sann takt* eller *verklig takt* som medelantalet informationsbitar i varje bokstav och uttrycks för ett n bokstäver långt meddelande enligt:

$$r = \lim_{n \to \infty} \frac{H(X_1, X_2, \ldots, X_n)}{n} \qquad (2.7)$$

där X är språkets alfabet och X_1, X_2, … representerar tecken i alfabetet.

För engelska språket har för stora värden på n, r beräknats att ligga i intervallet 1,0 - 1,5 bitar/bokstav[9]. Den *maximala entropin r'* för ett språk definieras som den maximala informationsmängden/bokstav om alla möjliga bokstavssekvenser skulle vara lika sannolika och beräknas enligt:

$$r' = \log_2 L \qquad (2.8)$$

där L är antalet bokstäver i alfabetet. I engelska är $L = 26$ så att den maximala entropin för engelska är $\log_2 26 \approx 4{,}7$ [bitar/bokstav]. Den verkliga takten i engelska (och i de flesta andra språk) är alltså enligt

9 C.E. Shannon: Prediction and entropy of printed English. *Bell Sys. Tech. Journal*, 30; 50-64, 1951.

ovan mycket mindre eller den är mycket redundant och strukturerad.

Ett språks redundans (ung. överflöd), D, definieras enligt:

$$D = r' - r \tag{2.9}$$

För det engelska språket med $r' = 4,7$ bitar/bokstav och $r = 1,5$ bitar/bokstav blir redundansen $D = 4,7 - 1,5 = 3,2$ medan förhållandet $D/r' = 0,68$ är ett relativt mått på språkets (i detta fall engelska) redundans.

2.8.4 Entydighetsdistans och ideal sekretess

Enligt avsnitt 2.5.1 kräver perfekt sekretess, om vi tillåter meddelanden av obegränsad längd, ett oändligt antal nycklar. Med ändlig nyckellängd går den betingade entropin, $H(K|C)$ (där K är nyckeln och C är chiffertexten), generellt mot noll, vilket medför att nyckeln kan bestämmas och chiffret knäckas.

Entydighetsdistansen är den kortaste chiffertext som gör att den betingade entropin $H(K|C)$ blir nästan noll. Detta betyder helt enkelt att entydighetsdistansen är den kortaste chiffertextlängd, som krävs för att kunna bestämma nyckeln och därmed alltså knäcka chiffersystemet.

Shannon har angett ett system med *ideal sekretess*[10] i vilket den betingade entropin $H(K|C)$ inte går mot noll även om chiffertextlängden går mot oändligheten. Detta förhållande gör att forcören inte kan bestämma nyckeln hur mycket chiffertext han än kan kommer över.

Fastän ett system med ideal sekretess inte kan uppnå perfekt sekretess är det obrytbart, ovillkorligt säkert – tillräckligt med information för att kunna bestämma nyckeln kan inte erhållas.

10 Shannon, C. E., ”Communication Theory of Secrecy Systems”, *Bell Syst. Tech. J.*, vol. 28, Oct. 1949, pp. 656-715.

Ibland kan man göra en ungefärlig uppskattning av entydighetsdistansen. Enligt en metod beskriven av Hellman[11] utgår vi från ett klartextmeddelande och en chiffertext, båda med N bokstäver. Bokstäverna tas från ett alfabet med L bokstäver. Det finns därför $L^N =$ $= 2^{r'N}$ möjliga meddelanden med längden N.

Om man tittat på de olika möjliga meddelandena kan man genast se att de flesta är helt meningslösa. Sådana meningslösa meddelanden typ ahuys... kzx kan ju lätt sorteras bort. Vi kan placera alla sådana lätt bortsorterade meddelanden i en grupp, M_2, som vi kallar slumptextmeddelanden. Resten, d.v.s. meddelanden som verkar vara meningsfyllda placeras i en annan grupp, M_1. Det gäller då att det

$$\text{i grupp } M_1 \text{ finns } 2^{rN} \text{ meddelanden} \tag{2.10}$$

medan det

$$\text{i grupp } M_2 \text{ finns } 2^{r'N} - 2^{rN} \text{ (slumptext)meddelanden.} \tag{2.11}$$

Apriorisannolikheterna för de två grupperna är

$$P(M_1) = \frac{1}{2^{rN}} \tag{2.12}$$

Vi förutsätter att ingen skickar meningslösa meddelanden och därför är

$$P(M_2) = 0 \tag{2.13}$$

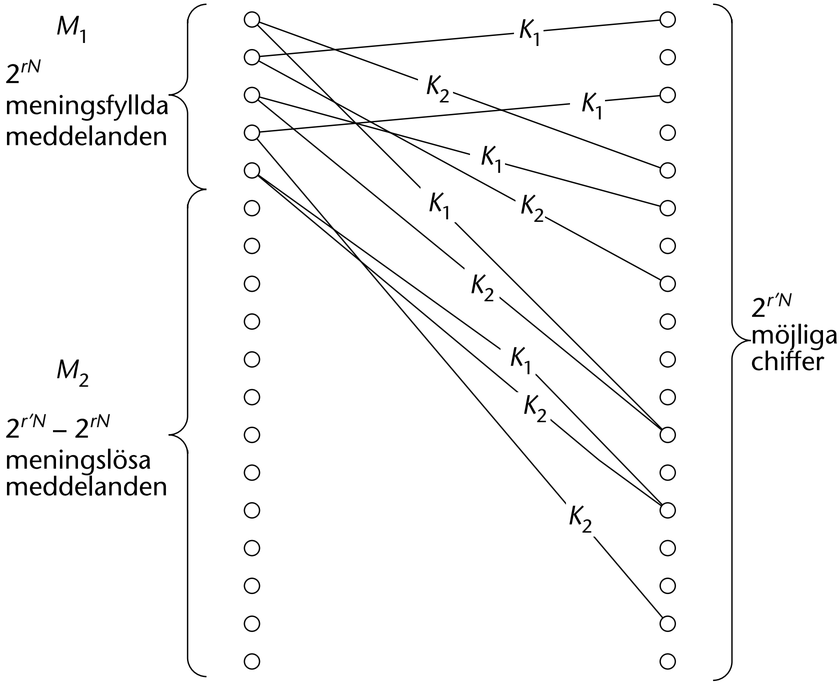
Nyckeln K är en sekvens bestående av $H(K)$ bitar, d.v.s. den har entropin $H(K)$. Detta medför att det finns $2^{H(K)}$ möjliga nycklar. Om nycklarnas sannolikhetsfördelning inte är känd är det enklast att förutsätta att den är likformigt fördelad därför är sannolikheten för en godtycklig nyckel:

$$P(K) = \frac{1}{2^{H(K)}} \tag{2.14}$$

11 Hellman, M. E., ”An Extension of the Shannon Theory Approach to Cryptography”, *IEEE Trans. Inf. Theory*, vol. IT 23, May 1978, pp. 289-294.

För att bestämma entydighetsdistansen utgår Hellman från en *slumpchiffermodell*. För en sådan modell gäller att en nyckel K och en chiffertext C genererar, med hjälp av dekrypteringsoperationen $D_K(C)$, en stokastisk variabel som är likformigt fördelad över alla $2^{r'N}$ *möjliga* meddelanden. Observera därvid att med möjliga meddelanden förstås alla kombinationer som kan bildas, d.v.s. meningslösa och meningsfyllda. Detta medför att vilket klartextmeddelande som helst kan bildas (d.v.s. med lika stor sannolikhet) av dekrypteringsoperationen $D_K(C)$ givet en nyckel K och en chiffertext C. Figur 2.8 visar en chifferrepresentation där endast två nycklar K_1 och K_2 är utritade. De meningslösa meddelandena M_2 är naturligtvis inte krypterade medan de meningsfyllda M_1 med alla olika nycklar kan krypteras till de $2^{r'N}$ möjliga. Lägg särskilt märke till det 14:e chiffermeddelandet. Det kan erhållas med båda nycklarna och det är fler sådana ”kollisioner” som gör chiffret obrytbart eller om varje möjligt chiffer kan bildas av ett antal nycklar, olika för varje chiffer, är chiffret obrytbart.

Om vi har en chiffertext C_i, som erhållits med transformationen $C_i = T_{K_i}(M_i)$, uppstår en *felaktig lösning*, F, om det finns någon annan nyckel, K_j (*falsk nyckel*), som utgående från M_i eller något annat klartextmeddelande, M_j, också kan krypteras till samma chiffertext C_i enligt:



Figur 2.8 Chifferrepresentation.

$$C_i = T_{K_i}(M_i) = T_{K_j}(M_i) = T_{K_j}(M_j) \tag{2.15}$$

En forcör som fått tag i chiffertexten kan därmed inte avgöra vilken nyckel som är den riktiga, eller, vilket är viktigare, han kan inte knäcka koden.

För varje riktig dekryptering motsvarande en viss chiffertext finns det $2^{H(K)} - 1$ falska nycklar tillgängliga, var och en med lika stor sannolikhet, $P(F)$, att producera en falsk dekryptering. Eftersom vi förutsatt att sannolikheten för varje klartextmeddelande är lika stor (likformig sannolikhetsfördelning) är alltså sannolikheten för varje oriktig dekryptering, $P(F)$, densamma som att erhålla ett meningsfyllt meddelande, d.v.s.

$$P(F) = \frac{2^{rN}}{2^{r'N}} = 2^{rN - r'N} = 2^{-DN} \qquad (2.16)$$

där $D = r' - r$, se uttrycket (2.9), är språkets redundans.

Väntevärdet för en oriktig dekryptering är lika med antalet falska nycklar multiplicerat med sannolikheten för en falsk nyckel, d.v.s.

$$E(F) = \left[2^{H(K)} - 1\right] P(F) = \left[2^{H(K)} - 1\right] 2^{-DN} \approx 2^{H(K) - DN} \qquad (2.17)$$

och eftersom $E(F)$ snabbt minskar med växande N, definieras det värde på N som gör att

$$\log_2 E(F) = H(K) - DN \approx 0 \qquad (2.18)$$

som det värde på N som gör att antalet oriktiga dekrypteringar är tillräckligt litet för att kunna knäcka koden.

Med detta värde på N definieras entydighetsdistansen enligt:

$$N = \frac{H(K)}{D} = \frac{H(K)}{\frac{D}{r'} \cdot r'} = \frac{H(K)}{\frac{D}{r'} \cdot \log_2 L} \qquad (2.19)$$

där $H(K)$ är nyckelns entropi, D/r' är språkets relativa entropi och L är antalet bokstäver eller tecken i språkets alfabet.

Av uttrycket (2.16) framgår att $H(K) \gg DN$ medför att det finns ett stort antal meningslösa meddelanden (som ju lätt kan sorteras bort) och därför en ökad möjlighet för forcören att lista ut vilket meddelande som är det riktiga.

Ur matematisk synpunkt skulle DN kunna sägas motsvara antalet ekvationer det finns för att bestämma nyckeln medan $H(K)$ motsvarar antalet okända. Endast om antalet ekvationer är mindre än antalet okända (bitar i K) är systemet obrytbart. För fallet att antalet ekvationer är lika med eller större än antalet okända är systemet inte obrytbart men det kan ändå vara beräkningsmässigt säkert, d.v.s. praktiskt olösbart med känd teknik. Framförallt är det antalet meningslösa meddelanden som möjliggör ett knäckande av chiffret – ju fler dessa är desto färre är de möjliga meningsfulla. För att minska mängden möjliga meningslösa meddelanden måste redundansen D minskas. Detta kan åstadkommas med ickeförstörande datakomprimering eller källkodning[12].

Exempel 2.6

Antalet nycklar för engelska som har 26 bokstäver är $26! \approx 4 \cdot 10^{26}$ eller generellt: Antalet nycklar för ett monoalfabetiskt substitutionschiffer med ett alfabet om s symboler är $s!$. Om alla nycklar är lika sannolika och den relativa redundansen i engelska förutsätts vara (se texten efter uttrycket (2.9)) D/r′ = = 0,68 kan vi beräkna entydighetsdistansen:

$$N \approx \frac{\log_2 26!}{0{,}68 \log_2 26} = \frac{\lg 26!}{0{,}68 \lg 26} \approx 28.$$

Detta resultat är i god överensstämmelse med empiriska data, som visar att en skicklig kryptoanalytiker kan få fram klartexten från en chiffertext med så lite som 25 tecken

◊

2.8.5 Sammanfattning: entydighetsdistansen

Entydighetsdistansen är ett mått på hur säkert ett kryptosystem är mot *forcering med bara chiffertext* och är en funktion av medelantalet falska nycklar, som för en given chiffertext kan dekrypteras till en meningsfylld klartext. Eftersom antalet falska nycklar, som kan dekrypteras till en meningsfylld klartext är mycket mindre än antalet möjliga falska nycklar, kommer de flesta av dessa nycklar att dekryptera en given chiffertext till meningslös klartext. Den relativa mängden meningslösa klartexter avspeglas i språkets relativa redundans, *D/r´*, och ett sätt att öka säkerheten i systemet är att öka entydighetsdistansen genom att minska den relativa redundansen. Minskning av den relativa redundansen görs genom att införa datakompression eller källkodning och genom att idealt minska den relativa redundansen till noll skulle entydighetsdistansen $N \to \infty$, d.v.s. det skulle teoretiskt behövas oändligt mycket chiffertext för

12 se Frank: ”*Telekommunikation avsnitt* 9.3”, Studentlitteratur.2004.

att kunna knäcka chiffret. Vi skulle alltså få ett chiffer med ideal sekretess. I verkligheten går det inte att minska den relativa redundansen till noll men datakompression kan ändå förlänga entydighetsdistansen avsevärt. Av den anledningen anses datakompression vara en mycket användbar metod för att höja säkerheten i kryptosystem.

Chiffertexter som är längre än entydighetsdistansen kan förutsättas ha endast en meningsfull dekryptering. Chiffertexter som är kortare än entydighetsdistansen kan ha flera möjliga dekrypteringar. Entydighetsdistansen är ett mått på hur mycket chiffertext som krävs för att det endast skall finnas en enda dekrypteringslösning.

2.9 Praktisk sekretess, blockchiffer

Om forcören har tillgång till en chiffertext som är längre än entydighetsdistansen kan kryptosystemet knäckas genom att pröva alla möjliga nycklar. Ett sådant förfaringssätt är emellertid opraktiskt i alla fall utom då nyckeln är mycket kort. Ta som exempel en nyckel som utgörs av en permutation av det svenska alfabetet, som innehåller 28 bokstäver. Det finns då $28! \approx 3 \cdot 10^{29}$ möjligheter. Vid en fullständig genomsökning av alla dessa möjligheter kan man förvänta sig att hitta rätt nyckel efter att ha undersökt ungefär hälften av alla. Med en datorsöktid på 1 μs per nyckel ger det en total söktid på drygt $4{,}8 \cdot 10^{15}$ år. Slutsatsen blir att forcören måste använda sig av betydligt mer sofistikerade metoder, t.ex. statistisk analys, för att hoppas på framgång.

2.9.1 Oordning och spridning

Som ovan nämnts är tillämpning av statistisk analys en metod som används av forcören. I själva verket kan en analys av de individuella tecknens och teckenkombinationernas relativa frekvenser användas för att knäcka många chiffer – se Figur 2.4 och Figur 2.5. För att försvåra forcörens analys har Shannon föreslagit två krypteringstranformationer som han kallar *oordning* och *spridning*.

Med oordning införs substitutioner så att förhållandet mellan nyckel och chiffer blir så komplicerat som möjligt. Avsikten är att försvåra en statistisk analys som skulle kunna reducera nyckelsökandet till en viss delmängd av nycklarna. Oordning säkerställer att praktiskt taget hela nyckeln måste tas i anspråk för att dekryptera även de kortaste chiffertexter.

Spridning görs genom transformationer som jämnar ut statistiska skillnader mellan olika bokstäver och bokstavskombinationer. Spridning med t.ex. det svenska (28-bokstavs-)alfabetet kan göras genom att en text bestående av en sekvens bokstäver, M_0, M_1, …, transformeras till en annan sekvens, Z_0, Z_1, …, enligt:

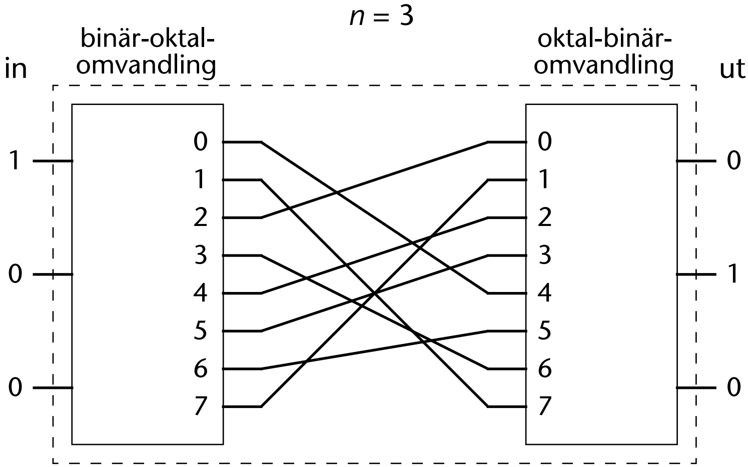
$$Z_n = \sum_{i=0}^{s-1} M_{n+i} \quad \text{mod } 28 \tag{2.20}$$

där varje bokstav i sekvensen Z_0, Z_1, … anges som ett heltal modulo-28 medan s utgörs av ett valt heltal. Genom denna spridningsmetod får den nya sekvensen Z_0, Z_1, … samma redundans som den ursprungliga M_0, M_1, … medan de enskilda bokstävernas frekvens i Z_0, Z_1, … blirmer likformigt fördelad än motsvarande i M_0, M_1, …. Effekten av spridningstransformationen blir att forcören måste skaffa en betydligt längre chiffertext än annars för att med statistiska metoder kunna utvinna användbar information om nyckeln.

2.9.2 Substitution

Den typ av substitution som krävs för att åstadkomma oordning i Shannons mening kräver mer komplexa substitutioner än de som åstadkoms med de enklare substitutionschiffren typ Caesar och Trithemius – se avsnitten 2.3.2 och 2.3.4.

I Figur 2.9 visas en mer komplex substitution utförd med en *olinjär transformation* i en substitutionsbox.



Figur 2.9 *Substitutionsbox eller S-box.*

in	000	001	010	011	100	101	110	111
ut	100	111	000	110	010	011	101	001

Tabell 2.2 In/ut-relationer för substitutionsboxen i Figur 2.9.

Bokstäverna i klartextmeddelandet omvandlas först till binära symboler som därefter binäroktalomvandlas (100 → 4).

De bildade oktala talen permuteras på ett olinjärt sätt och oktalbinäromvandlas därefter.

Antalet permutationer kan visas vara $(2^n)!$ och om n är t.ex. 128 blir $2^{128} \approx 3{,}4 \cdot 10^{38}$ och $(2^{128})!$ blir därför ett enormt stort tal. För $n = 128$ är substitutionen därför mycket komplex (oordning). Tyvärr är implementering av en sådan S-box praktiskt ogenomförbar eftersom den skulle kräva 2^{128} förbindningar.

Att substitutionsboxen enligt Figur 2.9 åstadkommer en olinjär transformation framgår om vi testar på två binära tal a = 001 och b = 010. Transformationen betecknas T och

$T(a) + T(b) = T(001) + T(010) = 111 + 000 = 111$

medan

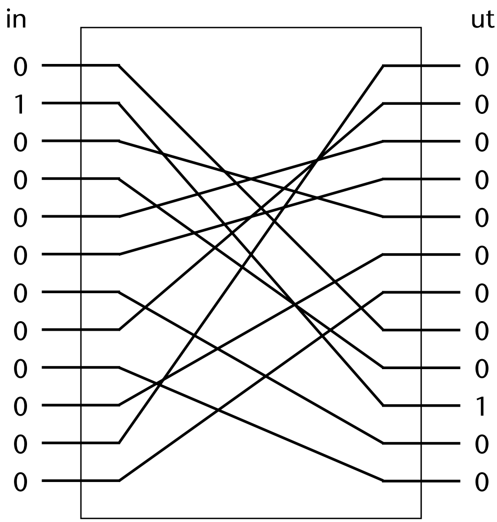
$T(a + b) = T(001 + 010) = T(011) = 110$

Alltså är $T(a) + T(b) \neq T(a + b)$ och substitutionsboxen är olinjär.

2.9.3 Permutation

Vid permutation sker endast en omkastning av bokstäverna i klartextmeddelandet (som kort blandas i en kortlek) så kan t.ex. klartextmeddelandet ”agronom” efter permutation ha blivit chiffertexten ”ngormao”. Figur 2.10 visar ett exempel på binär permutation (linjär operation).

Permutationsboxen bör inte användas enbart eftersom det går lätt att avslöja permutationsmönstret genom att sända en sekvens med enda etta och resten nollor (forcering med känd klartext) till P-boxens ingångar. Det blir med ett sådant förfarande lätt att bestämma en av P-boxens interna kopplingar.



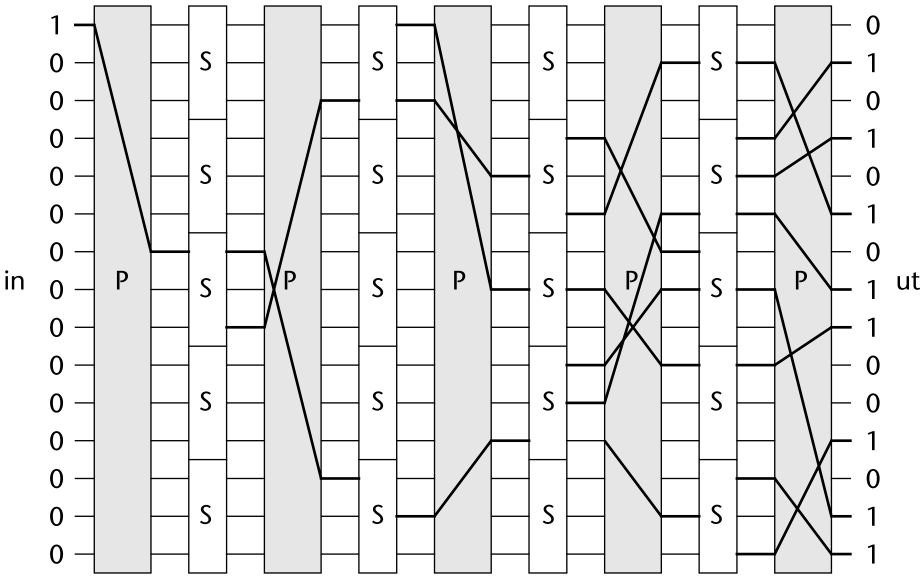
Figur 2.10 Permutationsbox eller P-box.

Flera sådana klartextforceringar med olika positioner för den ensamma ettan avslöjar snabbt P-boxens alla kopplingar.

2.9.4 Produktchiffersystem

För kryptering är S-boxtransformation eller P-boxtransformation var för sig otillräcklig. Shannon har föreslagit ett *produktchiffersystem*[13] som består av en kombination av dessa transformationer och som är betydligt kraftfullare än enbart en av transformationerna. Metoden har använts av IBM i deras LUCIFER-system[14].

Ett exempel på ett produktchiffersystem visas i Figur 2.11.



Figur 2.11 Produktchiffersystem.

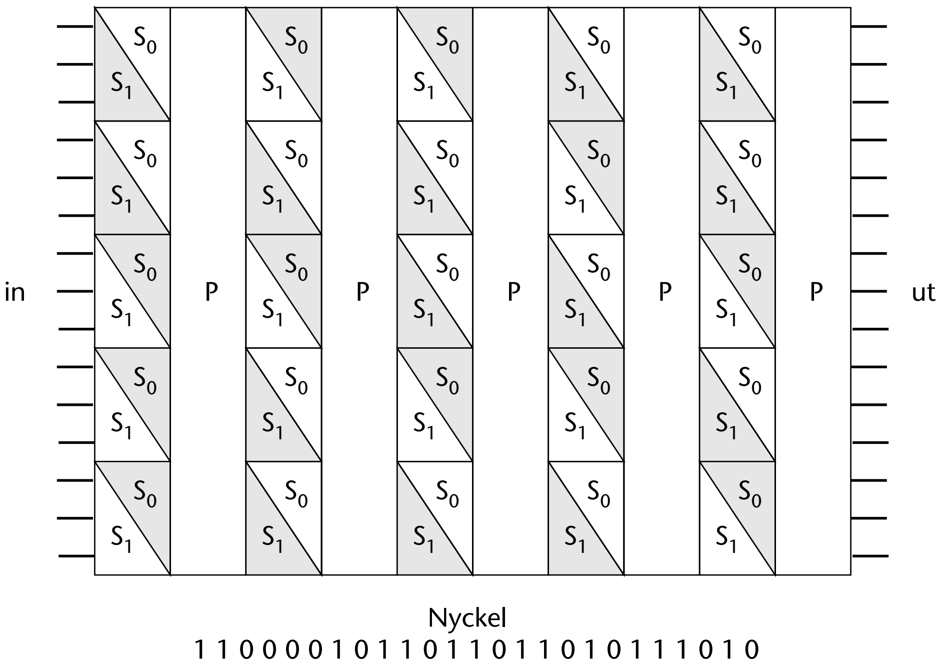
13 Shannon, C.E., ”Communication Theory of Secrecy Systems”, *Bell Syst. Tech. J.*, vol. 28, Oct. 1949, pp. 656-715.

14 Smith, J.L., ”The Design of Lucifer, a Cryptographic Device for Data Communications”, *IBM Research Report* RC-3326, 1971.

Dekryptering utförs genom att köra det krypterade ordet baklänges med användande av inversa S-boxar. Ett sådant system är dock svårt att implementera eftersom alla S-boxar är olika – en slumpgenererad nyckel kan inte användas – systemet är inte lämpat för upprepad användning av likadana kretsar.

Lucifersystemet kringgår dessa svårigheter genom att använda två olika sorters S-boxar, S_1 och S_0, med transformationer som tillåts vara offentliga. Ingångsdata (klartext) transformeras av en följd S- och P-boxar där S-boxarna, S_1 eller S_0, väljs med en nyckel – se Figur 2.12.

Strukturen hos produktchiffret i Figur 2.12 är typisk för många av dagens blockchiffer. Meddelandena delas upp i block, om vardera n bitar, som vart och ett krypteras med samma nyckel.



Figur 2.12 Gråa boxar motsvarar nyckelsymbolerna.

Varje n-bits block kan anta 2^n olika binära kombinationer, vilket möjliggör $(2^n)!$ olika substitutionsmönster. Ett exempel på detta följer i nästa avsnitt.

2.9.5 DES (the Data Encryption Standard)

LUCIFER-systemet var en föregångare till DES[15] (the Data Encryption Standard), det mest använda kryptosystemet i världen. DES kan uppfattas som ett blockkrypteringssytem med 2^{64} symboler. För detaljer i DES se avsnitt 2.17.1.

2.9.5.1 Analys och åsikter om DES

Redan när DES föreslogs som standard utsattes den för en hel del kritik. En invändning berörde S-boxarna. Eftersom S-boxarna är de enda olinjära komponenterna utgör de vitala delar för hela krypteringens säkerhet. När DES föreslogs misstänkte många att S-boxarna innehöll dolda kryphål som gjorde att the National Security Agency (NSA) kunde dekryptera meddelanden medan man falskeligen hävdade att DES var ”säkert”. Sådana spekulationer är naturligtvis omöjliga att vederlägga och det har aldrig framkommit några bevis för att kryphål faktiskt existerar.

Däremot har det nu slutligen avslöjats att S-boxarna i själva verket konstruerades för att förhindra vissa försök till forcering. När Biham och Shamir[16] uppfann differentialforceringsmetoden[17] erkändes

15 Först publicerad i *”the Federal Register”* 17 Mars, 1975. Antagen som standard för ”unclassified” tillämpningar 15 Januari 1977 i the *Federal Information Processing Standards (FIPS)* Publication 46. Ursprungligen bedömdes DES kunna användas som standard i 10-15 år, men visade sig vara betydligt mer hållbar. DES har genomgått översyn vart femte år och dess sista uppdatering skedde i Januari 1999 då den redan ersatts av AES (the Advanced Encryption Standard).

16 Eli Biham, Adi Shamir, ”Differential Cryptanalysis of DES-like Cryptosystems. Advances in Cryptology”, CRYPTO ’90, *Springer-Verlag*. 2-21.

det i några opublicerade kravspecifikationer gällande S-boxarna att de skulle konstrueras så att de omöjliggjorde differentialforcering av DES.

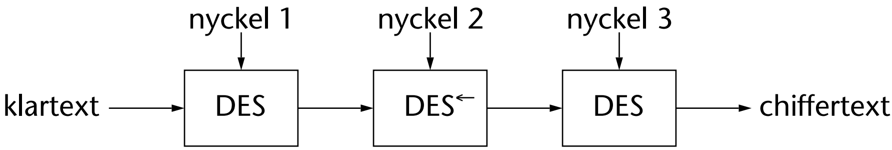
Differentialforcering var känt av IBM-forskarna när DES utvecklades men hemlighölls i nästan tjugo år tills Biham och Shamir återupptäckte metoden.

Den allvarligaste kritiken av DES hänför sig till nyckelstorleken 2^{56} som är för liten för att vara säker. Det ursprungliga förslaget var en 64-bits nyckel men detta reducerades senare till 56. IBM gjorde gällande att anledningen till reduktionen var att det var nödvändigt att inkludera 8 checkbitar, vilket medförde att 64-bitsminnet bara kunde innehålla en 56-bits nyckel. Med dagens datorer är det möjligt att söka igenom alla 2^{56} möjliga nycklar på relativt kort tid. Redan 1998 byggde the Electronic Frontier Foundation en ”DES Cracker” för $250.000 med 1536 chips som kunde söka efter 88.000.000.000 nycklar per sekund. I juli 1998 hittade den en DES-nyckel på 56 timmar.

Ett sätt att göra DES säkrare infördes 1999 kallat Triple DES (utvecklat av Walter Tuchman vid IBM). Det består av tre DES-krypteringssystem i rad med mittsystemet vänt. D.v.s. man har i tur och ordning: DES, $DES^{\leftarrow}$ och DES där $DES^{\leftarrow}$ definieras som dekryptering – se Figur 2.13. Det finns många sätt att använda DES tre gånger men alla sätt definieras inte som Triple DES.

---

17 Differentialforcering utförs vanligen med fritt vald klartext, vilket innebär att forcören måste förfoga över krypterad chiffertext för några delar av sin valda klartext. Det finns emellertid utvidgningar som tillåter forcering med känd klartext eller till och med forcering med bara chiffertext. Den grundläggande metoden använder sig av par av klartexter med en konstant *differens*. Differensen kan definieras på många olika sätt men det vanligaste sättet att bilda en differens görs med modulo-2 addition. Forcören jämför differenserna mellan motsvarande chiffertexter i hopp om att upptäcka statistiska fördelningsmönster. Vanligen förväntar man sig att hitta en speciell differens som är särskilt frekvent och att därigenom kunna skilja chiffret från slumpmässiga variationer.
Sedan differentialforcering kommit till allmän kännedom har det blivit en källa till oro för många chifferkonstruktörer.



Figur 2.13 Triple DES.

Triple DES har nyckellängden 168 bitar (tre 56-bits nycklar) men vid forcering blir den effektiva nyckellängden 112 bitar. En variant har nyckel 1 = nyckel 3 vilket innebär att nyckellängden reduceras till 112 bitar. En sådan mode är dock mindre motståndskraftig mot vissa typer av forceringsförsök.

Om nyckel 1 = nyckel 2 eller nyckel 2 = nyckel 3 reduceras Triple DES till vanlig DES. Därigenom kan Triple DES göras bakåtkompatibelt med äldre system som använder DES.

Förutom differentialforcering har man använt sig av linjär forcering[18] och the Improved Davies´ attack[19].

En ny och modernare krypteringsmetod som löser många av problemen med DES är AES (the Advanced Encryption Standard) som presenteras i nästa avsnitt.

2.9.6 AES (the Advanced Encryption Standard)

2 januari 1997 började NIST (the National Institute of Standards) en process för att ersätta DES. Standarden skulle få namnet AES (the Advanced Encryption Standard). Kravspecifikationerna angav blocklängden till 128 bitar och nyckellängder till 128, 192 och 256

---

18 Metoden tillskrivs Mitsuri Matsui som först tillämpade metoden 1992.

19 E. Biham, A. Biryukov, ”An Improvement of Davies´ Attack on DES”, *Journal of Cryptology*, vol. 10, no. 3, pp. 195-206, 1997.

bitar. Dessutom krävdes att AES skulle vara globalt tillgänglig och royaltyfri.

Efter en lång urvalsprocess valde man slutligen den 2 oktober 2000 de två belgiska krypteringsexperternas, John Daemen och Vincent Rijmen, förslag *Rijndael*, namnet bildat genom en hopslagning av uppfinnarnas efternamn. Rijndael antogs som AES standard 26 november 2001.

De tre viktigaste kriterierna vid valet av AES-kandidat var:

- säkerhet
- kostnad
- algoritm- och implementeringskarakteristikor

Säkerhet hos den föreslagna algoritmen var helt avgörande för om den överhuvudtaget skulle beaktas vid den fortsatta utvärderingen. Med kostnad avses beräkningsmässig effektivitet (hastighet och minnesbehov) vid olika implementeringar inklusive mjuk- och hårdvara samt smartcards. Algoritm- och implementeringskarakteristikor innefattar bl.a. flexibilitet och algoritmenkelhet.

I slutomgången återstod fem finalister som alla uppfattades som säkra. Valet av Rijndael motiverades av att dess kombination av säkerhet, utförande, effektivitet, implementerbarhet och flexibilitet ansågs överlägsen övriga finalister.

För detaljer i AES (Rijndael) se avsnitt 2.17.2.

2.9.6.1 Analys av AES

AES anses säkert mot alla kända forceringsförsök (2005).Olika aspekter av dess konstruktion inbegriper speciella egenskaper som skyddar den mot vissa typer av forcering. Exempel på detta utgör inverteringsoperationen i en ändlig talkropp som man använt vid S-boxkonstruktionen. Detta ger linjär approximation och differensfördelningstabeller i vilka infördelningen är nästan likformig. Därigenom omöjliggörs differential- och linjär forcering. Dessutom gör den linjära transformationen MixColumns att det blir omöjligt för differential- och linjärforceringar som innehåller ”få” aktiva S-boxar[20].

2.10 Praktisk sekretess, strömchiffer

Strömchiffer kan konstrueras för att bli mycket snabba, i själva verket mycket snabbare än något blockchiffer. Medan blockchiffer arbetar på stora datablock behandlar strömchiffer mindre delar i varje krypteringsmoment, vanligen bara en bit eller en byte i taget. Till skillnad från blockchiffer som för en och samma klartext alltid lämnar samma chiffertext varierar chiffertexten vid strömkryptering beroende på när, inom en viss tidscykel, klartexten ansluts till krypteringsprocessen.

Ett strömchiffer genererar en *nyckelström* och krypteringen utförs genom att klartexten adderas modulo-2. Om nyckelströmmen genereras oberoende av klartexten och chiffertexten får man ett synkront strömchiffer. Om krypteringen däremot är beroende av klartexten och chiffertexten får man ett *självsynkront strömchiffer*. De flesta strömchiffer är konstruerade som synkrona strömchiffer.

Strömchiffer kan uppfattas som en approximation av beteendet hos det enda, teoretiskt sett, oforcerbara kryptosystemet, the one time pad[21]. Stor möda har nedlagts på att generera nyckelströmmar som verkar vara slumpartade men ändå lätt kan skapas för dekryptering eftersom de kan genereras av algoritmer. Sådana strömchiffersystem använder sig av pseudoslumpsekvenser (PN-sekvenser = Pseudo-Noise sequences) med samma statistiska egenskaper som binomialfördelningen (slantsingling). Pseudoslumpsekvenser används ofta eftersom kryptering och dekryptering enkelt kan utföras med återkopplade skiftegister.

En viktig skillnad mellan en pseudoslumpsekvens och en äkta slumpsekvens är att pseudoslumpsekvensen genereras av en algo-

---

20 Konstruktörerna kallar denna egenskap *wide trail strategy*.

21 Se fotnot 6.

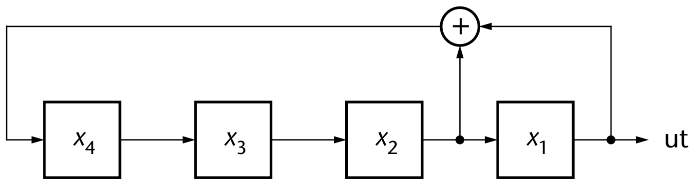
ritm och kännedom om algoritmen medför att man känner hela sekvensen vilket gör att chiffret enkelt kan forceras.

2.10.1 Nyckelgenerering med linjärt återkopplat skiftregister

Vid strömkrypteringsteknik används ofta skiftregister för att generera nyckel-PN-sekvenser. Ett återkopplat skiftregister kan nämligen användas för att åstadkomma en PN-sekvens.

Om den numeriska operationen i återkopplingsslingan i ett återkopplat skiftregister är linjär har vi ett linjärt återkopplat skiftregister. Figur 2.14 illustrerar ett exempel med 4 vippor som är kopplade så att de vid klockning genomlöper $2^4 - 1$ olika tillstånd. Allmänt kan ett återkopplat skiftregister med n vippor maximalt genomlöpa $2^n - 1$ olika tillstånd[22].

Om initialtillståndet i det linjärt återkopplat skiftregistret (x_4,x_3,x_2,x_1) enligt Figur 2.14 är 1000 blir de följande tillstånden 0100, 0010, 1001, 1100 o.s.v. Utsekvensen utgörs av bitarna som skiftas ut från högra delen, x_1, i registret d.v.s. 111101011001000 där biten längst till höger är den som kommer först medan den längst till vänster är den som kommer sist. Efter det att alla tillstånd genomlöpts och ovanstående sekvens genererats upprepas sekvensen – utsekvensen är periodisk. Egenskapen att kunna generera en (pseudo)slumpsekvens framträder genom att sekvensens autokorrelationsfunktion är liten (deltakorrelerad ($\delta(n) \approx 0$)).



Figur 2.14 Linjärt återkopplat skiftregister.

22 Av den anledningen kallas sådana sekvenser maximallängdssekvenser eller bara *m*-sekvenser.

2.10.2 Forcering av strömchiffer med linjärt återkopplat skiftregister

För att forcera ett strömchiffer skapat med ett återkopplat n-vippors skiftregister behöver en forcör endast $2n$ bitar klartext och motsvarande chiffertext för att bestämma skiftregistrets återkopplingar, initialtillstånd och hela nyckelsekvens och eftersom $2n$ bitar är ett mycket mindre tal än hela sekvenslängden $2^n - 1$ är känsligheten för forcering med känd klartext mycket stor.

Exempel 2.7

En chiffertext är bildad med hjälp av det återkopplade skiftregistret i Figur 2.14. En forcör försöker knäcka detta chiffer utan att ha tillgång till det återkopplade skiftregistrets interna kopplingar. Han har dock lyckats få tillgång till $2n = 8$ bitar chiffertext och motsvarande klartext enligt nedan:

klartext: 01010101

chiffertext: 00001100

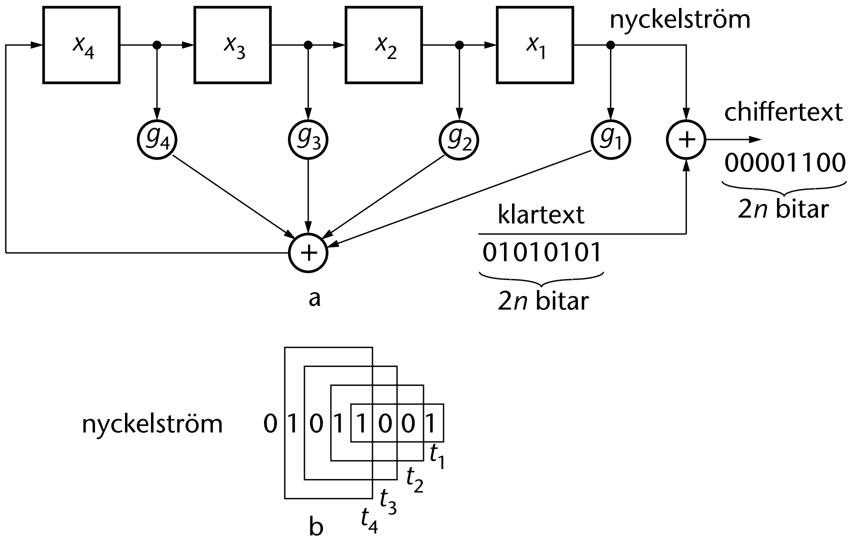
I de ovan angivna klar- och chiffertexterna anländer de högra bitarna först och de vänstra sist.

Forcören adderar (modulo-2) nu de två sekvenserna för att få nyckelströmmen: 01011001 – se Figur 2.15 a. Nyckelströmsekvensen visar innehållet i det återkopplade skiftregistret vid olika tidpunkter. De fyra högra bitarna, inom rektangeln längst till höger i Figur 2.15 b, visar registerinnehållet vid tidpunkten t_1, därefter följer registerinnehållet vid tidpunkten t_2 inom nästa rektangel o.s.v. Eftersom återkopplingen är linjär kan vi ställa upp följande samband:

$$g_4x_4 + g_3x_3 + g_2x_2 + g_1x_1 = x_5 \tag{2.21}$$

där x_5 är den siffra som återkopplas till ingången och g_i (= 1 eller 0) definierar den i:te återkopplingen.

Med hjälp av Figur 2.15 b kan vi ställa upp följande samband för de fyra tidpunkterna:



Figur 2.15 Forcörens strategi för att bestämma skiftregistrets återkopplingsvikter g_4, g_3, g_2 och g_1.

$$\left.\begin{array}{ll} ① & g_4 \cdot 1 + g_3 \cdot 0 + g_2 \cdot 0 + g_1 \cdot 1 = 1 \\ ② & g_4 \cdot 1 + g_3 \cdot 1 + g_2 \cdot 0 + g_1 \cdot 0 = 0 \\ ③ & g_4 \cdot 0 + g_3 \cdot 1 + g_2 \cdot 1 + g_1 \cdot 0 = 1 \\ ④ & g_4 \cdot 1 + g_3 \cdot 0 + g_2 \cdot 1 + g_1 \cdot 1 = 0 \end{array}\right\} \tag{2.22}$$

Med lösningen $g_1 = 1$, $g_2 = 1$, $g_3 = 0$ och $g_4 = 0$. Denna lösning motsvarar kopplingen i Figur 2.14. Forcören har alltså räknat ut hur skiftregistret är återkopplat och också registrets starttillstånd vid tidpunkten t_1. Han kan därmed beräkna hela nyckelsekvensen.

◊

För att generalisera Exempel 2.7 att gälla för ett godtyckligt linjärt återkopplat skiftregister med n vippor skriver vi om uttrycket (2.21) på följande sätt:

$$x_{n+1} = \sum_{i=1}^{n} g_i x_i \tag{2.23}$$

som på matrisform blir:

$$\boldsymbol{x} = \boldsymbol{g}\boldsymbol{X} \tag{2.24}$$

där

$$\boldsymbol{x} = \begin{bmatrix} x_{n+1} & x_{n+2} & \dots & x_{2n} \end{bmatrix} \quad \boldsymbol{g} = \begin{bmatrix} g_1 & g_2 & \dots & g_n \end{bmatrix} \tag{2.25}$$

och

$$\boldsymbol{X} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ x_2 & x_3 & \dots & x_{n+1} \\ \vdots & \vdots & & \vdots \\ x_n & x_{n+1} & \dots & x_{2n-1} \end{bmatrix} \tag{2.26}$$

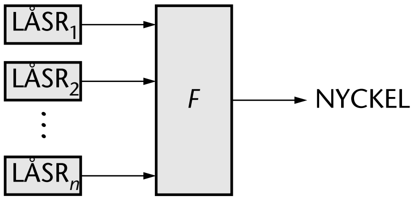
Eftersom kolumnerna i $\boldsymbol{X}$ är linjärt oberoende[23] är matrisen icke-singulär och därmed inverterbar och

$$\boldsymbol{g} = \boldsymbol{x}\boldsymbol{X}^{-1} \tag{2.27}$$

Frågan är nu hur pass komplicerad matrisinversionen är. Med n vippor i det återkopplade skiftregistret krävs det i storleksordningen n^3 operationer, vilket med n = 100 skulle kräva c:a 1 s för en dator med 1 μs operationscykel.

Svagheten i den här krypteringsmetoden ligger i det linjärt återkopplade skiftregistret. Om man däremot använder ett olinjärt återkopplat skiftregister används försvåras forcörens uppgift avsevärt. Ett sätt att eliminera linjäriteten är att koppla flera parallella återkopplade skiftregister till en olinjär boolesk funktion för att skapa en kombinationsgenerator – se Figur 2.16. Egenskaperna hos den av kombinationsgeneratorn bildade kombinationsfunktionen är kritiska för att hindra t.ex. forcering som bygger på korrelationsegenskaper.

23 se t.ex. Beker, H. och Piper, F. ”Cipher Systems”, *John Wiley & Sons, Inc.*, New York, 1982.

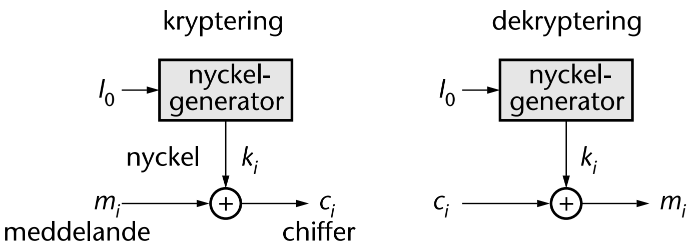


Figur 2.16 N parallella linjärt återkopplade skiftregister (LÅSR) använda för att skapa en olinjär kombinationsgenerator som genererar en olinjär kombinationsfunktion (F).

2.10.3 Synkrona och självsynkrona strömkrypteringssystem

Strömkrypteringssystem kan indelas i *synkrona* och *självsynkrona*. För de synkrona bildas nyckelströmmen oberoende av meddelandet. Detta gör det nödvändigt att återsynkronisera sändarens och mottagarens nyckelströmgeneratorer om någon symbol skulle råka försvinna under överföringen.

Vid start av ett synkront strömkrypteringssystem initieras nyckelgeneratorn av en startsekvens I_0 (som är känd) – se Figur 2.17. Chiffertexten bildas sedan genom att modulo-2-addera nyckelströmmen till meddelandeströmmen.

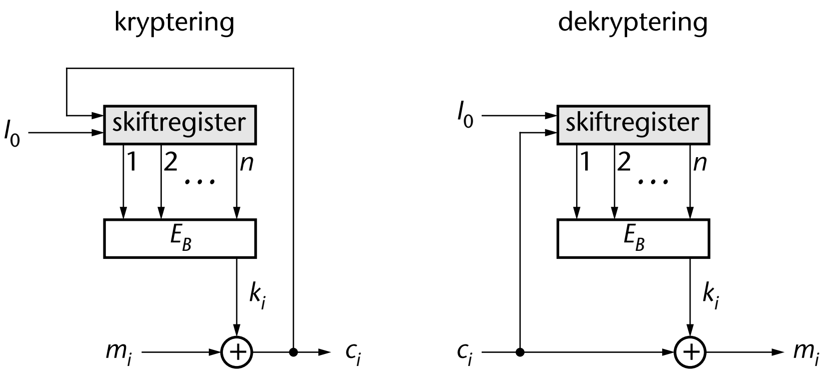


Figur 2.17 Principen för ett synkront strömkrypteringssystem.

Synkrona chiffersystem konstrueras vanligen för att åstadkomma oordning men inte spridning – se avsnitt 2.9.1. Eftersom de inte medför spridning bildas, om fel uppstår, inte heller någon felfortplantning.

I ett självsynkront strömkrypteringssystem bildas varje nyckelsymbol av ett bestämt antal symboler av den föregående chiffertexten. Denna återkopplingsmetod har också gett systemet det alternativa namnet chifferåterkoppling. Om antalet symboler som bildar nyckelsymbolen är n och en symbol försvinner under överföringen fortplantas felet n symboler framåt men systemet återsynkroniseras efter det att n riktiga chiffertextsymboler mottagits.

Figur 2.18 illustrerar en chifferåterkopplad nyckelgenerator. Varje chiffertextsymbol återkopplas till skiftregistrets ingång. Starten sätts igång av en känd insekvens I_0 och vid varje skift matas skiftregistrets utgångssignaler in till en blockkrypterare som är uppbyggd med en olinjär krypteringsalgoritm, E_B. Utgångsvärdet från blockkrypteraren blir nästa nyckelsymbol, k_{i+1}, som används för att modulo-2-adderas till nästa meddelandesymbol, m_{i+1}. Eftersom, efter några inledande skift, insekvensen till blockkrypteraren bara beror av chiffersymbolerna är systemet självsynkroniserande.



Figur 2.18 Chifferåterkoppling.

2.10.4 Strömchiffret RC4

RC4 är det mest använda mjukvarubaserade strömchiffret och används bl.a. i Secure Sockets Layer (SSL) för att skydda internettrafik och i Wired Equivalent Privacy (WEP) ett säkerhetsprotokoll för att skydda lokala trådlösa nätverk (WLAN, WireLess Local Area Networks). RC4 lever tyvärr inte upp till de högsta säkerhetskraven på moderna krypteringssystem och en del användningssätt av RC4 leder till mycket osäkra system (inklusive WEP). Av den anledningen rekommenderas inte RC4 för implementering i nya system. Emellertid är en del system baserade på RC4 tillräckligt säkra för praktisk användning i existerande system.

RC4 konstruerades av Ronald L. Rivest[24] 1987, då vid företaget RSA (RSA efter Rivest, Shamir och Adleman). Från början var algoritmen för RC4 hemlig men i september 1994 avslöjades den av någon okänd och så småningom kom den ut på Internet.

Algoritmen för RC4 är mycket enkelt uppbyggd – se Figur 2.21. Längden på nyckeln kan varieras och nyckelströmmen genereras byte för byte. RC4 använder sig av två vektorer eller register S och T med 256 positioner och en byte i varje position.

2.10.4.1 Nyckelinitiering

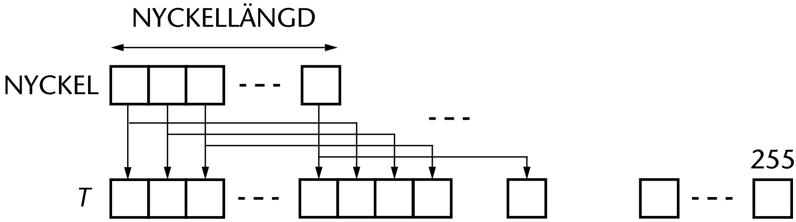
De båda vektorerna tilldelas för alla i mellan 0 och 255 värden enligt följande:

for i = 0 to 255 do

$S[i] = i;$

Detta, identitetspermutationen, innebär att varje byte får samma värde som sin plats i registret enligt: $S[0] = 0$, $S[1] = 1$, …, $S[255] =$ $= 255$.

24 Ronald L. Rivest, Professor of Electrical Engineering and Computer Science vid Massachusetts Institute of Technology (MIT).



Figur 2.19 NYCKEL kopieras till det temporära registret T.

$T[i]$ = NYCKEL [i modulo-NYCKELLÄNGD];

Eftersom nyckellängden (här benämnd NYCKELLÄNGD) vanligen är 5 till 16 bytes kopieras nyckeln (här benämnd NYCKEL) upprepade gånger till det temporära registret T tills det är fullt.

2.10.4.2 Permutering av *S*

$j = 0$;

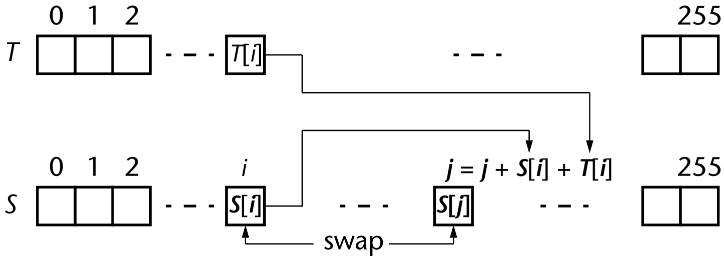
Börja med att sätta $j = 0$
Därefter sker en permutation för alla i mellan 0 och 255 enligt

$j = (j + S[i] + T[i])$ modulo-256;

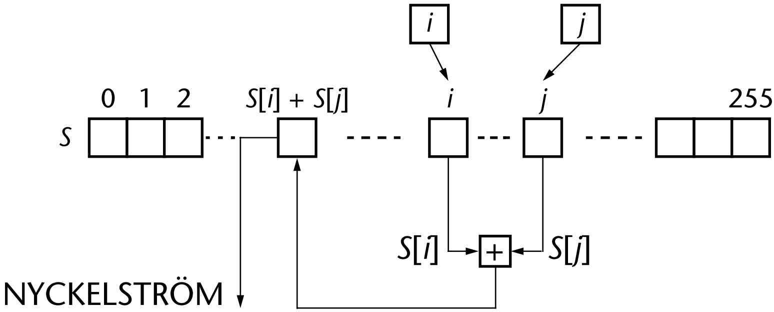
Swap ($S[i]$ och $S[j]$);

Värdena $S[i]$ och $S[j]$ byter plats med varandra enligt det temporära registret $T[i]$.

Sist sätts i och j till noll.



Figur 2.20 Permutering av S.



Figur 2.21 Lookup-steget i RC4. Utgångsbytes väljs genom att leta reda på (looking up) värdena S(i) och S(j) och addera dem modulo-256 och sedan hitta summan i S; S(S(i) + S(j)) används som en byte i nyckelströmmen. $\boxed{+}$ *betyder addition modulo-256.*

2.10.4.3 Nyckelströmsgenerering

För varje genererad nyckelbyte sker följande:

$i = (i + 1)$ modulo-256;

i ökas med 1 modulo-256.

$j = (j + 1)$ modulo-256;

j ökas med 1 modulo-256.

Swap ($S[i]$ och $S[j]$);

Värdena $S[i]$ och $S[j]$ byter plats med varandra.

NYCKELSTRÖM = $S[(S[i] + S[j])$ modulo-256];

output NYCKELSTRÖM;

2.10.4.4 Analys av RC4

Som förut nämnts klarar inte RC4 krav på ett säkert chiffer och rekommenderas inte för användning i nya tillämpningar. Nyckelströmmen som genereras av RC4 är något viktad till förmån för vissa bytesekvenser.

RC4 använder inte någon separat nonce[25] tillsammans med nyckeln. 2001 gjordes en ny och överraskande upptäckt av Fluhrer, Mantin och Shamir: Av alla möjliga RC4-nycklar är statistiken för ett antal av de första bytes av nyckelströmmen mycket oslumpmässiga, vilket läcker information om nyckeln. Ett sätt att komma runt detta problem är att inte ta med ”insvängningsförloppet” t.ex. genom att kasta de 1024 första bytes.

2.11 Hashfunktioner

Vid dataöverföring finns det många exempel på att kryptering används utan att chiffertexten måste dekrypteras. I själva verket är det ibland ett krav att chiffertexten inte kan dekrypteras. Ett exempel på detta är vid skydd av lösenord i datasystem. Här krävs att det går att verifiera att det avgivna lösenordet är korrekt utan att för den skull dekryptera det i databasen lagrade värdet. Det finns också många tillfällen när längre (hemliga) meddelanden behöver komprimeras till betydligt kortare bitsekvenser. Det är tyvärr då oundvikligt att mer än ett meddelande kan ge upphov till samma komprimerade bitsekvens, ett förhållande som automatiskt låter förstå att komprimeringsprocessen är irreversibel.

För att skapa dessa komprimerade bitsekvenser används *hashfunktioner* och beroende på tillämpning kan de innehålla en kryptografisk nyckel eller inte. I allmänhet fungerar en hashfunktion så att en godtyckligt lång symbol/bit-sekvens komprimeras till en sekvens med fix längd. Denna fixlängdssekvens kallas ett *digitalt fingeravtryck*, *digitalt sammandrag* eller bara *hash*. Som redan nämnts är det oundvikligt att två olika meddelandesekvenser kan ge upphov till samma digitala fingeravtryck. När detta sker säger man att en *kollision* uppstått. För att därför på ett unikt sätt kunna identifiera ett meddelande genom sitt digitala fingeravtryck måste hashfunktionen väljas så att sannolikheten för en kollision blir mycket liten.

25 nonce ≈ number once, ung. samma klartextsekvens ger aldrig samma chiffer mer än en gång.

För att den skall bli det måste antalet möjliga digitala fingeravtryck vara mycket stort. Det finns ingen formell definition som i sig innefattar alla de egenskaper som anses önskvärda för en kryptografisk hashfunktion (en hashfunktion innehållande en kryptografisk nyckel) men nedanstående egenskaper anses allmänt utgöra nödvändiga förutsättningar:

- *Preimage resistant*: Om det digitala fingeravtrycket h är givet skall det vara svårt att bestämma m där $h = \text{hash}(m)$[26].
- *Second preimage resistant*: Om meddelandet m_1 är givet skall det var svårt att hitta ett annat meddelande m_2 så att $\text{hash}(m_1) = \text{hash}(m_2)$.
- *Collision-resistant*: Det skall vara svårt att hitta två olika meddelanden m_1 och m_2 så att $\text{hash}(m_1) = \text{hash}(m_2)$.

Hashfunktioner används i många säkerhetstillämpningar som autenticering och meddelandeintegritet.

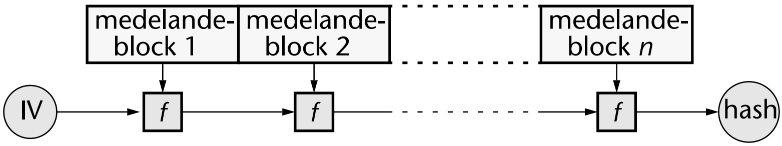
De två vanligaste hashfunktionerna är MD5 och SHA som båda bygger på Merkle-Damgårds hashfunktioner.

2.11.1 Merkle-Damgårds hashfunktioner

Merkle och Damgård har angivit en metod för att av ett meddelande med godtycklig längd bilda ett digitalt fingeravtryck med fix längd. Metoden går ut på att dela upp meddelandet i en serie lika långa block på vilka man i tur och ordning använder en kompressionsfunktion. Det sista blocket som processas innehåller också information om hela meddelandets längd, en uppgift som är avgörande för metodens säkerhet.

I Figur 2.22, där metoden illustreras, kallas kompressionsfunktionen f och den transformerar en insekvens av fix längd till en utsekvens, också med fix men kortare längd. Algoritmen startar med ett initialvärde som kommer från det cirkelformade blocket IV (InitieringsVektor).

26 hash(m) betyder hashfunktionen av m.



Figur 2.22 Merkle-Damgårds struktur.

För varje meddelandeblock och insekvens (från vänster till ett f-block) bildar kompressionsfunktionen f ett mellanresultat. Bitar som representerar hela meddelandets längd ansluts till meddelandet och får lämpligen utgöra en del av det sista meddelandeblocket, n. Utsekvensen från det sista f-blocket är hashvärdet för hela meddelandet.

Merkle och Damgård har visat att om kompressionsfunktionen är collision-resistant så är också den bildade hashfunktionen collision-resistant. Tyvärr har metoden en del mindre önskvärda egenskaper:

- forcering genom expansion av meddelandelängden (om en kollision har hittats så är det lättare att hitta fler) är alltid möjlig.
- forcering genom att hitta second preimage, d.v.s. givet m_1 finn m_2 så att hash(m_1) = hash(m_2) är alltid lättare än att söka igenom alla möjliga (brute force attack).

2.11.2 MD5, Message-Digest algorithm 5

MD5 skapades 1991 av Ronald Rivest för att ersätta dess föregångare, MD4. MD5 är en 128-bitars hashfunktion och används i många säkerhetstillämpningar men även för kontroll av filintegritet. Allvarliga brister som upptäcktes 2004 har dock gjort att användning av algoritmen i nya tillämpningar är tveksam.

2.11.2.1 MD5-algoritmens struktur

MD5 processar ett meddelande av godtycklig längd och lämnar ut en utsekvens, hashfunktionen, med den konstanta längden 128 bitar. Detta tillgår så att meddelandet först delas upp i 512-bits block och om det inte är jämnt delbart med 512 förlängs det på följande sätt: Först läggs en etta till i slutet av meddelandet och därefter kommer så många nollor att meddelandelängden blir 64 bitar kortare än en multipel av 512. Återstående bitar fylls ut med 64 bitar som representerar längden av det ursprungliga meddelandet. Meddelandet fylls alltid på med minst en etta så att om meddelandelängden blir en multipel av 512 minus de 64 bitarna representerande originalmeddelandelängden (d.v.s. längd modulo-512 = 448), läggs ett nytt 512-bits block till bestående av en etta följd av 447 nollor och avslutad med 64 bitar representerande meddelandets originallängd.
Huvudalgoritmen i MD5 arbetar på en 128-bits sekvens – se Figur 2.23 - indelat i fyra 32-bits ord kallade *A, B, C* och *D* som initieras med fixa konstanter. Huvudalgoritmen arbetar därefter i tur och ordning på varje 512-bits block. Varje block modifierar 128-bitssekvensen. Processandet av ett meddelandeblock består av fyra likadana rundor. Varje runda innehåller 16 likadana operationer bestående av en olinjär funktion *F* (*G, H* och *I*), modulo-addition och vänsterskift. Figur 2.23 visar en operation i en runda. De fyra olinjära funktionerna definieras enligt följande:

$$\begin{aligned} F(B,C,D) &= B \cdot C + \overline{B} \cdot D \\ G(B,C,D) &= B \cdot D + C \cdot \overline{D} \\ H(B,C,D) &= B \oplus C \oplus D \\ I(B,C,D) &= C \oplus (B + \overline{D}) \end{aligned} \tag{2.28}$$

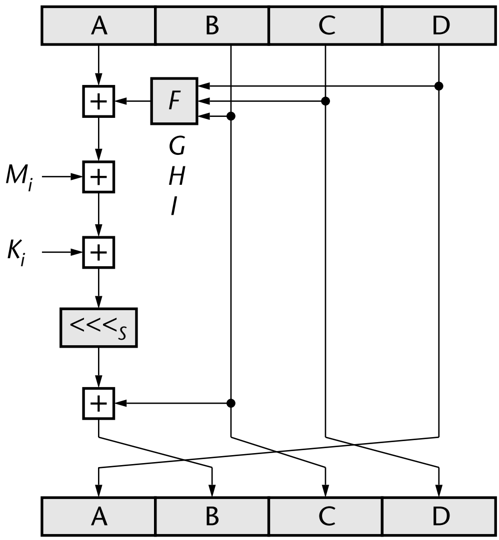
där

$\cdot$ motsvarar OCH

$+$ motsvarar ELLER

_ motsvarar NOT

$\oplus$ motsvarar modulo-2 addition.

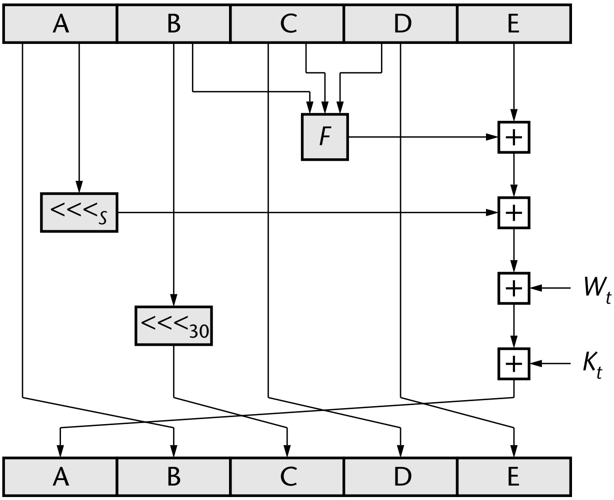


Figur 2.23 *Figuren illustrerar en MD5-operation. Algoritmen består av 64 sådana operationer uppdelade i fyra rundor om vardera 16 operationer. F, G, H och I är fyra olinjära funktioner av vilka en ny används i varje runda. M_i är ett 32-bits meddelandeblock medan K_i är en 32-bits konstant, olika vid varje operation. $\boxed{<<<_s}$ betyder s bitar vänsterskift, där s är olika för varje operation. $\boxed{+}$ betyder addition modulo-2^{32}.*

2.11.3 SHA, Secure Hash Algorithm

SHA-algoritmerna består av ett antal likartade kryptografiska hashfunktioner, av vilka den vanligaste, SHA-1, används i ett stort antal säkerhetstillämpningar och protokoll såsom TLS (Transport Layer Security), SSL (Secure Sockets Layer), PGP (Pretty Good Privacy), SSH (Secure SHell), S/MIME (Secure/Multipurpose Internet Mail Extensions) och IPsec (Internet Protocol security).

Den ursprungliga specifikationen för SHA-0 publicerades 1993 av US government standards agency NIST (National Institute of Standards and Technology).



Figur 2.24 En iteration i SHA-1. A, B, C, D och E är 32-bits ord i hashfunktionsordets tillstånd. F är en olinjär funktion som varierar. $\boxed{<<<_s}$ betyder s bitar vänsterskift, där s är olika för varje operation. K_t är en konstant och W_t är ett 32-bits ord erhållet av aktuellt 512-bits ingångsblock.

Både SHA-0 och SHA-1 producerar hashfunktioner med längden 160 bitar.

SHA-0 anses inte längre vara säker och det påstås att också SHA-1 har forcerats medan forcering av SHA-2 inte rapporterats (2005). Fyra varianter av SHA-2 publicerades 2001: SHA-224, SHA-256, SHA-384 och SHA-512 namngivna efter sina bitlängder.

2.12 Publika nyckelkryptosystem

1976 införde Diffie och Hellman[27] begreppet publika nyckelkryptosystem. Till skillnad mot konventionella kryptosystem, i vilka krypteringsalgoritmen kan avslöjas, p.g.a. att systemets säkerhet beror av endast *en* säkerhetsskyddad nyckel[28] (samma nyckel för kryptering och dekryptering), använder sig publika kryptosystem av *två olika nycklar*, en för kryptering och en för dekryptering.

Krypteringsnyckeln och krypteringsalgoritmen kan ges allmän kännedom (därav namnet *publika nyckelkryptosystem*) via exempelvis allmänt åtkomliga databaser utan att systemets säkerhet minskas. Det är bara dekrypteringsnyckeln som måste hållas hemlig.

Ett publikt nyckelkryptosystem har följande viktiga egenskaper:

1. För en klartext, M, eller chiffertext, C, som definieras av en nyckel K, gäller det för varje K och M, att om $C = = E_K(M)$ så är $M = D_K(C) = D_K[E_K(M)]$. Man säger att krypteringsalgoritmen och dekrypteringsalgoritmen utgör inverterbara transformationer.

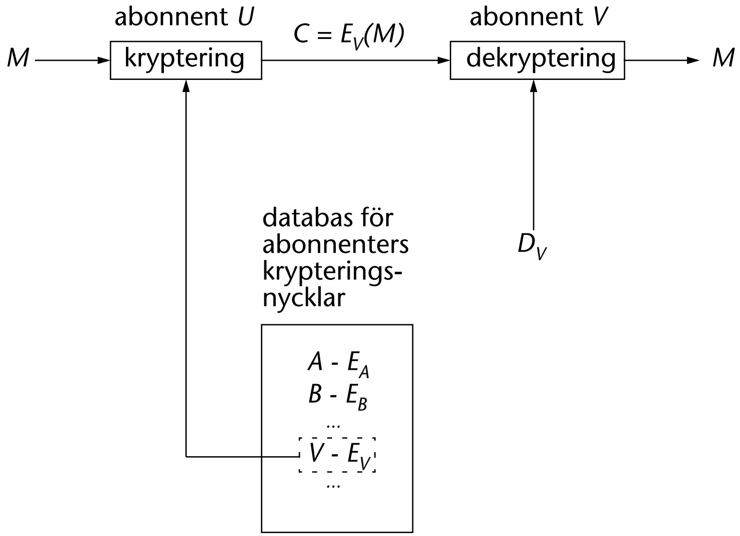
2. Det är lätt att beräkna E_K och D_K för vilket K som helst.

3. Det är praktiskt svårt att räkna ut D_K om man bara känner E_K.

Figur 2.25 visar schematiskt ett publikt nyckelkryptosystem. Abonnent U sänder meddelandet M till abonnent V genom att använda abonnent V:s krypteringsnyckel, V, som han hämtar i databasen. Med användning av krypteringsalgoritmen E_V får han chiffertexten $C = E_V(M)$, som sänds över den publika kanalen. Abonnent V är den enda som kan dekryptera chiffertexten C med dekrypteringsalgoritmen D_V för att erhålla $M = D_V(E_V(M)) = D_V(C)$.

27 Diffie, W. och Hellman, M. E., ”New Directions in Cryptography”, *IEEE, Trans. Inf. Theory,* vol. IT22, Nov. 1976, pp. 644-654.

28 Eng. safeguarded key.



Figur 2.25 Publikt nyckelkryptosystem.

Publika nyckelkryptosystem baseras på *trapdoor one-way functions*. Begreppet *one-way function* definieras som en funktion, $f(x)$, vars värde, givet x, är lätt att beräkna medan det är betydligt besvärligare att beräkna inversen, d.v.s. x om värdet på $f(x)$ är givet. Som exempel kan vi ta ett någorlunda krångligt polynom: $f(x) = x^7 + 13x^5 - 143x^3 + 109x + 119$. En *trapdoor one-way function* har samma egenskaper som en one-way function men är så konstruerad att man med speciell information lätt kan beräkna inversen. En viktig algoritm för att konstruera trapdoor one-way functions är RSA-algoritmen som behandlas i nästa avsnitt.

2.12.1 RSA-algoritmen

Med Diffies och Hellmans pionjärarbete (se Fotnot 27) introducerades ett nytt grepp på kryptografi som ledde till att kryptologer utmanades att utveckla nya kryptografiska algoritmer som kunde klara kraven från publika nyckelkryptosystem.

En av de första att besvara utmaningen var RSA-algoritmen som beskrevs 1977 av Ron Rivest, Adi Shamir och Len Adleman, alla vid MIT. Bokstäverna RSA är deras efternamns initialer.

Med RSA-algoritmen representeras varje meddelande i klartext och chiffertext som ett heltal i intervallet $(0, n - 1)$, d.v.s. klartext/chiffertext-längden måste vara mindre än eller lika med $\log_2 n$, i praktiken väljs klartext/chiffertext-längden att vara k där $2^k < n \leq 2^{k+1}$. Varje användare väljer sitt eget värde på n och dessutom ett par positiva heltal, e och d, för kryptering och dekryptering. Användaren placerar sin krypteringsnyckel (n,e) i en katalog för abonnenters krypteringsnycklar medan d i dekrypteringsnyckeln (n,d) hålls hemlig. Valet av e och d bestäms av nedanstående beskrivning:

Kryptering av ett klartextmeddelande, M, och dekryptering av en chiffertext, C, görs på följande sätt:

$$\text{Kryptering: } C = E(M) = M^e \text{ modulo-}n \tag{2.29}$$

$$\text{Dekryptering: } M = D(C) = C^d \text{ modulo-}n \tag{2.30}$$

Både kryptering och dekryptering går lätt att utföra och resultatet av en sådan krypterings/dekrypteringsberäkning blir ett heltal som hamnar i intervallet $(0, n - 1)$.

n bildas av produkten av två stora primtal, p och q, d.v.s.

$$n = p \cdot q \tag{2.31}$$

n ges offentlighet medan p och q hålls hemliga, eftersom forceringssvårigheten använder sig av det faktum att det är svårt att faktorisera n i p och q om p och q är tillräckligt stora.

Av p och q bildas också Eulers ϕ-funktion[29]:

$$\phi(n) = (p - 1)(q - 1) \tag{2.32}$$

Därefter väljs d att vara ett stort, godtyckligt heltal som är relativt prima till $\phi(n)$, d.v.s.

$$\gcd[\phi(n), d] = 1 \tag{2.33}$$

där gcd betyder greatest common divisor.

e bestäms som den multiplikativa inversen, modulo-$\phi(n)$, till d.

$$e \cdot d = 1 \text{ modulo-}\phi(n) \tag{2.34}$$

där $0 < e < \phi(n)$.

I nedanstående appendix, 2.12.2, visas att giltigheten av uttrycket (2.34) garanterar att sambanden (2.29) och (2.30) gäller.

2.12.2 Appendix, matematiska grunder för RSA-algoritmen

Enligt Fermats lilla sats gäller för ett godtyckligt heltal (meddelande), M, som är relativt prima till n att

$$M^{n-1} = 1 \text{ modulo-}n \tag{2.35}$$

I vårt fall är $n = pq$ och följaktligen

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

För att inse att $\phi(n) = \phi(pq)$ betrakta restklassmängden[30]

$$Z_n = \{0,1,\ldots,pq - 1\} \tag{2.36}$$

De rester som inte är relativt prima till n är mängden $\{p,2p,\ldots,$

$(q - 1)p\}$, mängden $\{q,2q,\ldots,(p - 1)q\}$ och mängden $\{0\}$. Således är

$$\begin{aligned} \phi(n) &= pq - [(q - 1) + (p - 1) + 1] = pq - (p + q) + 1 = \\ &= (p - 1)(q - 1) \end{aligned} \tag{2.37}$$

29 $\phi(n)$ = antalet positiva heltal $< n$, som är relativt prima till n. Om n är ett primtal, d.v.s. $n = p$ gäller alltså att $\phi(p) = p - 1$. På engelska finns även benämningen Eulers totient function för $\phi(n)$.

30 Z_n är mängden ickenegativa heltal $< n$.

Exempel 2.8

$\phi(21) = \phi(3 \cdot 7) = (3 - 1) \cdot (7 - 1) = 2 \cdot 6 = 12$; där de 12 heltalen är {1,2,4,5,8,10,11,13, 16,17,19,20}.

◊

Eftersom d är relativt prima till $\phi(n)$ finns en multiplikativ invers till d modulo-$\phi(n)$):

$$e \cdot d = 1 \text{ modulo-}\phi(n) \tag{2.38}$$

Vi visar nu att sambanden (2.29) och (2.30) gäller om e och d valts enligt ovanstående principer.

Det gäller:

$$D(E(M)) = (E(M))^d = (M^e)^d = M^{e \cdot d} \text{ modulo-}n \tag{2.39}$$

$$E(D(M)) = (D(M))^e = (M^d)^e = M^{e \cdot d} \text{ modulo-}n \tag{2.40}$$

och

$$M^{e \cdot d} = M^{k \cdot \phi(n) + 1} \text{ modulo-}n \quad \text{(för varje heltal } k\text{)}^{31} \tag{2.41}$$

Av (2.35) kan vi se att för alla M sådana att p inte är delare till M gäller att

$$M^{p-1} = 1 \text{ modulo-}p \tag{2.42}$$

och eftersom $(p - 1)$ är delare till $\phi(n)$ är

$$M^{k \cdot \phi(n) + 1} = M \text{ modulo-}p \tag{2.43}$$

som är trivialt sant när $M = 0$ modulo-p, och som medför att likheten gäller för *alla* M. På samma sätt gäller för q att

$$M^{k \cdot \phi(n) + 1} = M \text{ modulo-}q \tag{2.44}$$

Av uttrycken (2.43) och (2.44) följer att uttrycket (2.41) gäller.

Detta medför att

$$D(E(M)) = M \tag{2.45}$$

31 Diffie, W. och Hellman, M. E., ”New Directions in Cryptography”, *IEEE, Trans. Inf. Theory,* vol. IT22, Nov. 1976, pp. 644-654.

och

$$E(D(M)) = M \tag{2.46}$$

för alla M sådana att $0 \leq M \leq n$. E och D är därför inversa permutationer.[32]

2.12.3 Praktisk tillämpning av RSA-metoden

RSA-metoden bygger på det faktum att det är lätt att hitta två stora primtal, p och q, och multiplicera ihop dem medan det är betydligt svårare att faktorisera resultatet. Produkten $p \cdot q$ kan därför offentliggöras utan att dekrypteringsnyckeln svarande mot krypteringsnyckeln avslöjas. Det rekommenderas att var och en av faktorerna p och q består av omkring 100 digitala siffror. Multiplikationen $p \cdot q$ kan då utföras på en bråkdel av en sekund medan faktorisering av resultatet skulle ta miljardtals år.

2.12.3.1 Exempel enligt Rivest, Shamir och Adleman[33]

Låt $p = 47$ och $q = 59$. Vi får då $n = p \cdot q = 47 \cdot 59 = 2773$ och $\phi(n) =$ $= (p - 1)(q - 1) = 46 \cdot 58 = 2668$. d skall vara relativt prima till $\phi(n)$ och vi väljer $d = 157$. e skall vara multiplikativ invers modulo-2668 till d, d.v.s. e skall vara 17 (detaljerna i beräkningen av e visas i avsnitt 2.12.3.2).

Vi betraktar följande citat eller klartext av Julius Caesar (enligt W. Shakespeare):

Its all greek to me.

Varje bokstav i det engelska alfabetet (26 stycken) ersätts med ett tvåsiffrigt tal från 01 till 26 medan mellanslag kodas som 00. Ovanstående klartext kan då skrivas som:

0920 1900 0112 1200 0718 0505 1100 2015 0013 0500

32 Beviset härrör från Rich Schroeppel, University of Arizona.

33 Se Rivest, R. L., Shamir, A. och Adleman, L., ”A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”, *Communications of the ACM,* Number 2, vol. 21, Feb. 1978, pp. 120-126.

Varje klartextmeddelande skall uttryckas som ett heltal i intervallet $(0, n - 1)$. I detta exempel kan kryptering därför utföras på 4-siffriga block eftersom inte något blocks talvärde aldrig är större eller lika med $n - 1 = 2772$. Kryptering av det första 4-sifferblocket, 0920, utförs enligt:

$$C = M^e \text{ modulo-}n = 920^{17} \text{ modulo-}2773 = 948$$

Med samma metod fås så småningom hela det krypterade meddelandet:

0948 2342 1084 1444 2663 2390 0778 0774 0219 1655

Klartexten återfås genom användande av dekrypteringsnyckeln och det första blocket blir:

$$M = C^d \text{ modulo-}n = 948^{157} \text{ modulo-}2773 = 920$$

2.12.3.2 Beräkning av *e*

För att beräkna e används en variant av Euklides algoritm för att bestämma den största gemensamma divisorn (SGD) till $\phi(n)$ och d. Vi börjar med att beräkna talföljden $x_0, x_1, x_2, \ldots$, där $x_0 = \phi(n)$, $x_1 = d$ och $x_{i+1} = x_{i-1}$ modulo-x_i tills en term $x_k = 0$ hittas.

Då är $\text{SGD}(x_0, x_1) = x_{k-1}$. Beräkna för varje x_i tal a_i och b_i så att $x_i = a_i x_0 + b_i x_1$. Om $x_{k-1} = 1$ är b_{k-1} multiplikativ invers till x_1 modulo-x_0. Om b_{k-1} är ett negativt tal blir lösningen $b_{k-1} + \phi(n)$.

Beräkning av e enligt exemplet i avsnitt 2.12.3.1 blir då:

i	x_i	a_i	b_i	y_i
0	2668	1	0	
1	157	0	1	16
2	156	1	– 16	1
3	1	– 1	17	

Tabell 2.3

där

$$y_i = \left\lfloor \frac{x_{i-1}}{x_i} \right\rfloor \quad (\left\lfloor \frac{x_{i-1}}{x_i} \right\rfloor \text{betyder heltalsdelen av } \frac{x_{i-1}}{x_i})$$

$$x_{i+1} = x_{i-1} - y_i x_i$$

$$a_{i+1} = a_{i-1} - y_i a_i$$

$$b_{i+1} = b_{i-1} - y_i b_i$$

varav $e = b_3 = 17$

2.13 Elgamals[34] kryptosystem, diskret logaritm

Elgamals publika kryptosystem har en *trapdoor one-way function* som bygger på att det är relativt lätt att utföra exponentiering (d.v.s. upprepad multiplikation) modulo ett primtal, $\beta = \alpha^a$ modulo-p, medan beräkning av inversen, den diskreta logaritmen: $a = \log_\alpha \beta$ modulo-p, är mycket svårare. I ett kryptosystem utgörs den publika nyckeln av p, α och β medan a är en privat nyckel. Krypteringsoperationen är slumpad eftersom chiffertexten både beror på klartexten och ett slumpvärde, k ($0 \leq k \leq p - 1$), som väljs av sändaren. På så sätt finns det många chiffertexter som svarar mot samma klartext, i själva verket $p - 1$ stycken.

Med ovanstående samband mellan a, α, p och β fungerar ett Elgamalsystem i grova drag på följande sätt:

Klartexten, x, maskeras genom att den multipliceras[35] med β^k. Produkten (chiffertexten) kallas $y_2 = x \cdot \beta^k$. Tillsammans med chiffertexten överförs också $y_1 = \alpha^k$.

34 Taher Elgamal, egyptisk-amerikansk kryptolog, 1955 -.
35 Alla multiplikationer utförs modulo-p, d.v.s i en ändlig multiplikativ grupp – se hur detta görs i kapitel 3 avsnitt Algebra för kryptografi.

Mottagaren erhåller alltså $y = (y_1, y_2) = (\alpha^k, x \cdot \beta^k)$. Med hjälp av den privata nyckeln a (= $\log_\alpha \beta$) kan han sedan utifrån $y_1 = \alpha^k$ beräkna β^k. Slutligen kan han ”demaskera” delen $y_2 = x \cdot \beta^k$ genom att dividera med β^k för att få x.

Exempel 2.9 [36]

$p = 2579$.

$\alpha = 2$; α är ett primitivt element modulo-p.

$a = 765$.

$\beta = \alpha^a$ modulo-p = 2^{765} modulo-2579 = *949.*

Sänt meddelande:

$x = 1299$.

Sändaren väljer slumptalet

$k = 853$

och beräknar

$y_1 = \alpha^k \cdot$ modulo-p = $2^{853} \cdot$ modulo-2579 = 435.

$y_2 = x \cdot \beta^k$ modulo-p = $1299 \cdot 949^{853}$ modulo-2579 = 2396.

Mottagaren får alltså:

$y = (y_1, y_2) = (\alpha^k \cdot$ modulo-p, $x \cdot \beta^k$ modulo-p) = (435, 2396).

Eftersom mottagaren har tillgång till den privata nyckeln

$a = 765$

och det gäller att

$a = \log_\alpha \beta$ (diskret logaritm: $765 = \log_2 2^{765}$)

får vi alltså att

α^a modulo-p = β modulo-p

och följaktligen α^{ak} modulo-p = β^k modulo-p.

---

36 För algebran hänvisas till kapitel 3 avsnitt Algebra för kryptografi.

Med $y_1 = \alpha^k$ modulo-p = 435 fås α^{ak} modulo-p = β^k modulo-p = 435^{765}.

Dechiffreringen blir:

$x \cdot \beta^k \cdot (\beta^k)^{-1}$ modulo-p = $2396 \cdot (435^{765})^{-1}$ modulo-*2579* =

= $2396 \cdot 1980$ modulo-2579 = 1299.

OBS! $(435^{765})^{-1}$ modulo-2579 = 1980 är det tal som satisfierar

$435^{765} \cdot (435^{765})^{-1} = 1$ modulo-2579.

◊

Elgamals kryptosystem är naturligtvis osäkert om en forcör kan beräkna den diskreta logaritmen $a = \log_\alpha \beta$ eftersom han då kan dekryptera meddelandet på samma sätt som den avsedda mottagaren. Ett nödvändigt villkor för att kryptosystemet skall vara säkert är att det skall vara svårt att beräkna den diskreta logaritmen. Det anses allmänt att detta kan uppnås genom att valet av p görs med omsorg medan α är ett primitivt element modulo-p. Dessutom skall p bestå av minst 300 digitala siffror (c:a 1000 bitar) och $p - 1$ skall faktoriseras i minst en ”stor” primfaktor.

2.14 Kryptografi med elliptiska kurvor

På senare tid har ett med RSA konkurrerande system, elliptisk kurvkryptografi (ECC, Elliptic Curve Cryptography), ökat i användning. En av anledningarna till detta är att säkerhetskraven gjort att nyckellängden för RSA har ökat på senare tid, något som gjort att belastningen på applikationer som använder RSA ökat.

Fördelen med ECC, jämfört med RSA, är att den verkar ge lika stor säkerhet med mycket kortare nyckellängd. Å andra sidan har ECC inte funnits så lång tid att förtroendet för den blivit lika högt som för RSA.

2.14.1 Elliptiska kurvor

Elliptiska kurvor är inte ellipser men kallas så för att de beskrivs av kubiska ekvationer som liknar dem som används vid beräkning av ellipsens omkrets. Elliptiska kurvor definierade modulo ett primtal, p, är av central betydelse för publik-nyckel kryptografi.

Vi börjar med att studera elliptiska kurvor över de reella talen eftersom vissa begrepp är naturligare att motivera i ett sådant sammanhang.

2.14.1.1 Elliptiska kurvor över de reella talen

Definition 2.1 En *icke-singulär elliptisk kurva* utgörs av den reella mängden E av lösningar (x,y) till ekvationen

$$y^2 = x^3 + ax + b \tag{2.47}$$

där a och b är reella tal så att $4a^3 + 27b^2 \neq 0$ samt en speciell punkt, O, kallad punkten i oändligheten eller nollpunkten.

Villkoret $4a^3 + 27b^2 \neq 0$ är nödvändigt för att garantera att ekvationen (2.47) har tre distinkta rötter, reella eller komplexa. Om $4a^3 + 27b^2 = 0$ kallas motsvarande elliptiska kurva en *singulär elliptisk kurva*.

Figur 2.26 visar den icke-singulära elliptiska kurvan $y^2 = x^3 + ax + b$ där $a = -4$ och $b = 0$, d.v.s. $y^2 = x^3 - 4x$.

Med utgångspunkt från E som är en icke-singulär elliptisk kurva kan vi nu definiera en binär operation över E som gör E till en additiv kommutativ grupp[37].

Punkten i oändligheten, O, är identitetselementet så att

$$P + O = O + P = P$$

för alla $P \in E$. Betrakta med $P = (x_1,y_1)$ och $Q = (x_2,y_2)$ följande fall:

1. $x_1 \neq x_2$.
2. $x_1 = x_2$ och $y_1 = -y_2$.

37 Egenskapen kommutativ benämns också *abelsk*.

3. $x_1 = x_2$ och $y_1 = y_2$.

Fall 1. Kan åskådliggöras av linjen, som skär den elliptiska kurvan E i punkterna P och Q – se Figur 2.26. Av figuren framgår att linjen skär kurvan i ytterligare en punkt, R'. Motsvarande i x-axeln reflekterade punkt kallas R och vi definierar: $P + Q = R$.

Utgående från punkterna $P\ (x_1,y_1)$ och $Q\ (x_2,y_2)$ kan vi få ett uttryck för R:

Låt linjen, som skär den elliptiska kurvan i P och Q, ha ekvationen

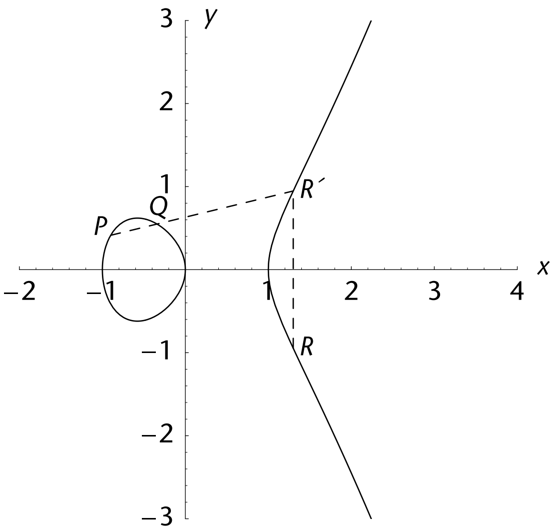
$$y = \lambda x + \mu \tag{2.48}$$

där

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \tag{2.49}$$

och

$$\mu = y_1 - \lambda x_1 = y_2 - \lambda x_2 \tag{2.50}$$



Figur 2.26 Den elliptiska kurvan $y^2 = x^3 - 4x$.

Vi substituerar uttrycken (2.48) och (2.50) i den elliptiska kurvans ekvation (2.47) och får:

$$(\lambda x + \mu)^2 = x^3 + ax + b \tag{2.51}$$

eller, uttryckt i polynomform:

$$x^3 - \lambda^2 x^2 + (a - 2\lambda\mu)x + b - \mu^2 = 0 \tag{2.52}$$

Rötterna till ekvation (2.52) utgörs av x-koordinaterna för skärningspunkterna mellan linjen och den elliptiska kurvan. Vi känner redan två av dessa nämligen x_1 och x_2 som båda är reella, vilket medför att den tredje också måste vara reell. Eftersom ekvation (2.52) är av tredje graden är rötternas summa lika med minus andragradstermens koefficient, d.v.s. λ^2.

Alltså är

$$x_3 = \lambda^2 - x_1 - x_2 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \tag{2.53}$$

där x_3 är x-koordinaten för punkten $R´$.

Om y-koordinaten för punkten $R´$ kallas $-y_3$ blir y-koordinaten för punkten R y_3. Med de två punkterna $P\,(x_1,y_1)$ och $R´\,(x_3, -y_3)$ på linjen kan vi på ett enkelt sätt, få ett alternativt uttryck för riktningskoefficienten λ:

$$\lambda = \frac{-y_3 - y_1}{x_3 - x_1} \tag{2.54}$$

som ger

$$\begin{aligned} y_3 &= \lambda\left(x_1 - x_3\right) - y_1 = \frac{y_2 - y_1}{x_2 - x_1} \cdot \left\{x_1 - \left[\left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2\right]\right\} - y_1 = \\ &= -\left(\frac{y_2 - y_1}{x_2 - x_1}\right)^3 + 2x_1\frac{y_2 - y_1}{x_2 - x_1} + x_2\frac{y_2 - y_1}{x_2 - x_1} - y_1 \end{aligned} \tag{2.55}$$

Fall 2. Där $x_1 = x_2$ och $y_1 = -y_2$ kan åskådliggöras med en linje, parallell med y-axeln, som skär den elliptiska kurvan i två punkter symmetriskt kring x-axeln. Detta fall är enkelt, vi definierar $(x,y) + (x,-y)$

$= O$ för alla $(x,y) \in E$, d.v.s. (x,y) och $(x,-y)$ är varandras inverser med avseende på den elliptiska kurvans additionsoperation.

Fall 3. I detta fall adderar vi en punkt P (x_1,y_1) till sig själv. Vi kan uppfatta detta fall som en urartning av Fall 1 på så sätt att de två punkterna P och Q sammanfallit. Linjen som i Fall 1 skär kurvan i P och Q blir i stället en tangent vars riktningskoefficient kan bestämmas ur kurvans ekvation (2.47) genom implicit derivering:

$$2y\frac{\mathrm{d}y}{\mathrm{d}x} = 3x^2 + a \qquad (2.56)$$

Genom att i den deriverade ekvationen (2.56) substituera tangeringspunktens P koordinater (x_1,y_1) kan vi beräkna ett uttryck för tangentens riktningskoefficient:

$$\lambda = \frac{3x_1^2 + a}{2y_1} \qquad (2.57)$$

För övrigt blir uttrycket detsamma som i fall 1. med den skillnaden att riktningskoefficienten beräknas på ett annorlunda sätt.

Av det ovanstående kan följande egenskaper för additionsoperationen sammanfattas:

1. Addition är sluten över mängden E.
2. Addition är kommutativ.
3. O är ett identitetselement med avseende på addition.
4. Varje punkt (eller element) på E har ett med avseende på addition inverst element.

2.14.1.2 Elliptiska kurvor modulo ett primtal

Vid elliptisk kurvkryptografi används ekvationen för elliptiska kurvor (2.47) med alla variabler och koefficienter begränsade till elementen i ändliga talkroppar. Två typer av elliptiska kurvor används i kryptografiska tillämpningar, dels primtalskurvor med variabler och koefficienter tagna från Z_p, dels binära kurvor med variabler och koefficienter tagna från $GF(2^n)$[38]. Primtalskurvor är bäst lämpade för mjukvarutillämpningar medan binära kurvor är bättre för hårdvarutillämpningar.

Elliptiska kurvor över Z_p definieras på samma sätt som elliptiska kurvor definieras över de reella talen, däremot finns det inte en geometrisk tolkning som vid definitionen över de reella talen:

Definition 2.2 En *elliptisk* kurva $y^2 = x^3 + ax + b$ över Z_p utgörs av lösningsmängden E till

$$y^2 \text{ modulo-}p = (x^3 + ax + b) \text{ modulo-}p \tag{2.58}$$

där a och $b \in Z_p$ är konstanter så att $4a^3 + 27b^2 \neq 0$ samt en speciell punkt, O, kallad punkten i oändligheten eller nollpunkten.

Additionsoperationen över E, definieras på följande sätt (alla aritmetiska operationer utförs i Z_p):

Låt $P = (x_1,y_1)$ och $Q = (x_2,y_2)$ vara punkter på E. Om $x_2 = x_1$ och $y_2 = -y_1$ gäller att $P + Q = O$ annars är $P + Q = (x_3,y_3)$ där

$$x_3 = \lambda^2 - x_1 - x_2 \tag{2.59}$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \tag{2.60}$$

och

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{om } P \neq Q \\[2ex] \dfrac{3x_1^2 + a}{2y_1} & \text{om } P = Q \end{cases} \tag{2.61}$$

Exempel 2.10

E är den elliptiska kurvan $y^2 = x^3 + x + 6$ över Z_{11}. Bestäm punkterna på E.

38 GF = Galois Field = Galois-talkropp.

Lösning:

Genom att bestämma $x^3 + x + 6$ modulo-11 för alla $x \in Z_{11}$ och därefter lösa y i ekvationen (2.58) kan vi undersöka om $z = x^3 + x + 6$ modulo-11 är en kvadratisk rest genom att tillämpa Eulers kriterium:

Definition 2.3 Eulers kriterium: Om p är ett udda primtal är z en kvadratisk rest modulo-p om och endast om

$$z^{(p-1)/2}\text{modulo-}p = 1 \tag{2.62}$$

Vi ställer upp en tabell – se Tabell 2.4 – för $x^3 + x + 6$ modulo-11 för alla

$x \in Z_{11}$ och $z^{(p-1)/2}$ modulo-11 där $z = x^3 + x + 6$:

x	z modulo-11 $= x^3 + x + 6$ modulo-11	$z^{(11-1)/2}$ modulo-11	kvadratisk rest	y
0	6	10	nej	-
1	8	10	nej	-
2	5	1	ja	4;7
3	3	1	ja	5;6
4	8	10	nej	-
5	4	1	ja	2;9
6	8	10	nej	-
7	4	1	ja	2;9
8	9	1	ja	3;8
9	7	10	nej	-
10	4	1	ja	2;9

Tabell 2.4 x och y-koordinater på den elliptiska kurvan $y^2 = x^3 + x +$ $+ 6$ över Z_{11}.

Det är alltså $x = 2$; $x = 3$; $x = 5$; $x = 7$ och $x = 10$ som är ekvationens x-lösningar. Återstår att beräkna motsvarande y-värden som fås genom att dra roten ur z modulo-11 $= x^3 + x + 6$ modulo-11 för $x =$ $= 2$; $x = 3$; $x = 5$; $x = 7$ och $x = 10$. Lyckligtvis finns en explicit formel för att beräkna kvadratrötterna till kvadratiska rester modulo-p för primtal för vilka det gäller att $p \equiv 3$ modulo-4 enligt:

$$y = \sqrt{z}\text{modulo-11} = \pm \sqrt{z^{(11+1)/2}} \text{ modulo-11} = \pm z^3 \text{ modulo-11}$$

För $x = 2$ får vi: $y = \pm (2^3 + 2 + 6)^3$ modulo-11 = ± 4096 modulo-11

= ± 4 modulo-11 = (4;7) o.s.v.

Resultatet återges i högerkolumnen i Tabell 2.4.

x och y-koordinaterna på den elliptiska kurvan $y^2 = x^3 + x + 6$ över Z_{11} återges i Figur 2.27. Som jämförelse har motsvarande analoga kurva, modulo-11, ritats in.

Eftersom varje (x,y)-koordinat på den elliptiska kurvan motsvarar ett element i en additiv grupp och det finns 12 element plus elementet/punkten i oändligheten är gruppen isomorf med Z_{13} och det gäller att varje element utom elementet i oändligheten är ett primitivt element, d.v.s. kan generera de övriga elementen. Ta som exempel elementet $\alpha = (2,7)$ (d.v.s. $x_1 = 2$; $y_1 = 7$). Vi bildar nästa element genom gruppadditionen, här benämnd g.a. :

$2\alpha = \alpha$ g.a. $\alpha = (x_1,y_1)$ g.a. $(x_1,y_1) = (2,7)$ g.a. $(2,7)$.

För att beräkna denna gruppaddition beräknar vi först λ enligt den andra raden i uttrycket (2.61):

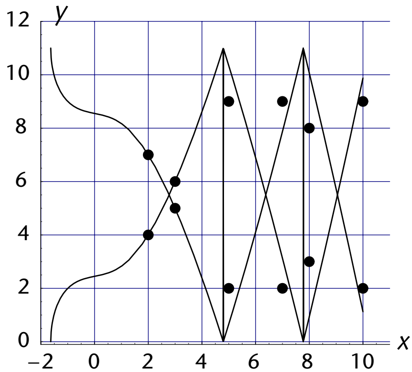
$$\lambda = \frac{3 \cdot x_1^2 + a}{2 \cdot y_1} \text{ modulo-11} = \frac{3 \cdot 2^2 + 1}{2 \cdot 7} \text{ modulo-11} =$$

$$= \frac{2}{3} \text{ modulo-11} = 2 \cdot 4 \text{ modulo-11} = 8$$

Därefter får vi med hjälp av uttrycket (2.59):

$x_3 = (\lambda^2 - x_1 - x_2) = (8^2 - 2 - 2)$ modulo-11 = 5

och med (2.60):



Figur 2.27 *x och y-koordinater på den elliptiska kurvan $y^2 = x^3 + x + 6$ över Z_{11} samt motsvarande analoga kurva modulo-11.*

$y_3 = \lambda(x_1 - x_3) - y_1 = (8(2 - 5) - 7)$ modulo-11 $= 2$

$\Rightarrow 2\alpha = (5,2)$

Genom att fortsätta på detta sätt får man alla element:

$\alpha = (2,7)$	$2\alpha = (5,2)$	$3\alpha = (8,3)$
$4\alpha = (10,2)$	$5\alpha = (3,6)$	$6\alpha = (7,9)$
$7\alpha = (7,2)$	$8\alpha = (3,5)$	$9\alpha = (10,9)$
$10\alpha = (8,8)$	$11\alpha = (5,9)$	$12\alpha = (2,4)$

Anmärkning: För att uppskatta antalet punkter på en elliptisk kurva, 13 i Exempel 2.10, definierad över Z_p, i Exempel 2.10: $p = 11$, kan följande teorem av Hasse[39] tillämpas:

$$p + 1 - 2\sqrt{p} \leq \#E \leq p + 1 + 2\sqrt{p} \tag{2.63}$$

där $\#E$ betecknar antalet punkter på den elliptiska kurvan. Det exakta antalet är svårare att beräkna men Schoof[40] har konstruerat en algoritm (1984) för detta ändamål.

◊

39 Helmut Hasse, tysk matematiker, 1898-1979.
40 René Schoof, holländsk matematiker.

2.14.1.3 Binära elliptiska kurvor över GF(2^m)

En ändlig talkropp GF(2^m) består av 2^m element[41]. Den kubiska ekvation som används i samband med binära elliptiska kurvor har en något annorlunda form än den som används för elliptiska kurvor modulo ett primtal. Formen är:

$$(y^2 + xy) \text{ modulo-}2^m = (x^2 + ax + b) \text{ modulo-}2^m \tag{2.64}$$

där variablerna och koefficienterna förutsätts vara element i GF(2^m).

2.14.1.4 Elliptisk kurvkryptering/dekryptering

Det enklaste systemet för elliptisk kurvkryptering fungerar på analogt sätt med Elgamals kryptosystem med den skillnaden att de multiplikativa operationerna byts mot additiva.
Systemet har en *trapdoor one-way function* som bygger på att det är relativt lätt att utföra multiplikation (d.v.s. upprepad addition) modulo ett primtal, $\beta = \alpha \cdot a$ modulo-p, medan beräkning av inversen, $a = \beta/\alpha$ modulo-p, (d.v.s. upprepad subtraktion), är mycket svårare. I ett kryptosystem utgörs den publika nyckeln av p, α och β medan a är en privat nyckel. Krypteringsoperationen är slumpad eftersom chiffertexten beror såväl på klartexten som ett slumpvärde, k, som väljs av sändaren. På så sätt finns det många chiffertexter som svarar mot samma klartext, i själva verket $p - 1$ stycken.

Med ovanstående samband mellan a, α, p och β fungerar ett elgamalsystem med elliptisk kurvkryptering i grova drag på följande sätt: (alla additions- och multiplikations-operationer utförs modulo-p)

Klartexten, x, maskeras genom att till den addera $\beta \cdot k$. Summan (chiffertexten) kallas $y_2 = x + \beta \cdot k$. Tillsammans med chiffertexten överförs också $y_1 = \alpha \cdot k$.
Mottagaren erhåller alltså $y = (y_1, y_2) = (\alpha \cdot k, x + \beta \cdot k)$. Med hjälp av den privata nyckeln $a = \beta/\alpha$ kan han sedan utifrån $y_1 = \alpha \cdot k$ beräkna $\beta \cdot k$. Slutligen kan han ”demaskera” delen $y_2 = x + \beta \cdot k$ genom att subtrahera $\beta \cdot k$ för att få x.

41 Räkneregler för ändliga talkroppar genomgås i kapitel 3.

Exempel 2.11

$p = 13$.

$\alpha = (2,7)$; α är ett primitivt element modulo-p.

$a = 7$.

$\beta = \alpha \cdot a$ modulo-$p = (2,7) \cdot 7 = (7,2)$.

Sänt meddelande:

$x = (10,9)$ som är en punkt på den elliptiska kurvan $y^2 = x^3 + x + 6$ över Z_{11}.

Sändaren väljer slumptalet

$k = 3$ $(0 \leq k \leq p - 1) = (0 \leq k \leq 12)$

och beräknar

$y_1 = \alpha \cdot k$ modulo-$p = (2,7) \cdot 3 = (8,3)$.

$y_2 = (x + \beta \cdot k)$ modulo-$p = ((10,9) + (7,2) \cdot 3)$ modulo-13 $= ((10,9) + (7\alpha \cdot 3))$ modulo-13 $= ((10,9) + 8\alpha)$ modulo-13 $=$

$= ((10,9) + (3,5))$ modulo-13 $= (9\alpha + 8\alpha)$ modulo-13 $= 4\alpha = (10,2)$.

Mottagaren får alltså

$y = (y_1, y_2) = (\alpha \cdot k$ modulo-p, $(x + \beta \cdot k)$ modulo-$p) = ((8,3), (10,2))$.

Eftersom mottagaren har tillgång till den privata nyckeln

$a = 7$

och det gäller att

$a = \beta/\alpha$ modulo-p

får vi alltså att

$\alpha \cdot a$ modulo-$p = \beta$ modulo-p

och följaktligen $\alpha \cdot a \cdot k$ modulo-$p = \beta \cdot k$ modulo-p.

Med $y_1 = \alpha \cdot k$ modulo-$p = (8,3)$ fås $\alpha \cdot a \cdot k$ modulo-$p = \beta \cdot k$ modulo-$p = (7,2) \cdot 3 = (3,5)$.

Dechiffreringen blir:

$(x + \beta \cdot k - \beta \cdot k)$ modulo-p = ((10,2) – (3,5)) modulo-13 = ((10,2) + + (3,6)) mod-13 = (10,9), d.v.s. dekrypteringen ger det riktiga meddelandet.

◊

2.14.1.5 Kryptosystem med diskret logaritm och med elliptiska kurvor

Enligt uppskattningar gjorda av Lenstra och Verheul[42] bör bitantalet, för att, ett på elliptiska kurvor med diskret logaritm baserat kryptosystem, uppgå till ungefär 160, vara säkert till år 2020. Detta kan jämföras med Elgamals kryptosystem med diskret logaritm som för motsvarande uppskattade säkerhet kräver minst 1880 bit. Anledningen till den stora skillnaden mellan dessa system är att det inte finns någon känd metod att beräkna diskreta logaritmer över elliptiska kurvor.

Tack vare detta har elliptisk kurvkryptografi fått stor praktisk betydelse för tillämpningar med begränsat minnesutrymme som trådlösa enheter och smartcard.

2.15 Dataintegritet och autenticitet

Att data inte har manipulerats av utomstående under överföringen utgör krav på dataintegritet och att data kommer från den som de uppges komma från samt når avsedd mottagare utgör krav på autenticitet.

För att uppfylla kravet på autenticitet finns ett antal funktioner som kan användas för att producera en autentikator. Autentikatorerna kan indelas i tre typer nämligen:

42 Arjen K. Lenstra och Eric R. Verheul, ”Selecting Cryptographic Key Sizes”, *Journal of Cryptology*, vol. 14, Dec. 2001, pp. 255-293.

- *Meddelandekryptering* vid vilken det krypterade meddelandet fungerar som autentikator för hela meddelandet.
- *Hashfunktion*: Meddelandetexten producerar, oberoende av längd, ett hashvärde med konstant längd, som tjänar som autentikator.
- *Message Authentication Code* (MAC): En funktion av meddelandetexten och en hemlig nyckel producerar ett värde med fix längd, som tjänar som autentikator.

PGP, Pretty Good Privacy är ett dataprogram som möjliggör dataöverföring utan avlyssning samt autenticitetskontroll (eller autenticering). PGP är i sina olika versioner världens mest använda kryptosystem.

Kerberos är ett autenticeringsprotokoll för nätverk som möjliggör säker kommunikation via osäkra nätverk. Kerberos bygger på ett kryptosystem med symmetriska nycklar och kräver en pålitlig tredje part).

2.15.1 PGP, Pretty Good Privacy

PGP utvecklades av Phil Zimmerman 1991. PGP används för autenticitetskontroll och bevarande av meddelandehemlighet i tillämpningar som e-post och fillagring.

PGP består av de fem tjänsterna: Autenticitetskontroll, bevarande av meddelandehemlighet, datakompression, e-postanpassning och segmentering. Dessa fem tjänster, översiktligt i Tabell 2.5, presenteras i följande 5 avsnitt.

tjänst/funktion	algoritm
autenticitetskontroll med digital signatur	DSS[1]/SHA eller RSA/SHA
meddelandehemlighet medelst kryptering	CAST[2] eller IDEA[3] eller 3DES eller RSA
datakompression	ZIP
e-postanpassning	radix-64-omvandling
segmentering	-

Tabell 2.5 Översikt av PGP-tjänster.

1 DSS = Digital Signature Standard.
2 CAST är en symmetrisk krypteringsalgoritm, initialerna kommer av skaparnas namn: Carlisle Adams och Stafford Tavares.
3 IDEA = International Data Encryption Algorithm.

2.15.1.1 Autenticitetskontroll

Autenticitetskontroll utförs genom att meddelandet förses med en hashkod (t.ex. 160 bit SHA-1). Hashkoden krypteras med sändarens *privata nyckel*[43] varefter resultatet, den *digitala signaturen*, läggs in framför meddelandet. Mottagaren dekrypterar hashkoden med hjälp av sändarens publika nyckel och jämför den med en i mottagaren lokalt genererad hashkod. Om den dekrypterade och den lokalt genererade hashkoden är lika accepteras meddelandet som autentiskt.

43 Vid PGP används en privat nyckel som finns tillsammans med den publika (enligt PGP-dokumentationen egentligen en *hemlig nyckel – secret key* – men för att inte förväxlas med begreppet secret key använt vid symmetrisk kryptering används hellre benämningen *privat nyckel – private key*).

2.15.1.2 Bevarande av meddelandehemlighet

Bevarande av meddelandehemlighet uppnås genom att meddelandet krypteras.

Ett problem som måste beaktas är distributionen av nycklar. I PGP används varje symmetrisk nyckel endast en gång och varje ny nyckel genereras som ett 128-bits slumptal för varje meddelande. Eftersom nyckeln, *session key*, används endast en gång, är den i realiteten en *one-time key*, som är kopplad till och sänds tillsammans med meddelandet. För att skydda nyckeln krypteras den tillsammans med mottagarens publika nyckel.

Hela sessionen kan beskrivas enligt:

1. Sändaren genererar ett meddelande och ett 128-bits slumptal för en sessionsnyckel som endast skall användas för detta meddelande.
2. Meddelandet krypteras med användande av CAST-128 (eller IDEA eller 3DES) med sessionsnyckeln.
3. Sessionsnyckeln krypteras med RSA med användande av mottagarens publika nyckel och sätts framför det krypterade meddelandet.
4. Mottagaren använder RSA med sin privata nyckel för att dekryptera och få fram sessionsnyckeln.
5. Sessionsnyckeln används för att dekryptera det krypterade meddelandet.

2.15.1.3 Bevarande av meddelandehemlighet och autenticitetskontroll

Först genereras en digital signatur, enligt avsnitt 2.15.1.1, som sätts framför meddelandet därefter fortsätter proceduren enligt avsnitt 2.15.1.2.

2.15.1.4 Datakompression

Som default-tillstånd komprimeras meddelandet efter det att den digitala signaturen har tillfogats meddelandet men innan krypteringen gjorts. Detta förfarande har fördelen att utrymme sparas såväl för e-post som fillagring. Som kompressionsalgoritm används ZIP.

2.15.1.5 E-postanpassning

När PGP används är åtminstone någon del av det översända meddelandet krypterat. Detta medför att det överförda meddelandeblocket helt eller delvis består av en ström av godtyckliga 8-bits ord. Eftersom många mailsystem endast tillåter block som består av ASCII-kod omvandlar PGP-systemet de godtyckliga 8-bits orden till skrivbara ASCII-symboler. Omvandlingen görs med radix-64-omvandling som i korthet fungerar så att varje grupp om tre 8-bits ord avbildas på fyra ASCII-symboler. Dessutom tillfogas en CRC[44] för att upptäcka fel som kan uppstå vid överföringen.

2.15.1.6 Segmentering

E-postsystemet har vanligen en övre gräns för hur långa meddelanden som kan överföras i ett meddelandeblock. I internetsystemet får t.ex. maximalt 50.000 8-bits ord överföras i ett meddelandeblock. Är meddelandet längre måste det delas upp i kortare segment som sänds var för sig. I PGP-systemet sker en uppdelningn i kortare segment automatiskt om meddelandet är för långt för att sändas via e-post. Denna meddelandesegmentering görs efter det att all annan databehandling, inklusive radix-64-omvandling, gjorts. I mottagaren måste PGP-systemet först ta bort alla e-postheaders och rekonstruera det ursprungliga meddelandeblocket innan dekompression, dekryptering m.m. görs.

44 CRC = Cyclic Redundancy Check.

2.15.2 Kerberos[45]

MIT (Massachusetts Institute of Technology) utvecklade autenticitetsprotokollet Kerberos för att skydda nätverk. Protokollet är baserat på Needham-Schroeders[46] protokoll. Protokollet tillåter individer att vid kommunikation över ett osäkert nätverk, på ett säkert sätt, bevisa sin identitet för andra. Upphovsmännen avsåg Kerberos att vara ett client-server-system med ömsesidig autenticering – både användare och använd tjänst skall verifiera sin identitet för varandra. Kerberos förekommer i ett flertal versioner. Versionerna 1-3 fanns bara internt inom MIT. Steve Miller och Clifford Neuman publicerade version 4 i slutet av 1980-talet. Version 5, utarbetad av John Kohl och Clifford Neuman, som kom 1993 var avsedd att övervinna begränsningar och säkerhetsproblem i version 4. Under en viss tid klassades Kerberos av de amerikanska myndigheterna som krigsmateriel och de förbjöd dess export eftersom den använde sig av DES-kryptering (med 56-bits nyckel). En svensk version gjorde systemet tillgängligt utanför USA innan de amerikanska exportbestämmelserna ändrades (omkring år 2000). Kerberos används i bl.a. Windows 2000, Windows XP och Windows Server 2003.

För detaljer i protokollet hänvisas till:

http://web.mit.edu/kerberos/

45 Protokollet fick sitt namn efter den monstruösa trehövdade hunden i Hades, känd från den grekiska mytologin.

46 Roger M. Needham och Michael D. Schroeder, ”Using Encryption for Authentication in Large Networks of Computers”, *Communications of the ACM*, vol. 21, Dec. 1978, Number 12, pp. 993-999.

2.16 Kvantkryptografi

Ett problem vid all form av sändning av chiffertext är nyckeldistributionen. Om en forcör kommer över en nyckel när den sänds till mottagaren kan han dekryptera alla överförda chiffertexter.

Vid kvantkryptografering kan, p.g.a. de kvantmekaniska lagarna, en forcör inte avlyssna överföringen utan att störa den. Principen fungerar så att två (eller fler) kvantmekaniskt hopkopplade (eng. entangled) fotoner endast kan befinna sig i ett bestämt antal tillstånd, t.ex. den ena vertikalt och den andra horisontellt polariserad eller den ena cirkulärpolariserad motsols och den andra medsols. Om nu sändaren förfogar över den ena fotonen och mottagaren över den andra får de alltså alltid motsatt resultat vid en mätning. Däremot kan sändaren inte bestämma vilken av två polarisationstyper fotonen i varje ögonblick skall ha, d.v.s. polariseringen är sant stokastisk.

Tillstånden har dessutom en egenskap kallad kvantum-ickelokalitet som inte gäller inom klassisk fysik och som innebär att de två kvantmekaniskt hopkopplade fotonerna inte behöver befinna sig på samma plats. Detta innebär idealt att om sändaren mäter ”vertikalpolariserad” kommer mottagaren att mäta ”horisontalpolariserad”. Vid praktiska mätningar med överföring via fiberkabel har det visat sig att korrelationen inte är hundraprocentig men dock bättre än rena slumpen (50%).

Varje försök att avlyssna överföringen innebär kvantmekaniskt att de hopkopplade fotonernas tillstånd påverkas och försvagar därigenom korrelationen på ett sätt som kan detekteras av sändare och mottagare.

Med hjälp av den ovan beskrivna effekten kan sändare och mottagare dela en helt slumpmässig nyckel. Den svaga korrelationen kan förbättras genom att sända ett dataord med fler bitar representerande de slumpvis varierande tillstånden. På dataordet kan sedan felkorrigerande kodning tillämpas. Den kryptografiska versionen av felkorrigering kallas privathetsförstärkning (eng. privacy amplification).

2.17 Appendix

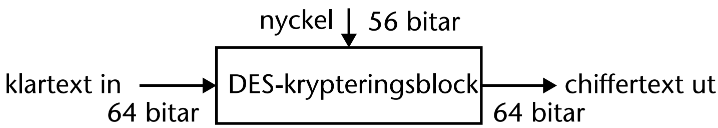
2.17.1 DES, detaljer

Ett block med 64 bitar ansluts till DES-krypteringens ingång – se Figur 2.28.

De 64 klartextbitarna på blockingången kan uppfattas som en symbol i ett alfabet med 2^{64} symboler som på blockutgången ersätts med en 64 bitars chiffertextsymbol.

Figur 2.28 illustrerar systemfunktionerna blockschemamässigt. Krypteringsalgoritmen börjar med en initialpermutation av de 64 klartextbitarna som är anslutna till krypteringsblockets ingång – se Tabell 2.6 .

Tabellen läses från vänster till höger, uppifrån och nedåt så att de 64 bitarna $x_1, x_2, x_3, \ldots, x_{64}$ permuteras till $x_{58}, x_{50}, x_{42}, \ldots, x_7$



Figur 2.28 DES-krypteringsblock.

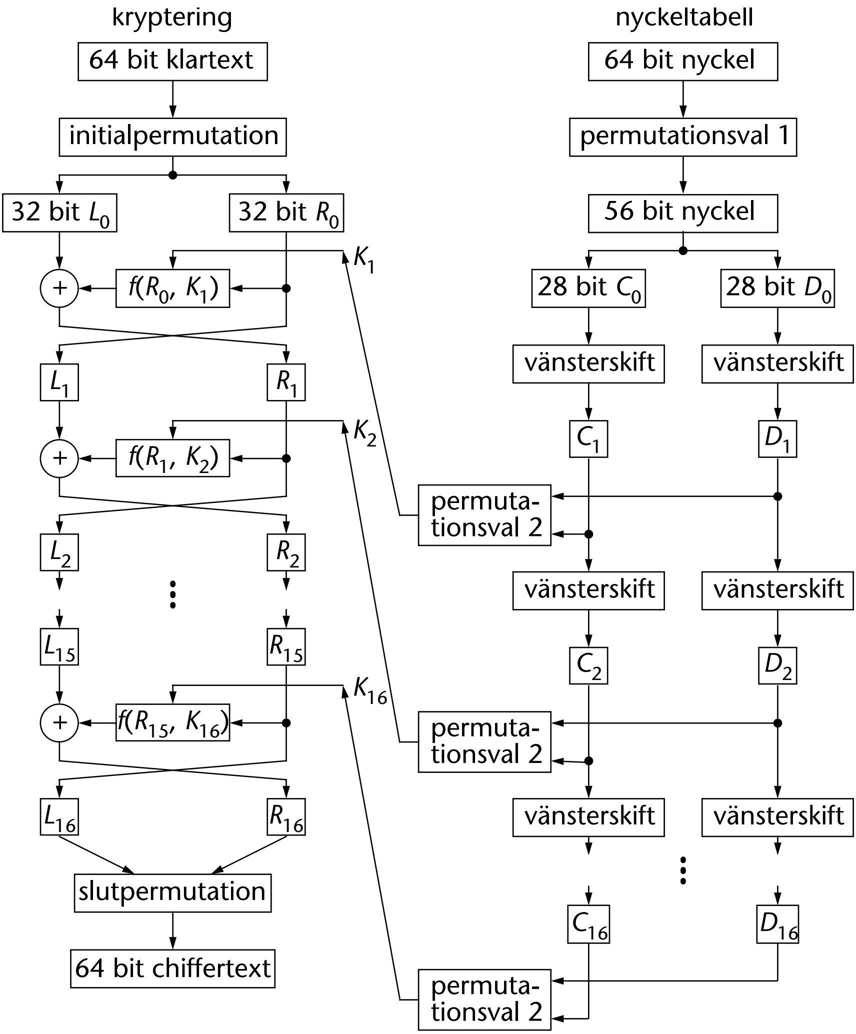
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Tabell 2.6 Initialpermutation i DES.

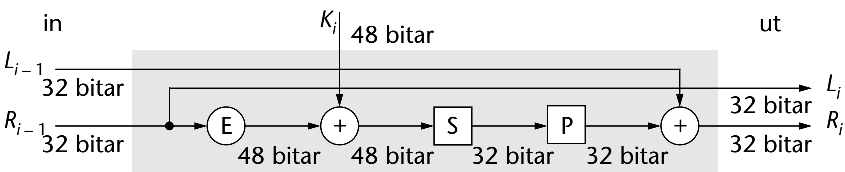
Därefter följer den del, som med 16 Feistel[47]-iterationer eller *rundor*, utför huvuddelen av krypteringsalgoritmen.

Varje runda utförs i en standardiserad krets, standardbyggblock – se Figur 2.30 och funktionen *f* i Figur 2.29. Sist kommer en slutpermutation som är inversen av initialpermutationen.

47 Horst Feistel 1915-1990, tysk kryptograf utvandrad till USA 1934, i husarrest 10 år, amerikansk medborgare 1944, förgrundsgestalt inom modern blockkryptering.



Figur 2.29 DES.



Figur 2.30 Standardbyggblock (SBB) som utför funktionen f – se Figur 2.29.

Efter initialpermutationen delas 64-bitsblocket upp i två 32-bits-block, L_0 (Left) och R_0 (Right). Vid varje runda förs det högra (R_i) 32-bitsordet till funktionen f varefter resultatet adderas (modulo-2) till det vänstra 32-bitsordet (L_i). Inför nästa runda byter de två orden plats och proceduren upprepas.

Funktionen f är nyckeloberoende och består av fyra steg enligt

1. **Expansion**. 32-bitsingångsordet expanderas först till 48 bitar genom att duplicera och flytta om hälften av bitarna enligt Tabell 2.7, d.v.s.ingångsbitarna $x_1, x_2, x_3 \ldots, x_{32}$ expanderas till utgångsbitarna $x_{32}, x_1, x_2, \ldots, x_1$.

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

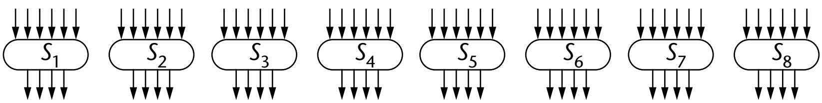
Tabell 2.7 Expansionstabell (E-tabell).

2. **Nyckelblandning**. Det expanderade ordet adderas (modulo-2) med en *rund nyckel* (eng. round key) enligt

$$(R_{i-1})_E \oplus K_i = B_1, B_2, \ldots, B_8 \tag{2.65}$$

genom att välja ut 48 bitar från en hemlig 56-bitars nyckel. I varje ny runda väljs en ny nyckel.

3. Substitution. Var och en av de åtta sex-bitsblocken, B_j, substitueras i åtta parallella (6 in, 4 ut) S-boxar – se Figur 2.31 – och som har samma speciella struktur – se Tabell 2.8.



Figur 2.31 S-boxar i funktionen f.

rad								kolumnnummer									
nr	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S_2
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	17	11	4	2	8	S_3
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S_4
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S_5
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S_6
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	0	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	S_7
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S_8
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

Tabell 2.8 Valfunktioner i S-boxar.

Varje rad i S-boxtabellen består av en permutation av 4-bitsvärdena 0, …, 15. Ingångsordet, $B_j = b_1, b_2, b_3, b_4, b_5, b_6$, substitueras på följande sätt: Heltalet motsvarande b_1b_6 används för att välja tabellens radnummer, medan heltalet motsvarande $b_2b_3b_4b_5$ används för att välja kolumnnumret. Om exempelvis B_1 = 110111 returnerar S_1 värdet i rad 3 och kolumn 11 som är heltalet 14 som representeras av det binära talet 1110. S-boxarna står för kärnan av säkerheten i DES. Utan dem skulle chiffret vara linjärt och i praktiken trivialt att knäcka.

4. **Permutation**. Det resulterande 32-bitsblocket ut från S-boxen permuteras därefter med användande av P-tabellen – se Tabell 2.9 – så att bitarna $x_1, x_2, \ldots, x_{32}$ permuteras till $x_{16}, x_7, \ldots, x_{25}$. Slutligen modulo-2-summeras 32-bitsresultatet med de 32 vänstra ingångsbitarna, L_{i-1}.

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

Tabell 2.9 Permutationstabell (P-tabell).

Efter 16 rundor flyttas data om enligt den slutliga inverspermutationen – se Tabell 2.10.

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

Tabell 2.10 Slutlig inverspermutation.

2.17.1.1 Nyckelval

Även valet av nyckel genomlöper 16 iterationer – se Figur 2.29. Ingångsnyckeln har 64 bitar med 8 paritetsbitar jämnt fördelade till positionerna 8, 16, 24, 32, 40, 48, 56 och 64. I blocket permutationsval 1 elimineras de åtta paritetsbitarna medan resterande 56 bitar permuteras enligt Tabell 2.11.

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

Tabell 2.11 Permutationsval 1.

De 56 bitarna ut från blocket permutationsval 1 uppdelas i de två 28-bitsblocken C och D. Vid var och en av de 16 följande nyckeliterationerna cirkulärskiftas bitarna i C- och D-blocken var för sig ett eller två steg åt vänster enligt

$$C_i = \mathrm{L}S_i(C_{i-1}) \text{ och } D_i = \mathrm{L}S_i(D_{i-1}) \qquad (2.66)$$

där $\mathrm{L}S_i$ anger antal vänsterskift – se Tabell 2.12.

iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
antal vänsterskift	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Tabell 2.12 Vänsterskift vid nyckeliteration.

Från var och en av de i blocken C_i och D_i vänsterskiftade bitarna utväljs 24 bitar för att bilda 48 delnyckelbitar (eng. subkey bits) som permuteras i blocket permutationsval 2 – se Figur 2.29 och Tabell 2.13.

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

Tabell 2.13 Permutationsval 2.

2.17.1.2 Dekryptering

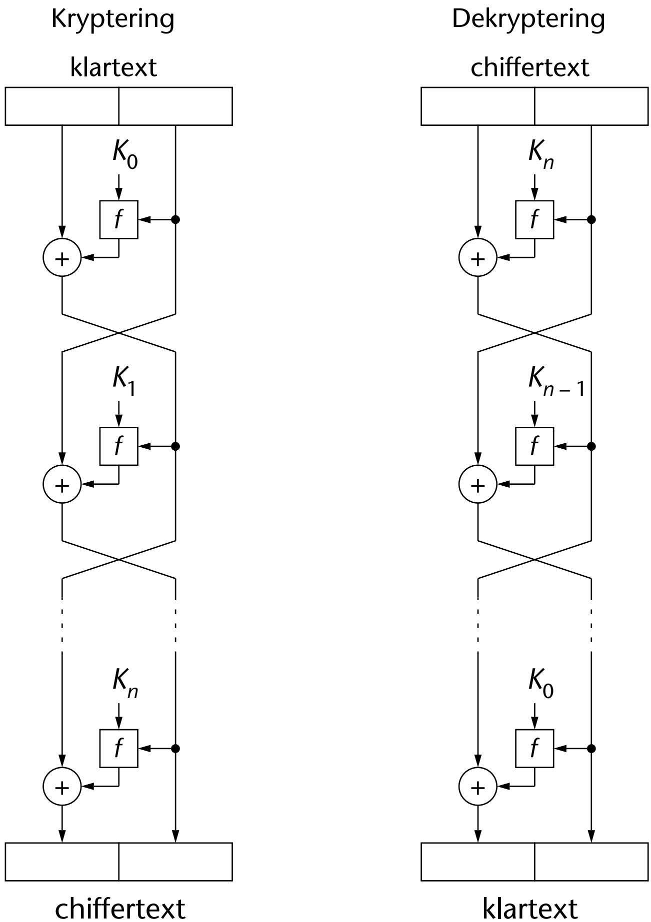
DES är ett exempel på ett kryptosystem med *symmetrisk nyckelalgoritm* (eng. symmetric-key algorithm) eftersom det använder samma nyckel vid kryptering och dekryptering. Feistelstrukturen gör dessutom att kryptering och dekryptering är processer som liknar varandra mycket. Den enda skillnaden är att delnycklarna används i omvänd ordning vid dekryptering – se Figur 2.32.

2.17.1.3 Arbetsmoder för DES

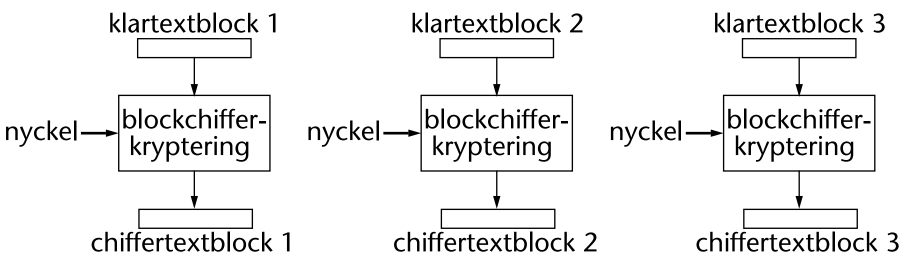
Fyra olika arbetsmoder har utarbetats för DES och de standardiserades i FIPS (the Federal Information Processing Standards) Publication 81 i December 1980.
De fyra arbetsmoderna är:

- Electronic CodeBook mode (ECB mode).
- CipherBlock Chaining mode (CBC mode).
- Output FeedBack mode (OFB mode).
- Cipher FeedBack mode (CFB mode).

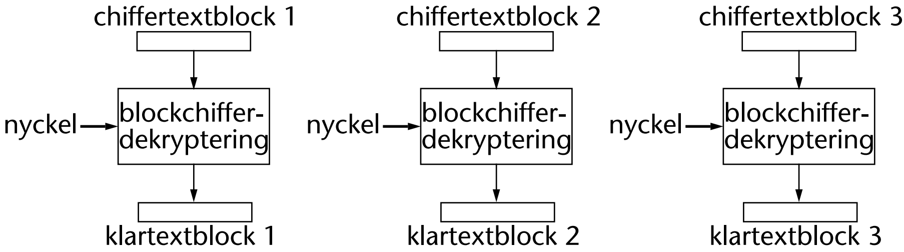
Vid ECB-moden, som är den enklaste krypteringsmoden, delas meddelandet i två block som vartdera krypteras separat.



Figur 2.32 Kryptering och dekryptering med Feistel-nät.

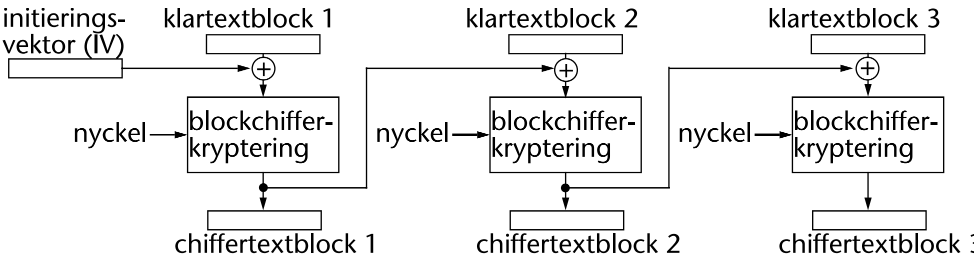


Figur 2.33 ECB-mode-kryptering.

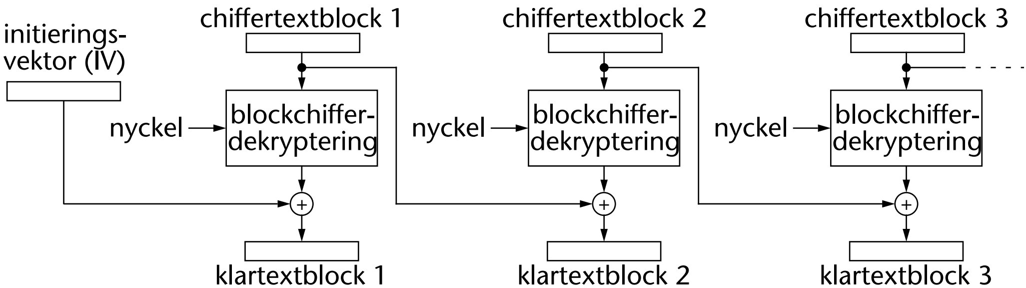


Figur 2.34 ECB-mode-dekryptering.

I CBC mode adderas (modulo-2) varje klartextblock med föregående chiffertextblock innan kryptering sker. På det viset blir varje chiffertextblock beroende av alla föregående klartextblock. Vid kryptering av det första klartextblocket finns inget föregående chiffertextblock varför man istället använder en *initieringsvektor* (eng. Initialization Vector, IV) – se Figur 2.35. Även vid dekryptering börjar man med en initieringsvektor – se Figur 2.36.



Figur 2.35 CBC-mode-kryptering.



Figur 2.36 CBC-mode-dekryptering.

I OFB och CFB mode genereras en nyckelström som adderas (modulo-2) till klartexten (d.v.s. den fungerar som strömchiffer – se avsnitt 2.10). OFB-mode fungerar i själva verket som ett synkront strömchiffer där nyckelströmmen genereras genom att upprepade gånger kryptera en 64-bits initieringsvektor. I CFB-mode utgår man från en 64-bits initieringsvektor och genererar en nyckelström genom att kryptera föregående chiffertextblock.

De fyra arbetsmoderna har olika för- och nackdelar. En uppenbar svaghet hos ECB-moden är att kryptering av identiska klartextblock alltid ger likadana chiffertextblock. Detta är särskilt illa om de krypterade meddelandeblocken kommer från en "låg-entropi-klartext" t.ex. innehållande klartextblock med 64 nollor eller 64 ettor. ECB-kryptering är ju då i praktiken meningslös.

I ECB och OFB-moderna förorsakar ändring av ett 64-bitsblock att motsvarande chiffertextblock ändras medan övriga chiffertextblock förblir opåverkade. Detta kan vara en önskvärd egenskap i vissa situationer om kommunikationskanalen är otillförlitlig. OFB-moden används p.g.a. denna egenskap ofta vid kryptering av satellitkommunikation.

Vid CBC- och CFB-moderna sker däremot en ändring av alla följande chiffertextblock vid ändring av ett klartextblock. Denna egenskap gör att CBC- och CFB-moderna är användbara för autenticering (autenticitetskontroll), d.v.s. dessa moder kan användas för att producera *meddelandeautenticeringskod* (eng. Message Authentication Code, MAC).

2.17.2 AES detaljer

I motsats till sin föregångare, DES, är AES ett substitutions-permutations-nät (SPN), inte ett Feistel-nät. AES är snabb i såväl mjuk- som hårdvara, lätt att implementera och kräver lite minne.

AES är ett itererat kryptosystem med blocklängden 128 bitar. Antalet rundor, Nr, beror av nyckellängden, Nb, och Nr = 10 om nyckellängden är 128 bitar, Nr = 12 om nyckellängden är 192 bitar och Nr = 14 om nyckellängden är 256 bitar.

Det finns en liten skillnad mellan AES och Rijndael (men systemen är i praktiken utbytbara). Skillnaden är att Rijndael kan ha alla nyckel- och blocklängder som är multiplar av 32 bitar mellan 128 och 256 bitar.

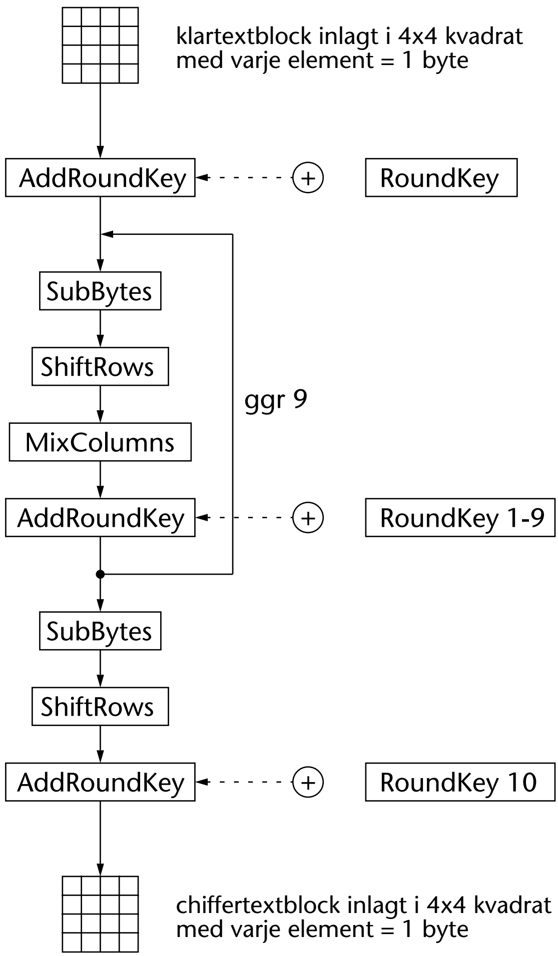
I AES uppdelas varje block i en 4x4 bytes kvadrat kolumnvis uppifrån och ner, vänster till höger. Operationer utförs på varje sådan kvadrat eller State.

AES-algoritmen utförs på följande sätt:

- En klartext, *x*, initierar State att vara *x* och utför operationen AddRoundKey som innebär att RoundKey adderas modulo-2 med State.
- Varje runda utom den sista (Nr – 1) består av fyra steg:

1. SubBytes på State är ett olinjärt substitutionssteg vid vilket varje byte ersätts av en annan enligt en lookup table.
2. ShiftRows på State utgör en permutation där elementen i varje rad i State skiftas cykliskt ett visst antal steg.
3. MixColumns på State kombinerar de fyra elementen i varje kolumn med hjälp av en linjär transformation.
4. AddRoundKey på State. Varje element i State kombineras med RoundKey. Varje RoundKey genereras av CipherKey med hjälp av en nyckellista (eng. KeySchedule). Den sista rundan utesluter detta steg.

- Utför SubBytes på State.
- Utför ShiftRows på State.
- Utför AddRoundKey på State.



Figur 2.37 AES-algoritmen med 128 bitars nyckel.

2.17.2.1 Beskrivning av stegen i AES

SubBytes

Till skillnad mot S-boxarna i DES, som utför pseudoslumpsubstitutioner, kan S-boxarna i AES definieras algebraiskt[48]. Operationen SubBytes utför en substitution på varje byte i State oberoende av de andra med hjälp av en S-box enligt:

1. Först utförs den multiplikativa inversen av varje klartextbyte i GF(2^8) modulo-$m(x)$ där $m(x)$ är det irreducibla polynomet $x^8 +$ $+ x^4 + x^3 + x + 1$. (Alla additioner utförs modulo-2). Klartextbytes 00000000 (00 i hexadecimal form) har ingen invers och avbildas på sig själv (00 → 00).

Exempel 2.12

Bestäm den multiplikativa inversen, modulo-$m(x) = x^8 + x^4 + x^3 + x$ $+ 1$, till klartextblocket 01010011 (53 i hexadecimal form) representerat av det moniska polynomet[49] $a(x) = x^6 + x^4 + x + 1$.

Lösning:

Använd den *utökade euklidiska algoritmen* på $a(x)$ och $m(x)$. Algoritmen utsäger att det finns polynom $a^{-1}(x)$ och $s(x)$ så att

$$a^{-1}(x) \cdot a(x) + s(x) \cdot m(x) = 1 \tag{2.67}$$

Efter att ha reducerat ekv. (2.67) modulo-$m(x)$ får vi

$$a^{-1}(x) \cdot a(x) = 1 \text{ modulo-}m(x) \tag{2.68}$$

där $a^{-1}(x)$ är den multiplikativa inversen till $a(x)$ modulo-$m(x)$.

Börja med att beräkna $\dfrac{m(x)}{a(x)}$:

$$\frac{m(x)}{a(x)} = x^2 + 1 \quad \text{rest } x^2 \text{ eller}$$

48 Mer om algebra i kapitel 3.
49 Moniska polynom har koefficienterna 0 eller 1.

$$m(x) = a(x)\overbrace{\left(x^2 + 1\right)}^{k_1(x)} + \overbrace{x^2}^{r_1(x)}$$

Fortsätt sedan att beräkna kvoter, k_i, och rester, r_i, tills den sista resten blir 1 enligt nedanstående mönster:

$$a(x) = \overbrace{x^2}^{r_1(x)}\overbrace{\left(x^4 + x^2\right)}^{k_2(x)} + \overbrace{(x + 1)}^{r_2(x)}$$

$$\overbrace{x^2}^{r_1(x)} = \overbrace{(x + 1)}^{r_2(x)}\overbrace{(x + 1)}^{k_3(x)} + \overbrace{1}^{r_3(x)}$$

Därefter arbetar man baklänges på följande sätt:

$$1 = \overbrace{x^2}^{r_1(x)} + \overbrace{(x + 1)}^{r_2(x)}(x + 1)$$

$$1 = \overbrace{x^2}^{r_1(x)} + \left[a(x) + \overbrace{x^2}^{r_1(x)}\left(x^4 + x^2\right)\right](x + 1)$$

$$1 = m(x) + a(x)\left(x^2 + 1\right) + \left[a(x) + \left(m(x) + a(x)\left(x^2 + 1\right)\right)\left(x^4 + x^2\right)\right](x + 1)$$

$$1 = m(x)\left(x^5 + x^4 + x^3 + x^2 + 1\right) + a(x)\left(x^7 + x^6 + x^3 + x\right)$$

Reducera modulo-$m(x)$:

$$\left(x^7 + x^6 + x^3 + x\right)a(x) = 1$$

$$\Rightarrow a^{-1}(x) = \left(x^7 + x^6 + x^3 + x\right)$$

Man kan undvika ”baklängesarbetet” genom att hålla reda på några hjälpkvantiteter när man utför sina euklidiska algoritmberäkningar:

rest	kvot	hjälpkvantitet
$x^8 + x^4 + x^3 + x + 1$		0
$x^6 + x^4 + x + 1$		1
x^2	$x^2 + 1$	$x^2 + 1$
$x + 1$	$x^4 + x^2$	$x^6 + x^2 + 1$
1	$x + 1$	$x^7 + x^6 + x^3 + x$

Tabell 2.14

Kolumnen ”hjälpkvantitet” startar alltid med 0 och 1. ”rest”-kolumnen börjar alltid med $m(x)$ och $a(x)$. För att fylla i varje följande rad divideras resterna i de två föregående raderna varefter den bildade kvoten och resten förs in i kvot- respektive restkolumnen. Därefter multipliceras värdet i kvotkolumnen med värdet på föregående rad i ”hjälpkvantitet”-kolumnen varefter produkten i ”hjälpkvantitet”-kolumnen en nivå ovanför adderas och summan förs in i ”hjälpkvantitet”-kolumnen. När resten reducerats till 1 är innehållet i ”hjälpkvantitet”-kolumnen = $a^{-1}(x)$, d.v.s. inversen av $a(x)$.

I binär form: 11001010

I hexadecimal form: CA

◊

2 Därefter utförs en affin transformation[50] (över GF(2)) definierad av

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \tag{2.69}$$

50 En affin transformation är varje transformation som bevarar kolinjäritet (alla punkter på en linje ligger fortfarande på en linje efter transformationen) och avståndsförhållande (mittpunkten på ett linjesegment förblir mittpunkt efter transformationen).

Exempel 2.13

Utför en affin transformation, enligt pkt. 2 ovan, på $a^{-1}(x) = x^7 + x^6 + x^3 + x$ eller i binär form: 11001010.

Lösning:

Bilda kolumnvektorn x_0 - x_7 för $a^{-1}(x) = x^7 + x^6 + x^3 + x$:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Utför därefter matrismultiplikationen och addera den sista kolumnvektorn i uttrycket (2.69):

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

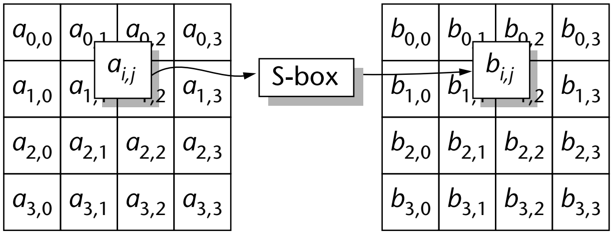
$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

I binär form: 11101101

I hexadecimal form: ED

◊

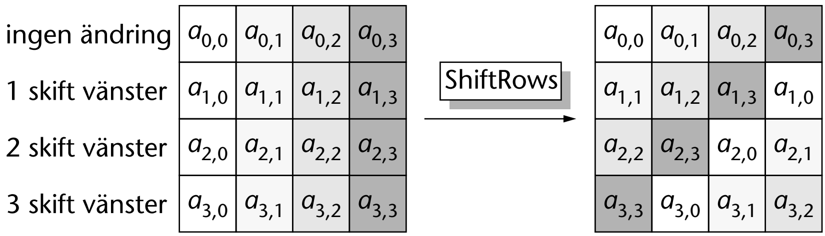
SubBytes-operationen kan schematiskt illustreras som i Figur 2.38, där värdena i S-boxen (lookup table) ges av Tabell 2.15.



Figur 2.38 Illustration av SubBytes-operationen.

	Y															
X	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DE
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Tabell 2.15 S-box (lookup table med talet XY in) för AES.



Figur 2.39 ShiftRows. Beroende på rad skiftas varje byte 0, 1, 2 eller 3 steg cykliskt åt vänster.

ShiftRows

ShiftRows arbetar på rader av State enligt Figur 2.40. På detta sätt kommer varje kolumn i utgångstillståndet i ShiftRows-steget att bestå av bytes från varje kolumn i ingångstillståndet.

MixColumns

I MixColumns-steget multipliceras kolumnerna, angivna som polynom över GF(2^8), i ingångstillståndet med $c(x)$ modulo-($x^4 + 1$) där

$$c(x) = \text{'}03\text{'}x^3 + \text{'}01\text{'}x^2 + \text{'}01\text{'}x + \text{'}02\text{'} \tag{2.70}$$

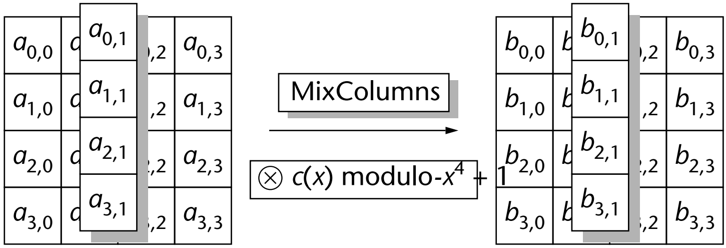
I uttrycket (2.70) är koefficienterna angivna med hexadecimala siffror.

Eftersom $c(x)$ och $x^4 + 1$ är relativt prima är $c(x)$ inverterbar.

Multiplikationen kan skrivas som en matrismultiplikation:

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{2.71}$$

Tillsammans med ShiftRows åstadkommer MixColumns spridning i chiffret – se avsnitt 2.9.1 om spridning.



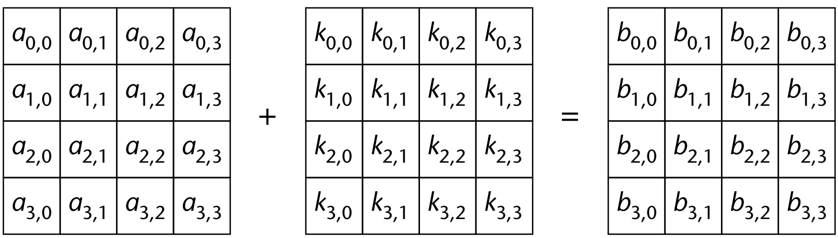
Figur 2.40 I MixColumns multipliceras varje kolumn med ett fixt polynom c(x).

AddRoundKey

I denna operation adderas, modulo-2, en RoundKey till State, se Figur 2.41. RoundKey erhålls av CipherKey från en key schedule (sv. nyckellista). RoundKey-längden är densamma som blocklängden Nb.

KeySchedule

Beskrivningen anger hur en KeySchedule för en 10-rundors version av AES konstrueras. En 10-rundors AES använder en 128-bits RoundKey. Det åtgår 11 RoundKeys, var och en med 16 byte. Algoritmen för KeySchedule är *ord*orienterad (ett ord består av 16 byte eller 32 bitar). Varje RoundKey består av fyra ord. Sammanlänkningen av alla RoundKeys kallas ExpandedKey och består av 11x4 = 44 ord. ExpandedKey anges som $w[0]$, …, $w[43]$, där varje $w[i]$ är ett ord.



Figur 2.41 I AddRoundKey adderas (modulo-2) varje byte i State med en byte från RoundKey.

ExpandedKey konstrueras med hjälp av operationen KeyExpansion[51].

2.17.2.2 Dekryptering av AES

För att dekryptera AES måste alla ovan beskrivna operationer utföras i omvänd ordning. Även KeySchedule skall utföras i omvänd ordning. Vidare måste operationerna ShiftRows, SubBytes och MixColumns ersättas av sina inversa operationer – se fotnot 51. Operationen AddRoundKey är sin egen invers och behöver följaktligen inte ersättas.

2.17.2.3 Arbetsmoder för AES

AES-moderna använder samma moder som DES-moderna – se 2.17.1.3 men även nya utarbetas[52].

---

51 Se http://csrc.nist.gov/archive/aes/rijndael/Rijndal-ammended.pdf
52 se mer om AES-moder i NIST (National Institute of Standards and Technology) och FIPS (Federal Information Processing Standards)

3 Algebra för kryptografi

3.1 Gruppbegreppet

En mängd G bestående av ett antal element sägs vara en grupp om vissa samband mellan elementen existerar.

Definition 3.1 Gruppbegreppet

En binär operation $\star$ på G är en regel, som för varje par element a och b på ett unikt sätt tillordnar ett element $c = a \star b$, som också är element i G.

När en sådan operation definieras på G, sägs G vara sluten under $\star$.

Följande binära operationer måste gälla för att G skall vara en grupp:

1. Associativitet: $(a \star b) \star c = a \star (b \star c)$ för

   $\forall\ a, b, c \in G$.

2. Identitet: Det finns ett identitetselement e i G så att

   $a \star e = e \star a = a$ för $\forall\ a \in G$.

3. För varje $a \in G$ finns ett unikt inverselement a^{-1} så att

   $a \star a^{-1} = e$.

Gruppen kan dessutom ha egenskapen att vara ”kommutativ” eller ”abelsk” om det för varje par a, b gäller att $a \star b = b \star a$.

Om operationen $\star$ är ”+” sägs gruppen vara en additiv grupp. Om operationen $\star$ är ”·” sägs gruppen vara en multiplikativ grupp.

Exempel 3.1

Exempel på grupper

Mängden av alla heltal är en oändlig kommutativ grupp under addition. Identitetselementet är 0. För varje heltal i finns ett inverst heltal $-i$ så att $i + (-i) = 0$. Detta gäller inte under multiplikation eftersom det då inte finns en multiplikativ invers.

Mängden av alla rationella tal utom 0 formar en oändlig kommutativ grupp under multiplikation. Identitetselementet är 1 och för varje rationellt tal, p/q, där p och q är heltal, finns ett inverst rationellt tal q/p så att $(p/q) \cdot (q/p) = 1$.

◊

Exemplen visar grupper med oändligt antal element. Det finns också grupper med ändligt antal element.

Exempel 3.2

Mängden G = {0,1}

Definiera en binär operation $\oplus$ ”modulo-2 addition” på följande sätt:

$0 \oplus 0 = 0$; $0 \oplus 1 = 1$; $1 \oplus 0 = 1$; $1 \oplus 1 = 0$

Mängden $G = \{0,1\}$ är en grupp under modulo-2 addition

$\oplus$ är associativ

0 är identitetselement

0 är sin egen invers

1 är sin egen invers

$\because$ Mängden G är en kommutativ grupp under modulo-2 addition.

◊

Antalet element i en mängd är mängdens kardinaltal. Antalet element i en grupp är gruppens ordning.

$\Rightarrow$ kardinaltal = ordning.

För kryptografi är ändliga grupper mest intressanta. En av de enklaste metoderna att konstruera ändliga grupper är m.hj.a. modulär aritmetik på heltal.

Definition 3.2 Addition modulo-*m*

Addition modulo-m uttrycks på följande sätt:

$a + b = c$ modulo-m

som fås genom att addera a och b på vanligt sätt och därefter dividera resultatet med m; c är den uppkomna positiva resttermen.

Exempel 3.3

13 + 18 = 11 modulo-20

9 + 5 = 4 modulo-10

18 + 27 = 0 modulo-5

3 + 8 = 11 modulo-20

2 + 4 = 0 modulo-6

15 + 100 = 50 modulo-65

◊

Addition modulo-m grupperar den oändliga heltalsmängden i m distinkta ekvivalensklasser. Två heltal a och b är i samma ekvivalensklass modulo-m om a kan skrivas som $a = x \cdot m + b$ för något heltal x.

Exempel 3.4

Om $m = 2$ får vi två ekvivalensklasser, den ena bestående av alla jämna tal och den andra av alla udda tal.

◊

Varje element i en ekvivalensklass modulo-m är ekvivalent på så sätt att det kan utbytas mot ett annat element i samma ekvivalensklass utan att ändra resultatet av modulo-m operationen.

Ekvivalensklasser bestående av heltal anges vanligen med sitt minsta icke-negativa tal.

Exempel 3.5

Ekvivalensklasser under modulo-5 addition

etikett	ekvivalensklass
0	{…,– 20,– 15,– 10,– 5,0,5,10,…}
1	{…,– 19,– 14,– 9,– 4,1,6,11,.....}
2	{…,– 18,– 13,– 8,– 3,2,7,12,.....}
3	{…,– 17,– 12,– 7,– 2, 3, 8,13,....}
4	{…,– 16,– 11,– 6,– 1,4,9,14,......}

◊

I fortsättningen refereras till en ekvivalensklass genom dess etikett.

Teorem 3.1 Ekvivalensklasserna {0,1,2,3,…,*m* – 1}

formar en kommutativ grupp av m:te ordningen under modulo-m heltalsaddition för varje positivt heltal m.

Exempel 3.6

Gruppen med ordning 7 under modulo-7 addition

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

Multiplikation modulo-m över heltal utförs på liknande sätt som addition modulo-m d.v.s. resultatet av multiplikationen divideras med m och den positiva resten behålls som resultatet av den modulära operationen.

Exempel 3.7

$11 \cdot 5 = 15$ modulo-20

$9 \cdot 7 = 13$ modulo-50

$4 \cdot 17 = 0$ modulo-2

$6 \cdot 4 = 6$ modulo-18

$2 \cdot 5 = 0$ modulo-10

$6 \cdot 5 = 0$ modulo-6

◊

Den avgörande skillnaden mellan modulär addition och modulär multiplikation är att den senare **inte** kan skapa en ändlig grupp av heltalen och en godtycklig modul.

Exempel 3.8

{1,2,3,4,5,6,7} under modulo-8 heltalsmultiplikation

Eftersom $2 \cdot 4 = 0$ modulo-8 och 0 inte ingår i mängden är operationen inte sluten över mängden och vi har inte en grupp. Även om vi låter 0 ingå i mängden får vi ändå inte en grupp eftersom 0 inte har en multiplikativ invers, d.v.s. det finns inget x så att $0 \cdot x = 1$ modulo-8. Problemet ligger i att mängden {1,2,3,4,5,6,7} har åtskilliga nolldivisorer. En nolldivisor utgörs av ett godtyckligt nollskilt tal a för vilket det finns ett nollskilt b så att $a \cdot b = 0$ modulo-m.

◊

I allmänhet

Om modulen m har andra faktorer än 1 i en given mängd så har denna mängd nolldivisorer under modulo-m multiplikation.

Slutsats

För att kunna konstruera en multiplikativ analogi till det förut givna Teorem 3.1 måste vi begränsa våra moduler m till primtal.

Teorem 3.1 Elementen $S = \{1,2,3,...,p - 1\}$

formar en grupp under modulo-p multiplikation omm (= om och endast om) p är ett primtal.

Exempel 3.9

Gruppen av ordning 6 under modulo 7 multiplikation

·	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

◊

På samma sätt som en grupp har en ordning har vart och ett av elementen i gruppen en ordning. Dessa två ordningsbegrepp, gruppens ordning och ett elements ordning är besläktade men inte lika.

Definition 3.3 Ordningen, ord(g), för ett gruppelement g

g är ett element i gruppen G med gruppoperationen ”·”. Ordningen för g är det minsta heltal, ord(g), så att $g^{\mathrm{ord}(g)}$ är gruppens identitetselement.

Exempel 3.10

Gruppen av ordning 6 under modulo-7 multiplikation

·	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

◊

Låt S vara en delmängd av gruppen G. S är en delgrupp till G om S är sluten under de gruppoperationer som gäller för G och i övrigt satisfierar alla gruppvillkor. En delgrupp är därmed också en grupp.

En delgrupp S är en äkta delgrupp till G om $S \subset G$ men $S \neq G$.

Exempel 3.11

Heltalsgruppen under modulo-9 addition innehåller de äkta delgrupperna {0} och {0,3,6}.

Heltalsgruppen under modulo-16 addition innehåller de äkta delgrupperna {0},{0,8}, {0,4,8,12} och {0,2,4,6,8,10,12,14}.

◊

3.2 Ringar

En ring är en samling element R med två binära operationer "+" och "·" definierade och med följande egenskaper:

1. R utgör en kommutativ additiv grupp. Det additiva identitets-elementet är "0".
2. Multiplikationsoperationen är associativ:

   $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ för $\forall\ a, b, c \in R$
3. Multiplikationsoperationen distribueras över additionsoperationen

   $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
4. En ring är en kommutativ ring om

   $a \cdot b = b \cdot a$ (multiplikationsoperationen är kommutativ)
5. En ring är en ring med identitet om

   multiplikationsoperatorn har ett identitetselement "1".

Exempel 3.12

Matriser med heltalselement utgör en ring med identitet vid normal matris-addition och -multiplikation. Identitetsmatrisen ger egenskapen multiplikativ identitet. Matrismultiplikation är i allmänhet inte kommutativ ⇒ vi har en icke-kommutativ ring med identitet.

Exempel 3.13

Heltal under modulo-*m* addition och multiplikation formar en kommutativ ring med identitet.

Exempel 3.14

Mängden av alla polynom med binära koefficienter formar en kommutativ ring under vanlig polynom-addition och -multiplikation. En sådan ring kallas $F_2[x]$ eller GF(2)[x].

◊

3.3 Talkroppar (eng. Fields)

F är en mängd element på vilken vi kan utföra de fyra räknesätten utan att lämna mängden.

Definition 3.4 Talkroppsbegreppet

F är en mängd objekt med operationerna "+" och "·" definierade. *F* är en talkropp om

1. *F* är en additiv kommutativ grupp med det additiva identitetselementet "0".
2. *F* exklusive det additiva identitetselementet "0" är en multiplikativ kommutativ grupp med det multiplikativa identitetselementet "1".
3. Multiplikation distribueras över addition, d.v.s.

   $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$.

En talkropp kan också definieras som en kommutativ ring med identitetselement i vilken varje element har en multiplikativ invers.

Exempel 3.15

Olika oändliga talkroppar

De rationella talen utgör en av de minsta oändliga talkropparna (kardinaltal $\aleph_0$, alef-noll).

De reella talen liksom de komplexa utgör oändliga talkroppar.

Heltalen utgör **inte** en talkropp eftersom de flesta heltalen inte har en multiplikativ invers.

◊

Talkroppar med ändlig ordning (= kardinaltal) är speciellt intressanta för kryptografi. Ändliga talkroppar upptäcktes av Évariste Galois och kallas också Galois-talkroppar. En Galois-talkropp av ordning *q* betecknas GF(*q*).

Den enklaste Galois-talkroppen är GF(2) som t.ex. kan representeras av mängden {0,1} med vanlig binär addition och multiplikation.

+	0	1
0	0	1
1	1	0

·	0	1
0	0	0
1	0	1

Galois-talkroppar av ordningen p där p är ett primtal kan konstrueras med hjälp av Teorem 3.1 och Teorem 3.1. Heltalsmängden $\{1,2,\ldots,p-1\}$ formar en multiplikativ kommutativ grupp modulo-p och heltalsmängden $\{0,1,2,\ldots,p-1\}$ formar en additiv kommutativ grupp modulo-p. Vid heltalsaritmetik distribueras multiplikation över addition och vi har därför enligt Definition 3.4.3 en talkropp.

Exempel 3.16

Använd heltalsmängden {0,1,2} under modulo-3 addition och multiplikation. Multiplikativ distribution över addition är säkrad genom att sätta $0 \cdot a = 0$ för $a \in GF(3)$.

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

·	1	2
1	1	2
2	2	1

◊

Ändliga talkroppar av primtalsordning är lätt att konstruera. Det finns även andra ordningar, som ger ändliga talkroppar, men q i GF(q) får inte vara ett godtyckligt heltal, endast $q = p^m$ där p är ett primtal och m ett heltal är möjliga. GF(q) = GF(p^m) är en utvidgningstalkropp (eng. extension field) till GF(p).

För att konstruera talkroppar av ordning p^m måste andra mer komplexa metoder än moduloräkning användas.

3.3.1 Grundläggande egenskaper hos Galois-talkroppar

En Galois-talkropp är fullständigt specificerad och därmed unik genom sin ordning.

Exempel 3.17

Låt β vara ett element i GF(q) medan 1 är det multiplikativa identitetselementet.

Eftersom GF(q) är sluten under "·", finns följande element i GF(q)

$1, \beta, \beta^2, \beta^3, \beta^4, \ldots$

men antalet element är ändligt $\Rightarrow$ någonstans uppstår en repetition och det första element som repeteras är 1.

◊

Definition 3.5 Ordningen hos ett talkroppselement

Om β är ett element i GF(q) så är ordningen hos β, ord(β) = det minsta positiva heltal m som ger $\beta^m = 1$.

Anm.: Denna definition är densamma som ordningen för ett gruppelement men den definierade ordningen gäller den multiplikativa operationen, inte den additiva.

Definition 3.6 Primitivt element

Ett element med ordningen (q – 1) i GF(q) kallas ett primitivt element i GF(q).

3.3.2 Den multiplikativa strukturen hos Galois-talkroppar

Teorem 3.1 Ordningen för varje element $\boldsymbol{\beta} \in \mathrm{GF}(\boldsymbol{q})$

är delare till $q - 1$, eller $\mathrm{ord}(\beta)|(q - 1)$.

Definition 3.7 Eulers $\boldsymbol{\phi}$-funktion

Antalet element med $\mathrm{ord}(\beta)$, där $\beta \in \mathrm{GF}(q)$, kan beräknas med hjälp av Eulers ϕ-funktion som betecknas $\phi(t)$:

$$\phi(\mathrm{ord}(\beta)) = \mathrm{ord}(\beta) \prod_{p|\mathrm{ord}(\beta)} \left(1 - \frac{1}{p}\right) \tag{3.1}$$

där produkten utförs över alla positiva heltal $p < \mathrm{ord}(\beta)$ där $p|\mathrm{ord}(\beta)$ och $\phi(1) = 1$.

Av Definition 3.7 följer att:

- Om p är ett primtal $\Rightarrow \phi(p) = p - 1$ eftersom alla nollskilda element är relativt prima till ett primtal.
- Om p_1 och p_2 är två olika primtal $\Rightarrow \phi(p_1 \cdot p_2) = \phi(p_1) \cdot \phi(p_2) =$ $= (p_1 - 1)(p_2 - 1)$.
- Om p är ett primtal $\Rightarrow \phi(p^m) = p^{m-1}(p - 1)$.
- Om p_1 och p_2 är två olika primtal $\Rightarrow$ $\phi\left(p_1^m \cdot p_2^n\right) = p_1^{m-1} \cdot p_2^{n-1}(p_1 - 1)(p_2 - 1)$

Exempel 3.18 Elementens ordning i GF(11)

Enligt Teorem 3.1 är de förekommande ordningarna till elementen i GF(11) delare till 10, d.v.s. 1, 2, 5 och 10.

Antalet element med ordning 1 är $\phi(1) = 1$,

antalet element med ordning 2 är, eftersom 2 är ett primtal, $\phi(2) =$ $= 2 - 1 = 1$,

antalet element med ordning 5 är, eftersom 5 är ett primtal, $\phi(5) =$ $= 5 - 1 = 4$,

antalet element med ordning 10 är

$\phi(10) = 10 \cdot \prod_{p|10} \left(1 - \frac{1}{p}\right) = 10\left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{5}\right) = 4$, d.v.s. det finns 4 primitiva

element i GF(11).

En undersökning visar att:

elementet med ordning 1 är 1,

elementet med ordning 2 är 10,

elementen med ordning 5 är 3, 4, 5 och 9,

elementen med ordning 10 är 2, 6, 7 och 8.

Om vi istället kallar elementen i GF(11)

$\{0,1,\beta,\beta^2,\beta^3,\beta^4,\beta^5,\beta^6,\beta^7,\beta^8,\beta^9\}$, får vi att

elementet med ordning 1 är 1,

elementet med ordning 2 är β^5,

elementen med ordning 5 är β^2, β^4, β^6och β^8,

elementen med ordning 10 är β,β^3,β^7och β^9.

◊

3.3.3 Den additiva strukturen hos Galois-talkroppar

Betrakta elementen i en Galois-talkropp uppkomna genom addition

$0,1,1 + 1,1 + 1 + 1,1 + 1 + 1 + 1,\ldots$

Antalet element är ändligt ⇒ någonstans uppstår en upprepning. Definiera $m(1)$ = summan av m stycken ettor.

Definition 3.8 Talkroppens karakteristik

Karakteristiken för en Galois-talkropp GF(q) är det minsta antal ettor, m, som gör att $m(1) = 0$.

I en Galois-talkropp med karakteristiken p gäller alltså att $p(\alpha) = 0$ för $\forall\ \alpha \in$ GF(q) eftersom $p(\alpha) = p(\alpha \cdot 1) = \alpha \cdot p(1) = \alpha \cdot 0 = 0$.

Om $q \neq p$ gäller att $p^m = q$ och p är ett primtal.

3.3.4 Primitiva polynom och Galois-talkroppar av ordningen p^m

Det som ska visas i detta avsnitt bygger på några redan erhållna resultat, nämligen:

1. GF(q) kan representeras av 0 och $q - 1$ konsekutiva potenser av ett primitivt element $\alpha \in$ GF(q).
2. Multiplikation i en Galois-talkropp, som inte har primtalsordning kan därför göras genom att representera elementen som potenser av α och addera exponenterna modulo-($q - 1$).
3. GF(q) innehåller en delkropp av ordning p vars additiva operation är heltalsaddition modulo-p.

I detta avsnitt ska den additiva operationen utsträckas till att gälla hela GF(q) = GF(p^m). Metoden för detta grundar sig på polynom vars koefficienter tas från GF(q) (i detta fall behöver q inte vara ett primtal men måsta vara en potens av ett primtal).

Vi inför beteckningen GF(q)[x] för alla polynom av godtyckligt gradtal av x med koefficienterna $\{a_i\}$ från den ändliga talkroppen GF(q). Om $q = p^m$ kan man tänka sig olika fall, d.v.s. koefficienterna kan tas från GF(p), GF(p^2),...,GF(p^m) = GF(q).

Exempel 3.19

$(x^5 + x^2) + (x^2 + x + 1) = x^5 + 2x^2 + x + 1$ GF(3)[x]

$(x^5 + x^2) + (x^2 + x + 1) = x^5 + x + 1$ GF(2)[x]

Definition 3.9 Irreducibla polynom

Ett polynom $f(x)$ är irreducibelt i GF(q)[x] om $f(x)$ inte kan faktoriseras i en produkt av lägre grads polynom i GF(q)[x].

Exempel 3.20

Polynom av grad 2 i GF(2)[x], som finns är

$x^2 + x + 1$ irreducibelt, ingen rot

$x^2 + x = x(x + 1)$ rötter: $x = 0$ och $x = 1$

$x^2 + 1 = (x + 1)^2$ dubbelrot: $x = 1$

$x^2 = x \cdot x$ dubbelrot: $x = 0$

men

$x^2 + x + 1 = (x - 1)^2$ i GF(3)[x]

◊

Teorem 3.1 Varje irreducibelt polynom över GF(2)[x]

av grad m är delare till $x^{2^m - 1} + 1$.

Definition 3.10 Primitiva polynom

Ett irreducibelt polynom $p(x) \in \mathrm{GF}(p)[x]$ av grad m kallas primitivt om det minsta heltal n för vilket $p(x)$ är delare till $x^n - 1$ är

$n = p^m - 1$.

Ett primitivt polynom $p(x) \in \mathrm{GF}(p)[x]$ är alltid irreducibelt i GF(p)[x] men irreducibla polynom är inte alltid primitiva

Teorem 3.2 Rötterna $\{a_j\}$ till ett m:te grads primitivt polynom $p(x) \in \mathrm{GF}(p)[x]$ är av ordning $p^m - 1$.

Av detta följer: En rot till $p(x)$ är α. α är av ordning $p^m - 1$. De $p^m - 1$ konsekutiva potenserna av α genererar en multiplikativ grupp som är av ordning $p^m - 1$. Multiplikationsoperationen görs genom att addera potensexponenten modulo-$p^m - 1$. Exponentrepresentationen kan uttryckas genom att reducera sekvensen av potenser modulo-det primitiva polynomet.

Exempel 3.21

Konstruktion av GF(8)

$p(x) = x^3 + x + 1$ är primitivt i GF(2)[x] eftersom det minsta polynom av formen $x^n - 1$ för vilket $p(x)$ är divisor måste vara

$$x^{p^m - 1} - 1 = x^{2^3 - 1} - 1 = x^7 - 1.$$

Låt α vara en rot till $p(x)$. De 8 elementen i GF(8) kan skrivas på följande sätt:

exponentialform	polynomform
0	0
1	1
α	α
α^2	α^2
α^3	$\alpha + 1$
α^4	$\alpha^2 + \alpha$
α^5	$\alpha^2 + \alpha + 1$
α^6	$\alpha^2 + 1$

◊

Vi kan konstatera att polynomrepresentationen av den ändliga kroppen GF(p^m) har koefficienter från grundkroppen GF(p).

GF(p^m) kallas ibland utvidgningstalkropp till GF(p).

På samma sätt som grupper kan innehålla många delgrupper, kan en talkropp GF(p^m) innehålla deltalkroppar GF(p^b), där b är delare till m.

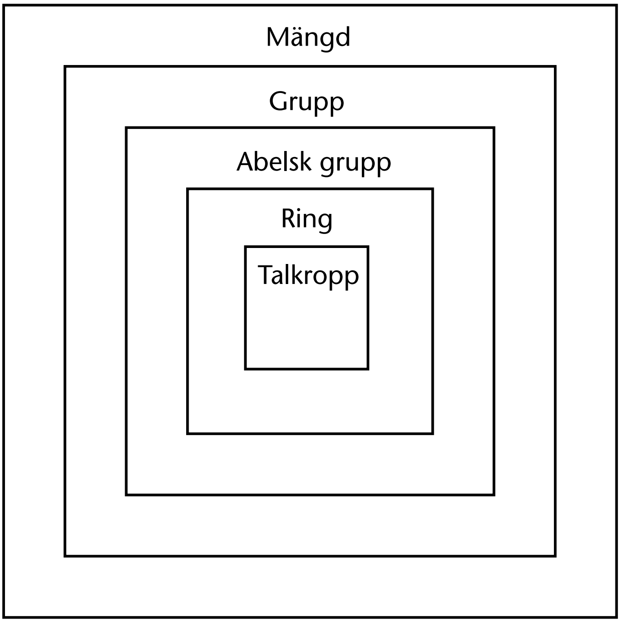
Exempel 3.22

GF(64) = GF(2^6) innehåller GF(2^6), GF(2^3), GF(2^2) och GF(2) som deltalkroppar.

◊

Sammanfattning

- Sambandet mellan de beskrivna algebraiska strukturerna kan åskådliggöras som i Figur 3.1.
- En grupp kan vara additiv eller multiplikativ men inte nödvändigtvis kommutativ.
- En abelsk grupp (= kommutativ grupp), är additiv eller multiplikativ med invers.
- En ring är kommutativ additiv men inte kommutativ multiplikativ.
- En talkropp är både kommutativ additiv och kommutativ multiplikativ, vilket är detsamma som en ring, som dessutom har egenskapen att vara kommutativ multiplikativ, något som kännetecknar en ring med identitet.



Figur 3.1 Samband mellan algebraiska strukturer.

Sakregister