Kryptering av dokument för överföring via telenätet.

(HS-IDA-EA-98-117)

Daniel Molin (a94danmo@ida.his.se)

Institutionen för datavetenskap
Högskolan i Skövde, Box 408
S-54128 Skövde, SWEDEN

Examensarbete på det datavetenskapliga programmet under vårterminen 1998.

Handledare: Henrik Gustavsson

Kryptering av dokument för överföring via telenätet

Examensrapport inlämnad av Daniel Molin till Högskolan i Skövde, för Kandidatexamen (BSc) vid Institutionen för Datavetenskap.

1998-06-11

Härmed intygas att allt material i denna rapport, vilket inte är mitt eget, har blivit tydligt identifierat och att inget material är inkluderat som tidigare använts för erhållande av annan examen.

Signerat: _____

Kryptering av dokument för överföring via telenätet.

Daniel Molin (a94danmo@ida.his.se)

Key words: Encryption, PSTN, Triple-DES, Secure transfer, Key distribution

Abstract

The objective of this project is to find a suitable system for the encryption of electronic documents to be sent over a telephone network. The environment is of a star-like shape and consists of a central document archive in the middle with several users connected to it. The system must be able to provide a very safe transfer of the documents between the user and archive. When choosing an appropriate system for encryption and key distribution, consideration is taken to what the system environment and users of the system demands from it. The system has to be very secure, reliable and easy for the user to use. A number of such systems are evaluated. Finally one system for document encryption and key distribution is chosen and specified. A prototype of the specified chosen encryption algorithm is also implemented.

Innehållsförteckning

Sammanfattning

Allt fler tjänster och arbeten sker med datorers hjälp, inte minst vad beträffar hantering av dokument av olika slag. Det är nuförtiden inte något ovanligt att exempelvis bokföringen i ett företag sker mha en dator. Med andra ord så skapas idag väldigt många olika typer av så kallade elektroniska dokument i vårt samhälle. Likväl som med den äldre typen av dokument är många av dessa väldigt viktiga och behöver således förvaras på ett säkert sätt.

Ett sätt att kunna erbjuda en säker förvaring av elektroniska dokument är att lagra dessa i en arkivcentral som är speciellt designad för att kunna erbjuda en sådan säker förvaring. För att användaren av en sådan arkivcentral skall kunna skicka över respektive hämta sina dokument krävs ett system för dokumentöverföring som kan hantera denna process. Kravet på detta system är att överföringen av dokumenten skall ske på ett säkert och enkelt sätt, dvs informationen får inte gå förlorad samtidigt som ingen obehörig person skall kunna få tillgång till dokumenten. Exempelvis måste kryptering tillämpas på dokumenten innan de skickas över, till eller från arkivcentralen, för att förhindra att någon obehörig person kan läsa det.

Detta examensarbete, på 20p, fokuserar på att finna en lämplig metod för kryptering av de elektroniska dokumenten innan de skickas över till arkivcentralen, vilket innefattar val av lämplig algoritm för kryptering samt precisering av lämpligt system för nyckeldistribution. Mediet för kommunikation mellan användaren och arkivcentral är begränsat till användning av telenätet.

Som underlag till arbetet har studier genomförts mha böcker men även mha information som är hämtad från Internet. Området kryptering förnyas ständigt vilket leder till att de nyaste uppgifterna kan hämtas på antingen Internet eller i tidskrifter.

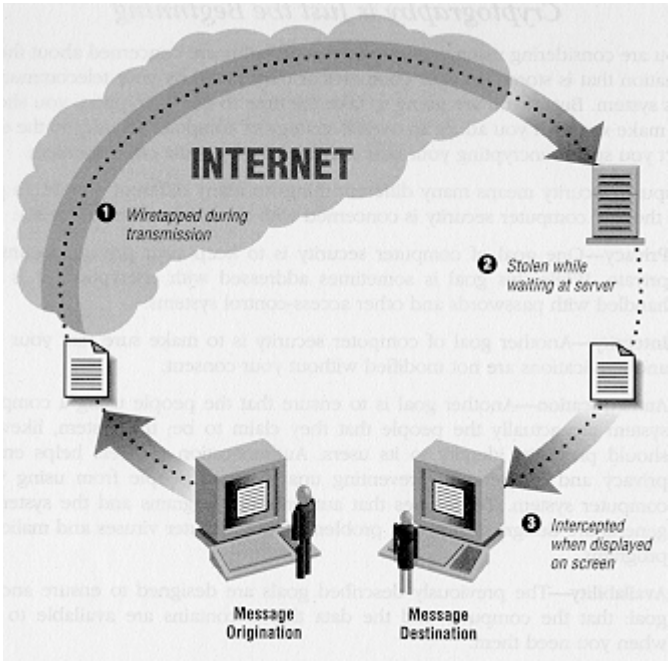
Resultatet av arbetet är att använda en Triple-DES algoritm, som är en så kallad symmetrisk algoritm, för kryptering av dokument samt de nycklar som skall överföras. Valet är baserat på att metoden för kryptering anses vara väldigt säkert och pålitligt, samtidigt som det är väldigt lättillgänglig och fri för distribution utan att några licenser erfordras.

En implementation har gjorts av en prototyp till det utvalda kryptosystemet för att påvisa att det är fullt möjligt med begränsade resurser, i form av tid och kunskap, att implementera kryptosystemet. Dessutom har systemets gränssnitt mot användaren exemplifierats och riktlinjer för dessa tagits fram.

Arbetet skall ge insikt i vilken nivå av säkerhet som behövs för den miljö som beskrivs i detta arbete (se kap. 2.7). Om ett färdigt överföringssystem skall användas skall kan detta arbete även fungera som en guide till att välja ett lämpligt sådant system.

1 Introduktion

I dagens samhälle används datorer och nätverk för att i allt större omfattning skicka information. De olika typer av information som sänds över nätverket blir allt fler. Det är numera inte ovanligt att mycket känslig information skickas mellan datorer över nätverk. Sådan känslig information kan exempelvis vara personnummer, hemliga koder, privata och personliga uppgifter eller hemlighetsstämplad företagsinformation. Även information som inte är direkt hemlig kan ändå orsaka skada om någon obehörig får tag på denna information och använder den på ett negativt eller destruktivt sätt.



Figur 1: Ett meddelande slickas över Internet [Gar95]

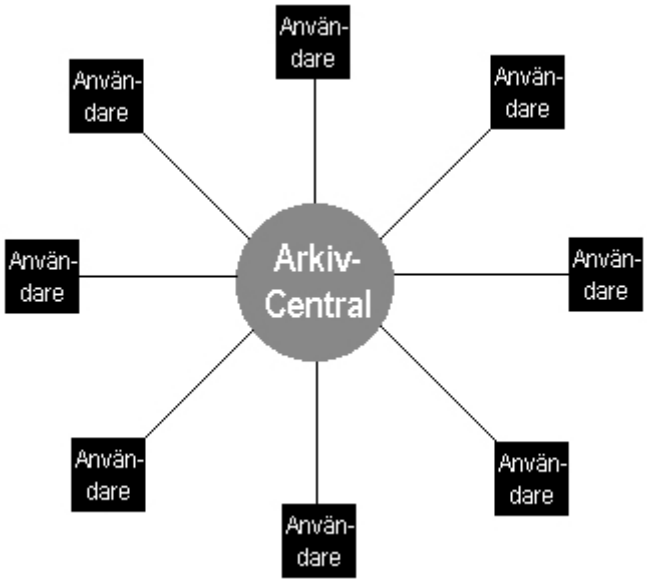
Ovanstående figur illustrerar ett scenario med två personer där den ena personen skickar ett meddelande över Internet till den andra. Om meddelandeöverföringen sker utan något form av skydd som döljer innehållet för obehöriga personer, kan en annan person lyssna av nätverket och läsa meddelandet när det skickas över Internet. Det kan även tänkas att någon, som har tillgång till den dator meddelandet skickas igenom, stjäl brevet när det tillfälligt lagras där. När mottagaren skall läsa sitt brev kan meddelandet snappas upp av en person som råkar befinna sig i samma rum och kan läsa det direkt från skärmen. Att skicka ett meddelande över exempelvis Internet kan jämföras vid om någon skickar ett vanligt vykort till någon, där vem som helst som kommer i kontakt med kortet under dess transport till mottagaren kan läsa hela innehållet i meddelandet.

Ett sätt att skydda sig mot att obehöriga personer tar del i den information som skickas över nätet är att använda sig av kryptering. Kryptering omvandlar tillfälligt meddelandet till förhoppningsvis ett för andra personer otolkningsbart skick. När det når mottagaren kan han eller hon återigen omvandla tillbaka meddelandet till ett läsbart format.

Studerandet av kryptering har pågått länge. Redan i det gamla Romarriket använde sig Julius Caesar av kryptering för att skicka hemliga meddelanden [Bec88]. Målet med krypteringsstudierna är att hitta den perfekta metoden för kryptering så det till hundra

procent går att garantera en säker överföring av data via ett nätverk utan att obehöriga personer kommer åt informationen. Eftersom dataöverföring kommer bli allt mer vanligt i framtiden [Bec88] är det viktigt att forskning sker inom detta område då det fortfarande inte finns någon metod för kryptering som garanterar en fullständig säkerhet. Även då det idag finns metoder som kan betraktas som i stort sett helt säkra, utvecklas system parallellt för att knäcka dessa algoritmer. Allteftersom tekniken hela tiden förbättras krävs det samtidigt att metoderna för kryptering blir alltmer komplexa.

Problemområdet som detta arbete fokuserar på är tillämpandet av kryptering av elektroniska dokument för en specifik problemmiljö. Denna miljö innehåller en central för arkivering av elektroniska dokument. Till denna central finns ett godtyckligt antal användare anslutna via modem i form av en stjärnstruktur. Dessa användare använder sig av telenätet för att skicka över samt hämta dokument till och från det centrala arkivet. Kryptering av de elektroniska dokumenten måste ske vid överföring samt vid lagring i centralen för att obehöriga personer ej skall få tillgång till denna information. Arkivcentralen ägs och drivs av ett företag. En användare utgör en kund till detta företag.



Figur 2: Arkivcentral med dess uppkopplade användare

Kunskapen och datorvanan hos användarna kan variera kraftigt från användare till användare. De kan dessutom vara involverade i väldigt många olika slags branscher. En användare kan exempelvis vara ett stort företag eller en statlig myndighet med relativt stor kompetens och rutin eller en privatperson eller ett mindre företag så som en bonde. Stora krav ställs därför på systemets design då det måste vara väldigt lättförståeligt och enkelt att använda.

Mycket forskning har gjorts inom området kryptering och många olika generella metoder eller generella algoritmer för kryptering av data, samt utvärderingar av dessa, har tagits fram. Några exempel på några sådana existerande krypteringsalgoritmer är; DES [Rik85], IDEA [Gar95], XOR [Bec88], RSA [Bec88] och PGP [Gar95]. Generella utvärderingar av olika typer av krypteringsmetoder har alltså genomförts sedan tidigare.

Ett syfte med detta arbete är att designa ett kryptosystem för distribution av dokument till en central lagringsplats via telenätet, som tar hänsyn till de speciella krav som kan

ställas på ett sådant system. Arbetet fokuserar på att skapa en systemdesign samt en implementation av ett delsystem som tillåter en säker och en för användaren enkel överföring av data. Detta arbete unikt eftersom ett lämpligt kryptosystem skall tas fram med avsikten att användas i den specificerade problemmiljön. Hänsyn till val av lämpligt system tas därför till vilka egenskaper som är viktiga för den specifika problemmiljön.

Tidigare forskning inom området kryptering har, som nämnts, gjorts i en relativt stor utsträckning men då mer generellt. Exempelvis har olika algoritmer enbart jämförts med varandra utan att ta hänsyn till någon specifik miljö. Denna tidigare forskning används dock i detta arbete på så sätt att vetskap om olika egenskaper för olika kryptosystem används för att jämföra med de krav som ställs på den specifika problemmiljön.

Arbetet skall ge insikt i vilken nivå av säkerhet som behövs för den miljö som beskrivs i detta arbete (se kap. 2.7). Om ett färdigt överföringssystem skall användas skall kan detta arbete även fungera som en guide till att välja ett lämpligt sådant system.

2 Bakgrund

Denna sektion har för avsikt att göra en kortfattad beskrivning av kommunikation över telenätet med inriktning på kryptering av den data som skall överföras, samt exempel på olika typer av datakryptering. Presentationer av några existerande metoder eller algoritmer för kryptering kommer att ske. Egenskaper, fördelar, nackdelar, samt tillämpningsområden för dessa metoder kommer att diskuteras. Även domänen för problemet i detta arbete kommer att redogöras. Syftet med denna information är att den skall ligga som grund för diskussion kring problemlösningar i det återstående arbetet.

Tryckt litteratur samt Internetsidor ligger som grund till beskrivningar och värderingar av olika kryptosystem. Referenser till dessa källor finns i samband med informationen i texten.

2.1 Kommunikation över telenätet

I rapporten kommer endast kommunikation över telenätet att innefattas i problemdomänen. Därför behandlar detta delkapitel enbart information som berör denna form av nätkommunikation.

PSTN är förkortning för Public Switched Telephone Network [Hal92], dvs telenätet vilket är ett nätverk för överföring av analoga signaler. När information skall överföras över PSTN måste först en linje upprättas mellan sändaren och mottagaren. Detta sker som bekant varje gång ett vanligt telefonsamtal skall göras. Således är kommunikationen i stort sett avskärmad från annan kommunikation över den resterande delen av telenätet.

För att kunna överföra digital data över PSTN, som är analogt, måste ett så kallat *modem* användas vilket är en förkortning av *MOdulator-DEModulator* [Hal92]. Ett modem har till uppgift att omvandla digitala signaler, som den sändande datorn skickar ut, till analoga så att de kan skickas över PSTN, samt att i omvänd ordning även omvandla inkommande analoga signaler till digitala signaler som en mottagande dator kan läsa.

2.2 Kryptosystem

Ordet *kryptering* kommer från det grekiska ordet kryptos vilket betyder dold [Rik85]. Med begreppet *klartext* menas en sådan text som är likvärdig vid det skriftspråk som en person använder sig av. Detta skriftspråk är relaterat till denna persons talspråk. Att kryptera en klartext innebär således att denna döljs genom att omvandla den till en kryptotext, som ej kan förstås av obehöriga. En behörig person kan sedan omvandla tillbaka den till klartext igen. Detta kallas för *dekryptera* en text. Tillvägagångssätt vad beträffar medel och metod för kryptering och dekryptetring kallas för *kryptosystem*. Att kryptera eller således dölja innehållet för obehöriga i ett meddelande när det skickas kan liknas vid att ett brev placeras i ett kuvert som klistras igen så att ingen annan person kan läsa det under tiden det transporteras till mottagaren.

Kryptering är inget nytt begrepp, utan har funnits sedan långt tillbaka i tiden. En av de först kända personer som använde kryptering var Julius Caesar [Bec88]. Han dolde texten i sina meddelanden genom att skifta varje bokstav tre steg framåt i alfabetet. Exempelvis blev då bokstav B bokstav E. Detta är ett exempel på en väldigt gammal och enkel form av kryptering. Skillnaden mellan dagens och gårdagens

krypteringsmetoder är att det längre tillbaka i tiden inte fanns tillgång till samma teknik som idag. Datorer är ett väldigt effektivt hjälpmedel som gör att det blir mycket enklare att knäcka en krypterad text. Naturligtvis kan på samma sätt konstrueras mer komplexa krypteringsalgoritmer med datorernas hjälp.

De tidiga enklare metoderna för kryptering av text baseras vanligtvis på att varje tecken i en text byts ut mot ett annat tecken efter olika specifika regler och mönster, eller att tecknen i en text kastas om till en annan ordning. Dessa typer av kryptering brukar betecknas som *utbyteschiffer* [Rik85] och *omkastningschiffer* [Rik85]. Ytterligare en typ av krypteringsalgoritmer är matematikbaserade krypteringsalgoritmer. Dessa bygger på att varje tecken identifieras med en siffra som sedan kan manipuleras mha olika matematiska algoritmer för att ta fram en kryptotext. Det är naturligtvis mer behändigt att manipulera med siffror än med tecken. I synnerhet när detta arbete sker med hjälp av datorer.

Samtidigt som kunskap om nätverkskommunikation har blivit större och mer utspridd har även hotet för avlyssning och tydning av data som sänds över nätverk blivit större. Inte sällan innehåller information som sänds över nätverk känsliga uppgifter som exempelvis lösenord. Dvs sådana uppgifter som, om de kommer i fel händer, kan utgöra obehag för en individ eller vara negativ för en organisation. Ytterligare ett exempel på hur illasinnade obehöriga personer, som använder sig av avlyssning, kan orsaka skada är s k *masquerading* [Bec88]. Masquerading innebär att en gammal sekvens av information, som kommits över genom avlyssning, används för att generera en ny sekvens med ändrat innehåll. Detta kan naturligtvis också få många negativa följder. Med andra ord så kan information som inte nödvändigtvis är hemlig användas på ett sådant sätt att den kommer till skada på något sätt. Detta är några orsaker till varför kryptering alltid bör användas för data som skickas över ett nätverk.

2.3 Kryptonyckel

I de första kryptosystemen låg hemligheten för kryptering och dekryptering i själva systemet vilket medförde att det var av högsta vikt att systemets funktion hölls hemligt. Detta var opraktiskt då det även var nödvändigt att hemlighålla de krypteringsapparater som innehöll kryptosystemet. Om någon obehörig person skulle få tillgång till den hemliga utrustningen skulle detta innebära att ett nytt kryptosystem fick konstrueras, vilket var både kostsamt och tidsödande.

Numera är det möjligt att mha *kryptonycklar* hålla systemen helt öppna, dvs informationen om hur kryptosystemet fungerar behöver ej hållas hemlig. Det enda som hålls hemligt är själva nyckeln. En kryptonyckel består av data som appliceras på kryptosystemet för att kunna ta fram en kryptotext eller en klartext.

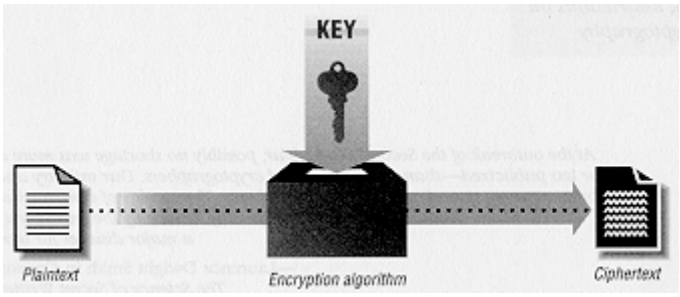
Kryptonycklar kan se ut och användas på olika sätt beroende på vilken krypteringsmetod som används. I de system där kryptering sker mha utbyteschiffer är längden på nyckeln lika lång som längden på texten eller datamängden eftersom varje tecken i klartexten är relaterad till exakt ett tecken i kryptonyckeln för att ta fram kryptotecknet. En nyckel som används i omkastningschiffer innehåller den informationen som talar om hur bitarna eller tecknen kastats om. I matematiska kryptosystem innehåller nyckeln en serie med siffror.

Användning av nycklar gör att ett väldigt flexibelt system fås. Genom att enbart byta ut nyckeln fås ett helt nytt kryptosystem. Att ofta byta kryptosystem gör att säkerheten blir högre. I vissa typer av kryptosystem används nyckeln endast en gång, så kallade

engångschiffer. Systemen blir även lättare att distribuera eftersom enbart kryptonyckeln behöver transporteras i hemlighet.

2.3.1 Symmetriska kryptosystem

I så kallade *Symmetriska kryptosystem* används exakt samma nyckel för kryptering och dekryptering.

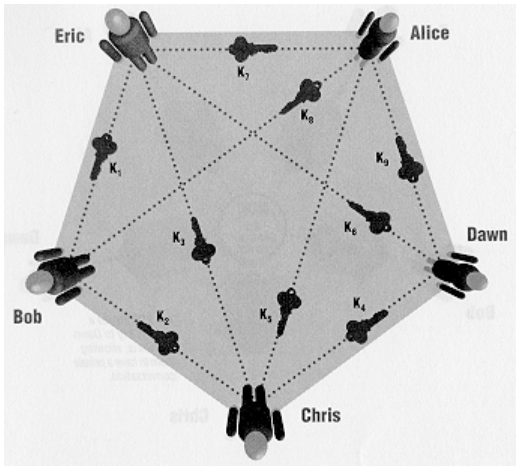


Figur 3: Symmetriskt kryptosystem [Gar95]

Vid en krypterad förbindelse måste alltså sändaren och mottagaren ha tillgång till en och samma nyckel, samtidigt som den måste vara okänd för övriga obehöriga personer. I ett nätverk med flera abonnenter där varje abonnent skall kunna ha en fullt krypterad förbindelse med varje annan abonnent, blir nyckelhanteringen både komplicerad och långsam eftersom sändaren måste skicka över denna gemensamma nyckel till mottagaren för att denne skall kunna dekryptera meddelandet. Låt säga att antalet personer eller abonnenter är *N* st. Då måste det finnas *M* st nyckelpar för att samtliga personer skall kunna skicka meddelanden till varandra.

$$M = \frac{N * (N-1)}{2} \qquad \text{(EQ 1)}$$

Själva distributionen av den hemliga och gemensamma nyckeln är som synes en väldigt viktig och känslig procedur.

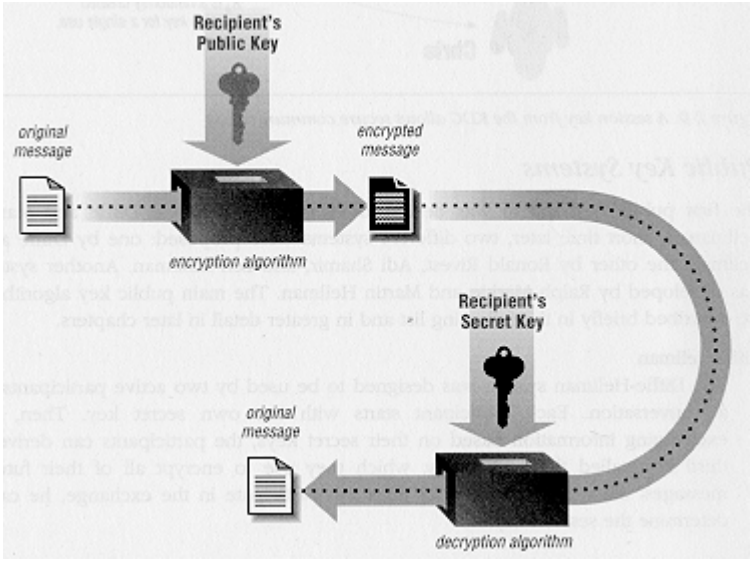


Figur 4: Distribution av nycklar [Gar95]

2.3.2 Osymmetriska kryptosystem

En lösning på problem som kan uppstå genom att använda sig av symmetriska kryptosystem (se kap. 2.3.1) är att använda sig av en nyckel för kryptering och en annan för dekryptering. Dessa system kallas för *osymmetriska kryptosystem*. Mottagarens nyckel för dekryptering av meddelande måste hållas hemlig. Däremot

kan sändarens nyckel för kryptering vara känd av alla. Därför kallas även dessa system för *öppen-nyckel-system*. Idén bakom öppna nycklar kom till år 1976 [Rik85].



Figur 5: Osymmetriskt kryptosystem [Gar95]

Det är möjligt att härleda en fungerande krypteringsnyckel om tillgång till dekrypteringsnyckeln finns, men inte tvärt om. Detta grundar sig på så kallade *enkelriktade funktioner*. Eftersom en persons allmänna nyckel är helt öppen och fri för distribution är det möjligt att en persons allmänna nyckel finns tillgänglig för allmänheten i en nyckeldistributionscentral där vem som helst kan hämta en krypteringsnyckel, för en specifik person, för att sedan sända krypterad information till denne utan att någon obehörig person får tillgång till den överförda datamängden.

En nackdel med osymmetriska kryptosystem är att de är ganska långsamma i och med att de kräver mycket beräkningar. Osymmetriska kryptosystem är faktiskt ca 1000 gånger långsammare än symmetriska system [Gar95]. Därför används öppna nycklar oftast enbart till att överföra nycklar så att det går att upprätta en kommunikation med ett symmetriskt kryptosystem, där samma nyckel används för kryptering respektive dekryptering.

Det finns flera olika sätt att implementera ett system med öppna nycklar. De flesta system baseras dock på metoden att använda sig av logaritmiska funktioner av olika slag. Ett exempel på hur det är möjligt att utnyttja sådana funktioner är att titta på följande formel:

$$y = b^x \bmod p \qquad \text{(EQ 2)}$$

I ovanstående formel där p är ett primtal är det matematiskt ganska lätt att beräkna y för ett godtyckligt b och x, dvs att göra beräkningar av tal av formen mod p är enkelt och kräver som högst $2 * \log_2 p$ multiplikationer [Bec88]. Om i stället y finns och x vill beräknas är det matematiskt ogenomförbart om fallet är sådant att p är ett väldigt stort primtal enligt nedanstående formel:

$$x = \log_b y \bmod p \qquad \text{(EQ 3)}$$

Att hitta x, om ett stort och bra primtal har hittats, kräver ungefär $p^{\frac{1}{2}}$ operationer [Bec88]. Tiden för att beräkna fram x växer exponentiellt med storleken på primtalet jämfört med tidigare då y skulle beräknas.

RSA är en ett exempel på en krypteringsmetod som använder sig av öppna nycklar. En beskrivning av algoritmen kommer att presenteras utförligare i ett senare stycke.

2.3.3 Problem kring nyckelhantering

Säkerheten hos ett kryptosystem vilar ytterst på nyckelhanteringen. Databehandling och datakommunikation har till uppgift att förenkla och rationalisera arbetsuppgifter, problemlösningar eller liknande uppgifter. Detaljer som kan uppfattas som krångliga eller svåra för användare gör att systemet kan uppfattas negativt. Alltså är det viktigt att nyckelhanteringen utformas så att den får maximal säkerhet samtidigt som den innebär ett minimalt besvär för användarna. Kryptering av data som skall skickas över ett nätverk ger störst problem.

Följande punkter är några delproblem kring nyckelhantering:

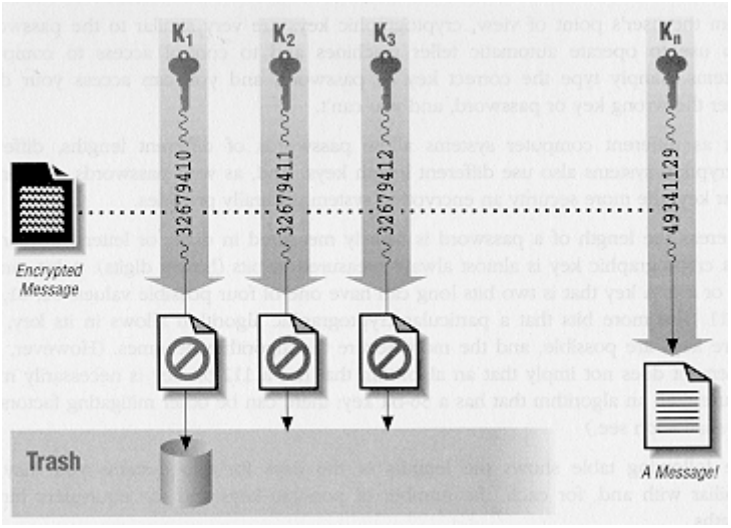
- När ett meddelande har skickats måste mottagaren veta vilken nyckel som har använts för att kunna dekryptera meddelandet. Vid symmetrisk kryptering används endast en nyckel. Denna måste då hållas hemlig. Ändå måste den skickas över nätverket till mottagaren. Således måste denna nyckel skickas till mottagaren på ett säkert sätt. Problemet är att hitta ett lämpligt media för denna överföring.
- Styrkan hos ett krypterat meddelande beror delvis på hur stor mängd data som krypterats med samma nyckel. Därför är det bra att byta nyckel så ofta som möjligt. Om det går att lyckas med att överföra en nyckel på ett väldigt säkert sätt går det sedan använda den till att dekryptera nycklar. Denna nyckel kallas för en *sekundärnyckel*. Ett problem är hur ofta denna sekundärnyckel bör bytas.
- Den första nyckeln är oftast svårast att överföra och är vanligtvis både krångligt och tidsödande. I ett system där de två parterna har en separat och strikt uppkoppling kan detta genomföras utan några större problem. I ett större system där flera personer använder samma nät blir det betydligt mer komplicerat i synnerhet då varje person skickar hemlig information till flera personer. Lösningen kan vara att upprätta en s k nyckeldistributionscentral som har till uppgift att tillhandahålla och distribuera nycklar. Problemet här är att avgöra hur en sådan skall se ut och avgöra om den över huvud taget behövs.

2.4 Forcering

Naturligtvis förekommer det att obehöriga personer försöker komma åt den hemliga information som genomgått en krypteringsprocedur. *Forcering* [Rik85] innebär bearbetning av en kryptotext för att utan kännedom om använd kryptonyckel kunna härleda den dolda klartexten eller den dolda nyckeln. Ett begrepp i dessa sammanhang är *forceringsmotstånd*. Forceringsmotstånd är ett mått på svårigheten i att forcera ett kryptosystem. Önskvärt är att ett kryptosystem har ett så högt forceringsmotstånd som möjligt. Det finns ingen enhetlig skala och mätmetod för att enkelt avgöra ett kryptosystems styrka. Den tid det tar för att knäcka ett kryptosystem beror dels på vilken metod som tillämpas samt vilken hårdvara som finns att tillgå.

Ett sätt att försöka dekryptera ett meddelande är att helt enkelt systematiskt gå igenom och pröva olika nycklar för att se om något läsligt resultat kan fås ut. Detta kallas för *nyckelforcering* [Gar95]. Ett kanske smartare sätt, för vissa typer av kryptosystem, är att analysera den dekrypterade texten och finna mönster. I en vanlig engelsk text är exempelvis tecknen t, o, e och blanktecken de mest frekvent förekommande. Med denna vetskap kan sedan den krypterade texten analyseras med hänsyn till vilka

tecken som är mest vanligt förekommande där. Detta fungerar bara på enklare typer av kryptosystem men är ändå ett exempel på hur analyser kan gå till.



Figur 6: Nyckelforcering [Gar95]

Det finns underlag för hur styrkan hos ett kryptosystem väljs. Som beslutsunderlag får analys göras av på hur motståndaren ser ut, dvs hur ser hotbilden ut? Är det en motståndare med begränsade ekonomiska resurser samt mindre kvalificerad personal och utrustning kan det tänkas att det räcker med ett enklare kryptosystem. Faktum är att det inte alltid är så lätt att bedöma relationen mellan styrkan hos kryptosystemet och motståndarnas forceringsinsatser, speciellt som relationer varierar med teknikens utveckling. Minskad kryptostyrka innebär alltså en kraftigt ökad osäkerhet om vilka hotbilder som systemet kan klara av.

Några fördelar med att välja ett kostsamt system som kan stå emot försök till forcering av systemet är exempelvis att man slipper de ekonomiska förluster som uppkommer när någon obehörig kommer åt den hemliga informationen. Den prestigeförlust som en forcering innebär kan också undvikas. Ett nytt kryptosystem behöver ej installeras det använda skulle bli forcerat.

De tillfällen då ett enklare kryptosystem kan löna sig är exempelvis i mindre projekt som innebär kortsiktiga ekonomiska vinster.

2.5 Några existerande krypteringsmetoder

I detta kapitel presenteras några av de mer kända och vanligast förekommande krypteringsmetoderna, samt de som kommer att beröras i längre fram i rapporten. Dock kommer ej någon beskrivning av krypteringsmetoderna att ske på en djupare nivå. Syftet är att bli introducerad till hur kryptering ser ut och fungerar praktiskt, samt att få en inblick i vilka system som används idag.

2.5.1 XOR

Denna algoritm, som är symmetrisk, är mycket enkel i sin funktion. För framtagning av kryptomeddelandet används den binära operationen *exclusive-or* som förkortas *XOR* [Bec88]. Funktionen har även symbolen ⊕. För den som är obekant med operationen ser den ut som följande:

$x \oplus y$	**0**	**1**
0	0	1
1	1	0

Tabell 1 : Funktionen XOR.

Metoden använder binära chiffer för kryptering, dvs texten som skall krypteras måste först omvandlas till binär kod för att kunna genomgå krypteringsalgoritmen. Till krypteringen används en binär nyckel som är minst lika lång som det binära meddelandet. För att kryptera en text utförs XOR operationen på varje par av bitar från klartexten samt nyckeln.

Nedan följer ett exempel på hur den fungerar. Låt säga att klartexten ’SVERIGE’ skall krypteras med nyckeln ’DANMARK’. För enkelhetens skull kan alfabetets bokstäver tilldelas siffrorna 1 till 29 som sedan översätts till binära tal. Då fås alltså:

	S	V	E	R	I	G	E
Klartext:	10011	10110	00101	10010	01001	00111	00101
	D	A	N	M	A	R	K
Nyckel:	00100	00001	01110	01111	00001	10010	01011
	W	W	K	Ö	H	W	N
Krypterat:	10111	10111	01011	11101	01000	10111	01110

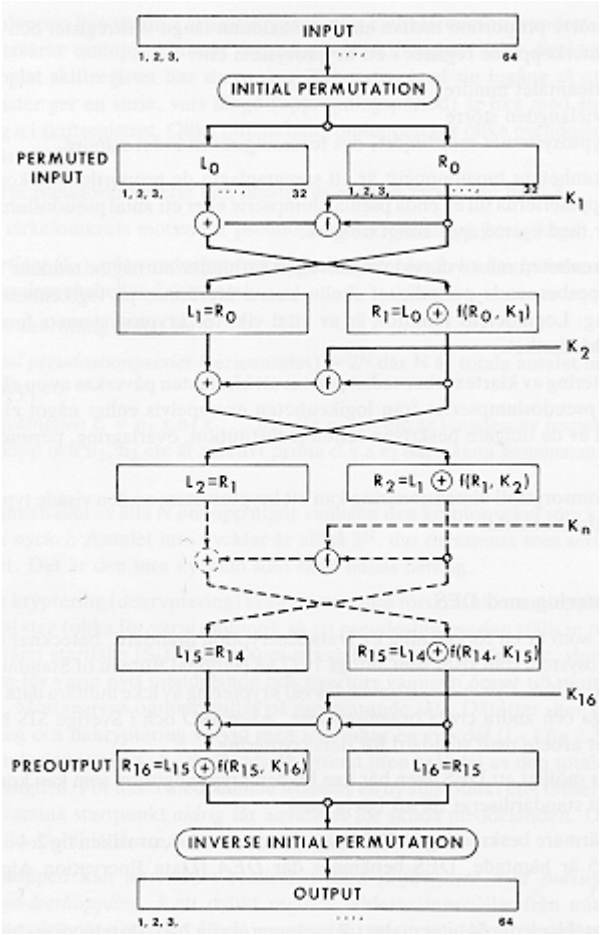
Det krypterade meddelandet blir således ’WWKÖHWN’. Detta är ett väldigt enkelt men effektivt sätt att utföra en bra kryptering på. Om ej tillgång finns till nyckeln är det nästan omöjligt att knäcka den, förutsatt att inte en nyckel som är väldigt kort i förhållandet till meddelandet används. Därför är det önskvärt för att få en bra säkerhet att nyckeln skall vara minst lika långt som meddelandet. Algoritmen som använder sig av binära operationer lämpar sig väl för datorer som också arbetar binärt. Detta gör den till en väldigt snabb krypteringsmetod. Nackdelen är nyckelhanteringen. De blir långa och besvärliga att distribuera. Ett sätt att lösa problemet med långa nycklar är att använda sig av algoritmer som beräknar fram en serie av slumptal. En sådan slumptalsgenerator baserar beräknandet av ett slumptal på föregående värde på slumptal. Alltså är det möjligt, om två likadana algoritmer existerar, att få fram samma serie, såvida samma startvärde tilldelas.

2.5.2 DES

DES står för *Data Encryption Standard* och är ett kryptosystem skapat av IBM som antogs år 1977 av National Bureau of Standards [Rik85]. Systemet har för avsikt att användas för kryptering av data för användning inom statliga eller privata organisationer. Dock anses DES ej tillräckligt säkert för att användas på statliga hemlighetsstämplade dokument.

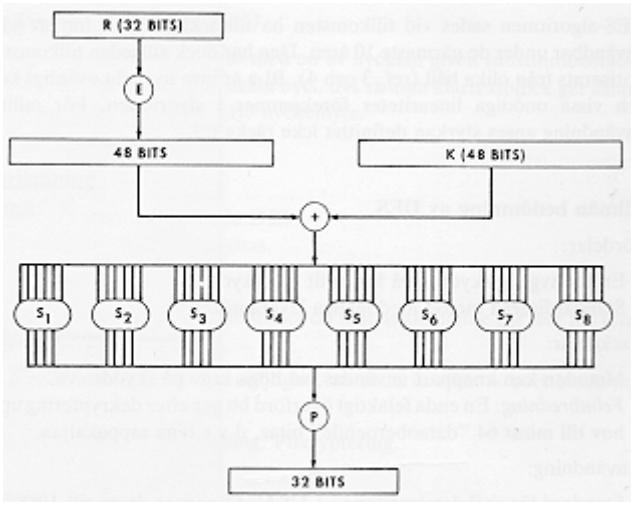
Algoritmen är symmetrisk, dvs samma nyckel används för kryptering och dekryptering. Metoden är ett chiffer som arbetar i blockform. Dvs, algoritmen delar

upp meddelandet i block av lika längd. DES använder sig av block med längden 64 bitar. Nyckeln består av 56 bitar. Datablockets bitar genomgår en operation med nyckelbitarna enligt specifika regler i 16 steg enligt ett iterativt förfarande. Under krypteringens gång delas nyckeln upp i 16 st delnycklar av längden 48. Även blocket med datan kommer internt att delas upp i 2 st delblock med längden 32. Innan ett delblock genomgår en operation med delnyckeln expanderas den till längden 48 så att den får samma längd som delnyckeln. Följande bild visar kortfattat hur algoritmen går till väga.



Figur 7: Kryptering med DES algoritmen [Rik85]

I figur 8 kan beskådas hur de 16 olika delnycklarna opererar, samt hur det ingående meddelandeblocket av längden 64 bitar delas upp i två delblock med längden 32 bitar vardera. Vid slutet av algoritmen sätts återigen delblocken ihop till ett block bestående av 64 bitar men är då av ett krypterat format.



Figur 8: Delfunktionen f(R_n, K_{n+1}) [Rik85]

Ovanstående figur visar funktionen f(R_n, K_{n+1}). Det ingående delblocket R_n av längden 32 bitar genomgår en operation som ökar dess längd till längden av delnyckeln K_{n+1}, dvs 48 bitar. Sedan utförs XOR-operationen på dessa. Resultatet genomgår en operation som sätter det manipulerade delblocket till dess ursprungliga längd 32 bitar.

Fördelarna med denna metod är bl a att den faktiskt har en relativt bra skyddsnivå för användning i en civil miljö. Den är en standardiserad metod som många leverantörer använder sig av.

Några nackdelar med detta kryptosystem är att det inte kan användas då väldigt höga krav på säkerheten och skyddsnivån ställs. En enda felöverförd bit innebär att texten blir helt oläslig efter kryptering. Den är m a o väldigt känslig för störningar. Eftersom den är en symmetrisk algoritm uppstår även problemet med distribution av nycklar. Då antalet personer i ett kommunikationssystem blir stort blir antalet nycklar exponentiellt större i förhållandet till antal personer eftersom ett unikt nyckelpar måste finnas för varje möjligt par av sändare och mottagare.

En variant av DES är en metod som kallas för *Triple-DES*. Som namnet antyder används vid kryptering DES tre gånger på en och samma data. Vid varje delkryptering används en unik nyckel, dvs algoritmen använder sig av en nyckel av längden 168 bitar, vilken är tre gånger så lång som den nyckel DES använder sig av.

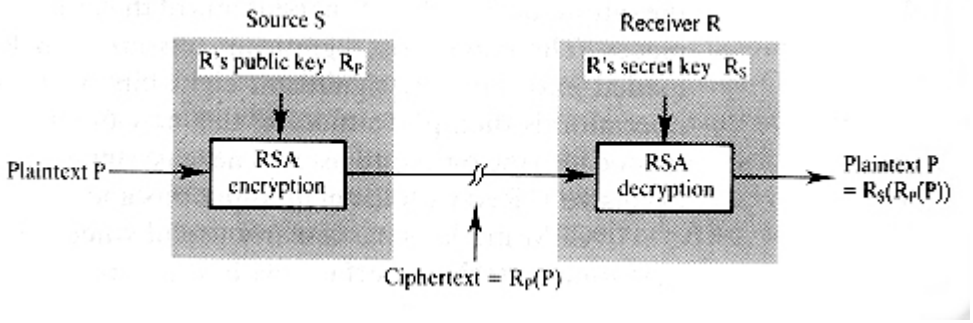
2.5.3 IDEA

IDEA står för International Data Encryption Key och är skapad i Schweiz av James L. Massey och Xuijia Lai år 1992. Det är en symmetrisk algoritm som använder sig av en 128-bitars nyckel. IDEA är som DES ett chiffer och fungerar på liknande sätt fast med en annan algoritm. Skillnaden är också att nyckeln för IDEA är dubbelt så lång som för en nyckel i DES. Att systematiskt prova samtliga nyckelkombinationer för att dekryptera meddelandet skulle ta en sådan orimligt lång tid att det är onödigt att försöka. Då algoritmens funktionalitet liknar den i DES kommer inte någon närmare förklaring av IDEA algoritmen att ges.

2.5.4 RSA

En osymmetrisk krypteringsmetod som använder sig av öppna nycklar är *RSA* algoritmen. Ronald *Rivest*, Adi *Shamir* och Leonard *Adlemans* är namnen på de tre

personer som år 1977 tog fram algoritmen. Det är deras initialer som har bildat namnet på krypteringsmetoden [Bec88]. I denna krypteringsalgoritm spelar primtal och faktorering en viktig roll. Metoden baseras på problemet att det för vissa tal som är väldigt stora är svårt att utföra en faktorering av dessa, i alla fall inom en rimlig tidsrymd.



Figur 9: Meddelandeöverföring med RSA kryptering [Hal92]

Nedan visas väldigt förenklat hur algoritmen går till väga för att beräkna de bägge nycklarna samt hur dessa används för att kryptera och dekryptera meddelandet [Jär97].

- Välj två stora primtal *p* och *q*.
- Beräkna $n = p * q$.
- Beräkna *phi* som $(p - 1)(q - 1)$.
- Kassera *p* och *q* som nu inte behövs mera.
- Välj ett tal *e* som är relativt prima *phi*, det vill säga att talens SGD (Största Gemensamma Divisor) är 1. Eller på ren svenska *phi* / *e* går ej att förkorta. De triviala lösningarna $e = 1$ och $e = phi$ är ej giltiga val.
- Beräkna *d* som den multiplikativa inversen till *e* MOD *phi*, det vill säga $(e * d)$ MOD $phi = 1$. Undvik av säkerhetsskäl att välja $e = d$ även om det skulla vara en lösning till ekvationen.
- *e* är nu de publika exponenten och *d* den privata exponenten.
- Den publika nyckeln är nu paret $\{n, e\}$ och den privata $\{n, d\}$.
- Ta den text, bokstav *T* som ska krypteras.
- Ta den privata nyckeln och kryptera *T* till *K* enligt följande: $K = T^d$ MOD *n*.
- Skicka *K*
- Alla som har tillgång till den publika nyckeln kan nu dekryptera *K*, $T = K^e$ MOD *n*. Eftersom så gott som alla kan få tillgång till en publik nyckel så är användningsområdet för denna krypteringsform tänkt att fungera åt andra hållet. Någon av dem med en publik nyckel använder denna för att kryptera ett meddelande ($K2 = T2^e$ MOD *n*) som bara personen med den privata nyckeln kan dekryptera ($T2 = K2^d$ MOD *n*).

Det är mycket svårt (med den kunskap som finns idag) att lista ut *d* utifrån den publika nyckeln $\{n, e\}$. Men om faktoriseringen av *n*, till *p* och *q*, så skulle det vara möjligt att räkna fram *d*. Säkerheten i RSA bygger på att faktoriseringen av *n* är för

svår för att vara genomförbar om mycket stora primtal *p* och *q* används. En riktlinje för att uppnå en beräkningsmässigt säker kryptering är att välja primtal som har 100 siffror eller mer, detta ger ett *n* på ungefär 200 siffror. Då blir det praktiskt omöjligt att faktorisera *n* inom en realistisk tid.

Fördelen med RSA är att en den passar väl i en miljö där nyckeldistributionen är stor, dvs varje person i nätverket skickar eller tar emot meddelanden till och från flera andra personer.

En nackdel med algoritmen är att beräkningarna tar förhållandevis lång tid vilket gör den något långsam att använda i alltför stor utsträckning.

2.5.5 PGP

Phil Zimmermann är namnet på den man som i mitten på 80-talet påbörjade arbetet med *PGP* vilket står för *Pretty Good Privacy*. 1991 stod den första färdiga versionen klar för användning [Gar95]. PGP är ett osymmetriskt system, men algoritmen använder sig internt även av symmetriska krypteringsmetoder. Tre olika typer av nycklar förekommer inom systemet. I likhet med RSA finns det en nyckel för sändaren och en annan för mottagaren. Varje gång som ett nytt meddelande skall skickas över nätet skapas även en så kallad *sessionsnyckel* som är en symmetrisk nyckel. När information skickas över nätet är det denna nyckel som krypteras med det osymmetriska systemet. Själva meddelandet krypteras med sessionsnyckeln.

Följande är de olika delstegen som PGP utför när en meddelandeöverföring skall utföras:

1. Skapa en slumpvis sessionsnyckel för meddelandet.
2. Kryptera meddelandet med IDEA algoritmen och sessionsnyckel.
3. Kryptera sessionsnyckeln med RSA algoritmen och mottagarens öppna nyckel.
4. Ett paket skapas av det krypterade meddelandet och nyckeln för att sedan skickas iväg till mottagaren.

PGP använder sig således av flera olika kryptosystem. Olika system har för det mesta både för- och nackdelar. PGP tar vara på olika systems fördelar och kombinerar ihop dem på ett sådant sätt att ett starkt, säkert och effektivt system fås. Exempelvis har RSA den nackdelen att den är resurskrävande när beräkningar skall göras. Alltså sköter endast RSA krypteringen av de symmetriska nycklarna. Kryptering av själva meddelandet görs istället med IDEA som är både snabbt och effektivt. Den ömma tån vad gäller IDEA och andra symmetriska kryptosystem är nyckeln. Det är av yttersta vikt att kryptonyckeln hålls hemlig, vilket inte är alltför enkelt när den måste vara distribuerad på hos både sändaren och mottagaren. PGP löser detta genom att skapa en ny nyckel för varje meddelandeöverföring.

2.6 Västgöta företagsarkiv AB

Västgöta Företagsarkiv AB (*VFAB*) är ett nystartat företag som är inriktat på arkivering och dithörande tjänster. De erbjuder sina kunder ett heltäckande system för flödet av dokument och handlingar under sin livstid från uppkomst till slutarkivering eller destruktion. Verksamheten bedrivs i enlighet med de säkerhetsföreskrifter som är utfärdade av Finansinspektionen och i enlighet med Riksarkivets författningar.

En kund till VFAB skall kunna känna sig trygg i den mening att lagar och förordningar följs, utan att kunden skall behöva sätta sig i vad dessa är, samt hitta en

lämplig förvaringsplats som uppfyller alla dessa krav. P g a att arkiveringen följer alla lagar och restriktioner kan kunden vara säker på att viktiga dokument förvaras tryggt och säkert ur företagets synvinkel.

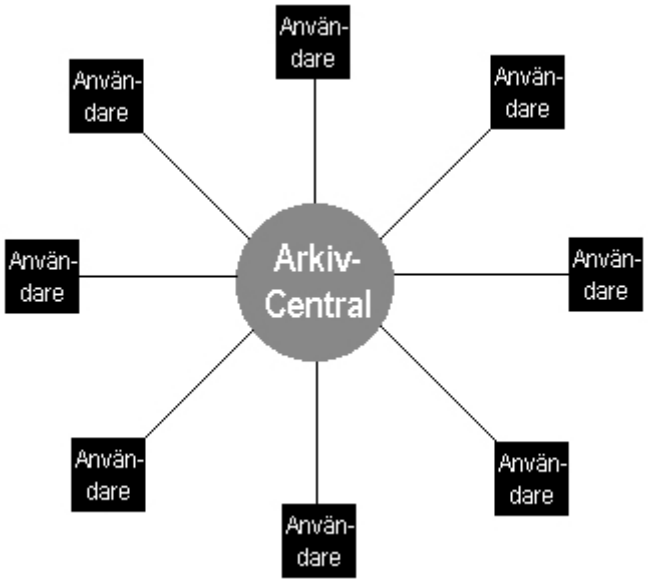
VFAB, som är beläget strax utanför Mariestad, är fortfarande inte något väletablerat företag och har för närvarande inte några anställda, men arbetar just nu mycket med att utöka sin kundkrets och utveckla sin service. Kontinuerligt knyter VFAB nya kontakter med viktiga kunder som troligtvis kommer göra företaget mer stabilt inom en snar framtid på ca 1-2 år.

Västgöta företagsarkiv AB erbjuder även andra tjänster relaterat till arkivering, som exempelvis; utbildning av arkivvård, destruktion av material, samt dokumentation av ett företags historik mm.

En ny affärsidé hos VFAB är att förtaget i framtiden skall kunna erbjuda ett system för arkivering av elektroniska dokument. I stort kommer dessa dokument att behandlas på samma sätt som de övriga dokumenten, dvs de kommer att lagras centralt i ett säkert arkivrum på ett pålitligt lagringsmedia av någon form.

2.7 Generaliserad problemdomän

Västgöta företagsarkiv AB är således behov av ett system som hanterar överföring av elektroniska dokument på ett säkert sätt. För att detta arbetet skall kunna användas för andra problem av liknande karaktär, har problemmiljön för det tänkta framtida systemet generaliserats till följande utseende:



Figur 10: Arkivcentral med dess uppkopplade användare

Denna miljö innehåller en central för arkivering av elektroniska dokument. Till denna central finns ett godtyckligt antal användare anslutna via modem i form av en stjärnstruktur. Dessa användare använder sig av telenätet för att skicka över samt hämta dokument till och från det centrala arkivet. Kryptering av de elektroniska dokumenten måste ske vid överföring samt vid lagring i centralen för att obehöriga personer ej skall kunna få tillgång till denna information.

Troligtvis kommer dessa elektroniska dokument, på samma sätt som de övriga, endast bli placerade i arkivet för att sedan lagras där ett antal år utan att kunden plockar fram

dem under den tid de ligger placerade där. Dokumenten kan vara av olika format men har alla gemensamt att de på något sätt är viktiga för kunden.

3 Problembeskrivning

Huvudproblemet är att hitta lämpligt system som skall kunna hantera och erbjuda en säker och pålitlig överföring av elektroniska dokument över PSTN. Systemet skall dessutom vara väldigt enkelt att hantera för användarna. Fokus kommer att ske på själva tillämpandet av kryptering av elektroniska dokument för den generella problemmiljön. Flera olika krypteringsalgoritmer som kan användas för att uppfylla de krav som finns på systemet kommer att studeras och utvärderas. Kraven som finns på systemet gäller specifikt för den generella problemmiljön. Förutom själva algoritmerna för kryptering finns det även andra mycket viktiga faktorer som ingår i systemet att studera. Exempelvis olika metoder för nyckelhantering vilket måste fungera på ett sådant sätt att systemet kan uppfylla de krav som ställs på det.

Detta kapitel kommer att ta upp avgränsningen för problemet, en närmare beskrivning av det avgränsade problemet, samt vad det förväntade resultatet är av arbetet.

3.1 Problemavgränsning

Ett system behöver utvecklas för överföring av elektroniska dokument från användare till en arkivcentral och omvänt från arkivcentral till användare, samt hantering av dessa dokument i en databas. Kommunikation för överföring av dokumenten kommer att ske via telenätet. Systemet skall kunna erbjuda en pålitlig och säker överföring av data. Dvs, ingen information skall kunna gå förlorad samtidigt som obehöriga personer ej heller skall kunna få tillgång till informationen. Detta gäller även för den data som finns lagrad i det centrala arkivet. Detta skall tillgodoses mha bland annat kryptering, viruskontroll, pålitlig hård- och mjukvara, samt en pålitlig och säker hårdvarumiljö.

På grund av det stora arbete som skulle behövas för att ta fram ett fullständigt system som tar hand om hela öveföringsprocessen samt förvaltning av dokument kommer vidden på problemområdet att skäras ned och fokus kommer ske på en mindre del av systemet.

Detta arbete kommer endast att fokusera på den delen av systemet som hanterar kryptering av data för överföring via PSTN, samt även diskutera hur kryptering av data för lagring av data i arkivcentralen kan gå till väga. Systemet för kryptering av data inkluderar, förutom algoritmer för kryptering, även olika aspekter så som nyckelhantering etc. Metoder för själva dataöverföringen och viruskontroll kommer ej att tas fram.

3.2 Problemprecisering

Problemet kommer bestå i att, med vetskap om de krav och egenskaper som finns och önskas för krypteringssystemet, ta fram en metod som hanterar krypteringen av de elektroniska dokument som skall skickas mellan arkivcentralen och användaren. Det är av yttersta vikt att systemet skall vara säkert och enkelt att hantera för användaren.

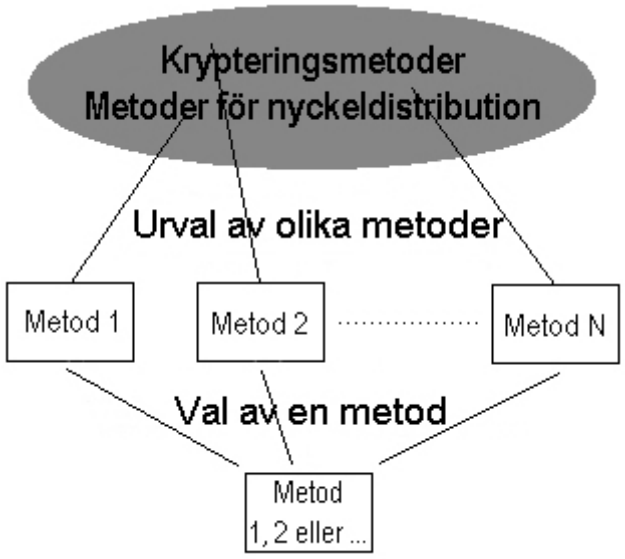
Förutom att systemet skall vara säkert och enkelt att hantera för användaren, skall det även vara snabbt, effektivt och pålitligt. Eftersom access till arkivcentralen sker relativt sällan är det viktigare att säkerheten är bra än att systemet är snabbt. Andra betydelsefulla egenskaper är att systemet skall ha ett utseende som inger pålitlighet och säkerhet för systemet hos användaren.

3.3 Förväntat resultat

Det förväntade resultatet kommer att utgöras av ett delsystem som ingår som komponent i ett större system. Detta senare system hanterar hela överföringsprocessen från användare till arkivcentral, och omvänt från arkivcentralen till kunden, samt hantering av dokument i det centrala lagringsutrymmet. Delsystemet utgörs av metoder för kryptering av data för överföring över PSTN samt hantering av distribution av nycklar.

De finns generellt två olika huvudtyper av metoder för kryptering, samt flera olika övergripande metoder för distribution av nycklar. Dessa två metoder för kryptering av data samt förslagsvis 4–5 metoder för nyckeldistribution kommer att studeras och utvärderas. Eftersom nyckelhantering är en del av krypteringssystemet kan sägas att en metod för kryptering av data innehåller en typ av krypteringsalgoritm samt en typ av metod för nyckeldistribution. Om en kombination av dessa två delsystem görs fås 8–10 alternativa metoder för kryptering av elektroniska dokument.

Dessa utvalda metoder för kryptering och nyckelöverföring kommer sedan att utvärderas. För- och nackdelar samt olika egenskaper kommer att diskuteras. En av metoderna kommer att lyftas fram och föreslås som en del av det slutgiltiga dataöverföringssystemet.



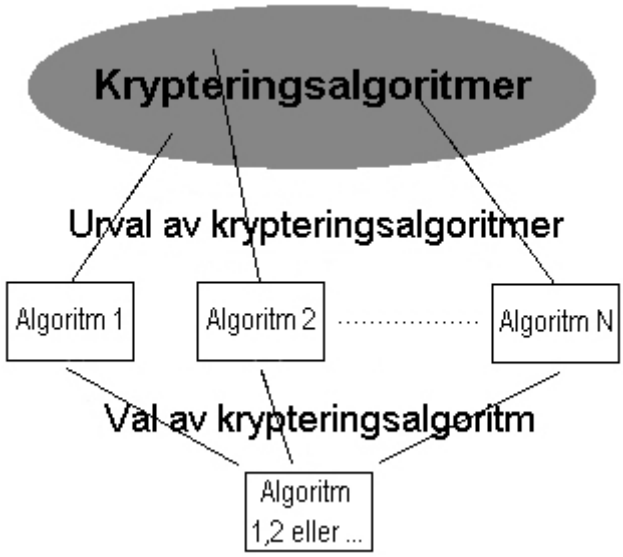
Figur 11: Processen för framtagning av en lämplig generell metod

Valet av typ av kryptosystem skall vara baserat på att den aktuella metoden innehar de egenskaper som krävs av den givna problemdomänen, dvs den skall vara anpassad för den aktuella miljön för att kunna erbjuda en bra säkerhet, samt en snabb och smidig hantering av dokumenten. Systemet skall vara väldigt enkelt och effektivt för företaget men framför allt för företagets kunder att använda. Systemet skall vara så utformat att det även uppfattas av kunden som ett väldigt säkert och pålitligt system.

Hur skall mätning av ovanstående kriterier så som säkerhet, snabbhet, pålitlighet och enkelhet ske? Många av dessa krav är omöjliga att mäta med ett exakt mått. Snabbhet kan mätas lätt med måttet tid. Säkerhet däremot är svårare att mäta enbart med ett tal. Istället kan exempelvis en jämföring med andra system göras, dvs en komparativ studie får föras. Ett exempel på att försöka mäta säkerheten hos ett system är att titta på hur många olika nyckelkombinationer som finns och sedan beräkna hur lång tid det

skulle ta för en viss typ av dator att pröva alla nycklar. Kravet på enkelhet kan inte heller mätas på ett enkelt sätt. Istället får exempelvis studier göras av olika system och undersökningar utföras med användare för att ta reda på vilken typ av system de finner mest önskvärda.

När valet av generella metoder för kryptering och nyckeldistribution har gjorts är nästa steg att välja en lämplig krypteringsalgoritm samt göra en mer detaljerad specifikation av hur nyckelhanteringen skall gå till. Även här kommer förslagsvis 3 krypteringsalgoritmer att studeras och utvärderas utifrån specificerade krav och önskade egenskaper. Kriterier som spelar in är exempelvis snabbhet, säkerhet, nyckellängd, samt enkelhet vid implementation. En algoritm skall slutligen väljas och implementeras.

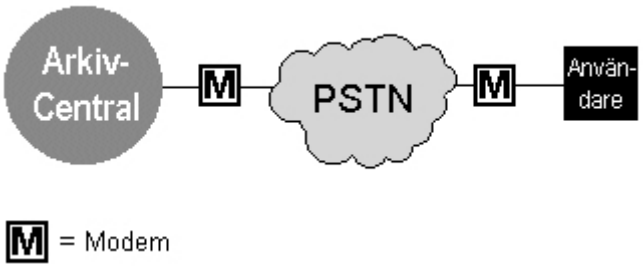


Figur 12: Processen för framtagning av lämplig krypteringsalgoritm

Arbetet skall ge insikt i vilken nivå av säkerhet som behövs för specifika miljön. Om ett färdigt överföringssystem skall användas skall kan detta arbete även fungera som en guide till att välja ett lämpligt sådant system.

4 Metod för kryptering och nyckeldistribution

Kommunikationen mellan användare och arkivcentralen ser ut på följande sätt:



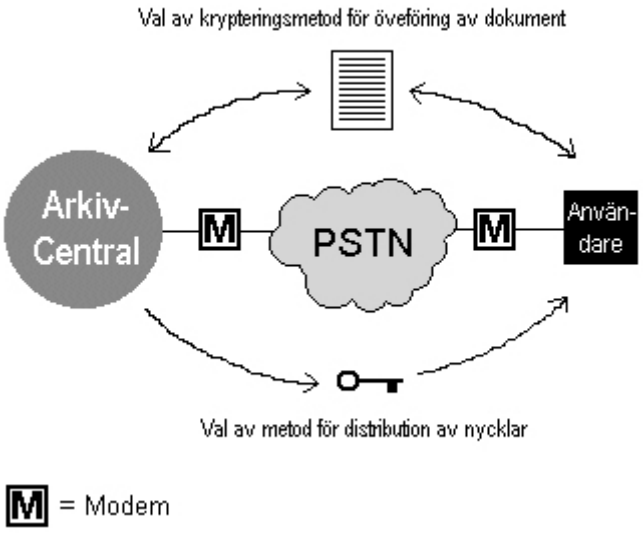
Figur 13: Kommunikationen mellan arkivcentralen och användare

Uppkoppling sker via telenätet mha modem. Det är via denna uppkoppling de elektroniska dokumenten sedan skall skickas. Olika övergripande metoder för en säker och lätthanterlig överföring av dokument skall tas fram. Detta innefattar metoder för kryptering av data, samt metoder för distribution av nycklar. Utvärdering av olika egenskaper för de olika metoderna kommer att göras. Dessa utvärderingar kommer att ligga till grund för valet av lämplig metod.

Utvärderingarna kommer att vara baserade på bakgrundskunskap som är inhämtad från böcker samt även från Internet. De uppgifter som är av sådan karaktär att de exempelvis föråldras snabbt bör hämtas från Internet, samtidigt som den mest grundläggande informationen om exempelvis vissa kryptosystem kan inhämtas från tryckt litteratur. Att använda sig av böcker är ett bra sätt att snabbt få en bra kunskaplig överblick. Internet är ett bra media då specifika uppgifter behöver hittas snabbt, samtidigt som informationen där oftast är nyare än i exempelvis böcker. Detta är bra då området kryptering ständigt utvecklas.

4.1 Möjliga lösningar/metoder

Problemet är att dels finna en lämplig metod för kryptering av de elektroniska dokumenten och dels att hitta ett bra sätt att hantera distribution av nycklar. Kryptering av dokument måste ske för att förhindra att obehöriga personer får tillgång till informationen i dessa dokument. I de kryptosystem, som kan tänkas användas till ett problem av detta slag, används så kallade nycklar för att kryptera data. Hur dessa nycklar skall förmedlas mellan arkivcentralen och användare är ett problem. Även denna hanteringsprocess för distribution av nycklar måste uppfylla vissa krav så som exempelvis säkerhet, snabbhet och enkelhet.



Figur 14: Val av metoder för överföring av dokument och nycklar

En användare skall inte behöva lägga ner mycket arbete och tid för att hantera nycklar. Nycklar ingår endast som en komponent i kryptosystemet som är till för att överföringen skall vara säker. Helst skall denna säkerhetsmodul endast utgöra en för användaren osynlig del av själva dokumentöverföringsprocessen.

4.1.1 Metoder för kryptering av dokument

Här finns det två olika huvudtyper av krypteringsmetoder för data, att studera. Den ena typen är de metoder som använder samma nyckel för kryptering som dekryptering; så kallade symmetriska kryptosystem. Den andra typen av system är osymmetriska system där olika nycklar används för kryptering och dekryptering.

De olika metoderna för kryptering kommer att diskuteras och analyseras med avseende på vilka generella egenskaper de har. Dock kommer endast de egenskaper som är intressanta för just denna typ av problem att tas upp. Bakgrundsinformation till diskussion kring dessa metoder är hämtad från kap. 2 och dess referenser.

4.1.1.1 Symmetriska kryptosystem

Symmetriska system innefattar de första typerna av krypteringssystem. De är relativt enkla i sin utformning. Algoritmerna för kryptering är överlag relativt små och lätta att implementera.

Att symmetriska kryptosystem är enkla och små gör att en algoritm av denna typ blir väldigt snabb. Detta kan vara väldigt värdefullt om kryptering görs ofta och när det rör sig om stora datamängder.

De symmetriska kryptosystem som används idag är trots sin enkelhet väldigt effektiva i sin uppgift att dölja klartexten i ett dokument för obehöriga personer. Med andra ord är de väldigt säkra system. Hemligheten med dess säkerhet ligger delvis i längden på nycklarna. Antalet nyckelkombinationer är så stort att det inte är möjligt att inom en rimlig tidsrymd, eller för en rimlig summa pengar, prova alla nyckelkombinationer. Om en person har obegränsat med pengar och således kan investera tillräckligt mycket i en utrustning som har till uppgift att hitta en nyckel, kan denna person naturligtvis göra detta inom en rimlig tidsrymd så som en dag eller vecka. Vanligtvis kostar denna utrustning mer än vad informationen i det hemliga dokumentet är värd.

Med detta som grund och baserat på vad dokumenten är värda är det möjligt att hävda att vissa kryptosystem är helt säkra.

Flera symmetriska kryptosystem har funnits en längre tid och är väl specificerade och dokumenterade. De har varit i praktiskt bruk en längre tid och i flera olika sammanhang. De har således också hunnit bli utsatta för många olika tester som har till uppgift att undersöka säkerheten hos dem. Även ingrepp från obehöriga personer, som försöker komma åt den hemliga informationen, har gjorts. Detta gör att tillförlitligheten för dessa specifika kryptosystem är väldigt hög. Två exempel på system av detta slag är DES och IDEA som är standardiserade och väl specificerade metoder för symmetrisk kryptering av data.

Det finns inte bara fördelar med symmetriska kryptosystem. Ett problem hos dessa system är hanteringen av nycklar. Sändare och mottagare använder samma nyckel för kryptering och dekryptering. Hur skall distribution av nycklar ske på ett smidigt och säkert sätt?

Det är som tidigare nämnt relativt lätt att implementera algoritmer av denna typ. Många olika typer av algoritmer finns även färdigimplementerade och gratis att hämta över Internet. Tillgängligheten för produkter av detta slag är således hög.

4.1.1.2 Osymmetriska kryptosystem

Osymmetriska kryptosystem är en nyare typ av system för kryptering av data. De första typerna av osymmetriska system kom till i mitten av 70-talet [Gar95]. Dessa är till skillnad från symmetriska kryptosystem, som använder binära operationer vid kryptering, mer komplexa i sin utformning. För att kryptera använder de sig av matematiska algoritmer som är av en mer komplicerad natur.

På grund av de långa matematiska beräkningar som måste göras vid kryptering och dekryptering är dessa system betydligt långsammare än symmetriska kryptosystem.

Säkerheten i osymmetriska system baserar sig på den matematiska svårigheten att faktorisera stora tal. Det finns idag ingen bra eller effektiv metod att göra detta på. De metoder som används för att faktorisera ett tal tar mycket lång tid för större tal. Således är det möjligt att konstatera att dessa typer av kryptosystem är mycket säkra såvida inga nya bra metoder för faktorisering hittas.

Då dessa system innehåller fler och mer avancerade matematiska beräkningar är de även något mer komplicerade att implementera. Det finns även några standardiserade och mer beprövade osymmetriska system. RSA är kanske det mest kända av dessa. Många av de företag som har tagit fram och implementerat de mest kända systemen, som exempelvis RSA tillåter ej att de distribueras och kopieras gratis för allmänt bruk. Vissa bolag tillåter dock privatpersoner att använda deras programvaror utan någon avgift. Om önskemålet är för ett företag att använda kända system gratis och legitimt, blir det svårt att finna ett sådant system på Internet eller något annat ställe kostnadsfritt. Tillgängligheten för dessa system är därför lägre än för symmetriska system.

En stor fördel med osymmetriska system är dess egenskaper vad gäller distribution av nycklar. Eftersom olika nycklar används för kryptering och dekryptering är det, till skillnad från symmetriska kryptosystem, riskfritt att skicka krypteringsnyckeln till sändaren, eftersom denna nyckeln ändå inte kan användas för att dekryptera meddelandet. Det gör inget om någon obehörig får tillgång till denna nyckel. Osymmetriska system lämpar sig väldigt bra för distribution av nycklar i en miljö där

varje användare har kontakt med många andra användare. Att distribuera nycklar för symmetriska system för en sådan miljö blir väldigt komplext.

4.1.2 Metoder för distribution av nycklar

Om ett osymmetriskt kryptosystem används för att skydda de elektroniska dokument som skall skickas över telenätet uppstår inga direkta problem vad beträffar säkerheten vid distribution av nycklar. Den öppna nyckeln, dvs krypteringsnyckeln, kan helt enkelt skickas över uppkopplingen som går över PSTN utan att hänsyn behöver tas till att någon obehörig person får tag på denna nyckel.

Annorlunda blir det om ett symmetriskt kryptosystem används. Skulle krypteringsnyckeln skickas över på samma sätt och en obehörig person får tag på denna nyckel innebär detta att denna person även kan läsa det hemliga elektroniska dokumentet, förutsatt att personen ifråga vet vilken krypteringsmetod som används. Det skall även tilläggas att processen för att tyda den analoga information som skickas via modem är väldigt komplicerat och utrustningen som behövs för detta är dyr.

Att kryptera denna nyckel innan den skickas över skapar ytterligare ett problem vad gäller nyckelöverföringen, dvs ytterligare en nyckel skapas för att kryptera den första. Ett annat medel för överföring och distribution av nyckel måste alltså användas. Nedan finns flera alternativa metoder för att överföra nycklar mellan arkivcentralen och användare. Detta urval av metoder är baserat på att de är de mest vanligt förekommande och mest lättillgängliga, dvs de är vedertagna metoder för distribution av nycklar. Några fler tänkbara metoder finns egentligen inte som är rimliga att använda för denna problemdomän.

Det finns inte mycket tryckt material eller publicerad information om dessa olika metoder för nyckeldistribution. Bakgrundsinformationen till dessa metoder vid diskussion och utvärdering är därför oftast begrepp av vedertagen karaktär. Information från kap. 2 med dithörande referenser används också.

4.1.2.1 Via användarens uppkoppling mot arkivcentralen

Det kanske enklaste och snabbaste sättet att distribuera nycklar mellan arkivcentral och användare är att använda sig av den befintliga uppkopplingen via telenätet. Överföring och installation av nyckel sker då automatiskt av överföringsprogrammet och nycklar slipper således hanteras av användaren. Systemet blir väldigt bekvämt och lätthanterligt för kunden vilket är väldigt bra.

Att enbart skicka över nyckeln i dess befintliga format ger en relativt dålig säkerhet. Om någon skulle lyssna av linjen och tolka datan, vilket är en komplicerad och dyrbar process, och således får tillgång till nyckeln kan vederbörande troligtvis även dekryptera meddelandet utan alltför stora svårigheter. Att använda sig av kryptering är då nästan onödigt. Om osymmetrisk kryptering används blir dock säkerheten hög eftersom det inte går att härleda dekrypteringsnyckeln från krypteringsnyckeln och det är just denna krypteringsnyckel som distribueras över telenätet.

Om istället nyckeln, som skickas över uppkopplingen, krypteras fås en betydligt högre säkerhet i överföringssystemet. Nyckeln kan exempelvis krypteras mha en kortare nyckel eller lösenord som användaren själv har hittat på.

Överföring av nyckel blir mycket snabb och enkel vilket är en fördel, även då snabbheten i systemet kanske inte är av högsta vikt. Detta gör även att distribution av nycklar kommer att utgöra en väldigt låg kostnad av systemet. Implementering av systemet kommer inte heller att innebära speciellt mycket arbete eftersom de

algoritmer som krävs för att utföra kryptering av nycklarna, mha användarens korta lösenord, inte är avancerade.

4.1.2.2 Brev

En alternativ metod för att förmedla nycklar mellan arkivcentralen och användare är att helt enkelt skicka ut den med vanlig post. Nyckeln finns då nedskriven på ett papper som användaren sedan manuellt får skriva in i datorn på begäran av dokumentöverföringsapplikationen.

Detta system kan anses vara relativt osäkert. Om någon obehörig person verkligen vill ta reda på vad lösenordet är så är det inte alltför svårt att stjäla detta brev. Det är troligtvis lättare och framför allt billigare än att tolka ett meddelande som sänds över PSTN. Om dock säkerheten för en användare eller kund inte är mycket viktig kan systemet erbjuda en acceptabel säkerhetsnivå.

En stor nackdel är att systemet är relativt långsamt. Att skicka ett brev inom Sverige med normal post tar minst en dag. När användaren väl får brevet och matar in nyckeln i datorn behöver dock han eller hon vid framtida överföringar inte tänka på eller bry sig om hantering av nycklar, tills den dag det kan vara aktuellt att byta nyckel.

Denna metod för nyckelöverföring är en väldigt enkel lösning på problemet, dvs ingen avancerad procedur för nyckelöverföring behöver implementeras. En nackdel är att användaren tvingas att handskas med nycklar, som är en liten del av dokumentöverföringssystemet och som helst skall vara osynlig för användaren. Dessa nycklar är vanligtvis även väldigt långa, exempelvis från 17 siffror för DES och 39 siffror för IDEA. En nyckel som är så lång är naturligtvis obekväm för användaren att mata in och det är lätt hänt att skriva in fel siffror. Det kan även vara en positiv faktor att användaren måste skriva in nyckeln för hand. Att användaren får se nyckeln med egna ögon kan skapa ett förtroende hos användaren för säkerheten hos systemet.

Systemet för distribution av nycklar blir väldigt enkelt att implementera eftersom själva överförandet inte ligger i applikationen. Överföringen blir däremot relativt kostsam eftersom brev, kuvert, porto, samt arbete med detta behövs.

4.1.2.3 Epost

En annan nyckelöverföringsmetod är att skicka krypteringsnycklarna via Epost. På samma sätt som med ett vanligt brev finns nyckeln nedskriven i det elektroniska brevet och användaren får sedan manuellt skriva in nyckeln.

Epost är tillskillnad från vanlig post ej helt osynlig från övriga obehöriga personer. Det är möjligt att med inte alltför stora svårigheter läsa av den trafik som finns i ett nätverk och således också andra personers Epost. Denna metod för distribution av nycklar kan därför anses ha en relativt låg säkerhet såvida inte brevet krypteras.

En fördel med detta system är dock att det går väldigt snabbt att skicka nyckeln. Mottagaren får brevet i nästan samma ögonblick som sändaren skickar iväg det.

Denna metod är också en ganska enkel lösning på problemet för överföring av nycklar. Användaren tvingas även här att handskas med nycklar som då både har positiva och negativa sidor. Att manuellt skriva mata in nyckel innebär en del arbete för användaren men det behöver endast göras vid byte av nyckel vilket sker vid ett fåtal tillfällen.

En stor brist med detta sätt att distribuera nycklar är att inte alla har tillgång till Internet och således en Epostadress. Det är alltså inte möjligt att använda den här metoden för alla användare.

Detta system har ungefär samma egenskaper vad beträffar enkelhet för användaren och enkelhet vid implementation. Kostnaden för distribution blir däremot väldigt låg.

4.1.2.4 Besök

En representant från den organisation eller företag som ansvarar för arkivcentralen kan besöka användaren/kunden och leverera nyckeln. Vid besöket installeras även nyckeln i systemet.

Detta sätt att distribuera nycklar är en mycket säker metod eftersom budbäraren garanterar att ingen obehörig kan ta del av någon information om nyckeln, förutsatt att denne person är lojal mot systemet.

Denna metod eller tjänst tar relativt lång tid att utföra vilket gör systemet långsamt. Om flera användare behöver tillgång till varsin nyckel samtidigt kan väntetiden bli lång.

Det är en enkel lösning på problemet men naturligtvis också en väldigt kostsam lösning. Positivt är att användaren slipper hantera nycklar. Att en budbärare levererar och installerar nyckeln ger ett bra intryck hos kunden.

4.1.2.5 Telefon

Att via ett telefonsamtal mellan arkivcentral och användare muntligt leverera nyckeln är ytterligare en metod som liknar metoden att leverera via brev eller Epost.

Den måste nog anses vara en metod med låg säkerhet eftersom det kan vara svårt att garantera sig om att det är rätt person som arkivcentralen pratar med och således levererar krypteringsnyckeln till. Det är dessutom lättare att lyssna och tolka ett telefonsamtal än att tolka ett elektronsikt meddelande som sänds över telenätet via modem. Om detta system skall kunna användas måste speciella rutiner inrättas, som har till uppgift att verifiera användaridentitet, för att säkerheten skall kunna uppnå en rimlig nivå.

Systemet är snabbt förutsatt att användaren kontaktar arkivcentralen vid rätt tidpunkt. Om full tillgänglighet skall uppfyllas måste det vara möjligt för användaren att kontakta arkivcentralen via telefon dygnet runt under alla dagar på året. Detta är naturligtvis kostsamt. Systemet är enkelt att implementera eftersom ingen avancerad procedur för nyckelöverföring behöver implementeras. Nackdelen är att användaren tvingas hantera nycklar.

4.2 Kriterier för val av metod

Vad skall val av metod för kryptering samt system för nyckelöverföring vara baserat på? Nedan följer ett antal kriterier som bör vara uppfyllda för de metoder som väljs. Kriterierna innefattar dels egenskaper för systemet som berör användaren men även detaljer som spelar in vid implementering och installation av systemet.

Västgöta företagsarkiv AB är ett företag som specifikt efterfrågat ett system av denna typ. VFAB ställer vissa krav på det önskade systemet. Det är utifrån dessa krav som följande kriterier i detta kapitel är baserade på. Även då de från början avser att användas för VFAB’s problemdomän så är de framtagna kriterierna av en sådan generell karaktär att de även fungerar för andra typer av organisationer.

4.2.1 Kryptering av dokument

För varje typ av kryptosystem har tidigare egenskaper för dessa redovisats. Här kommer olika egenskaper hos det önskade systemet som hanterar kryptering av de elektroniska dokumenten att diskuteras. Kraven eller de önskade egenskaperna kan vara av varierade grad. Exempelvis mindre viktigt eller mycket viktigt.

4.2.1.1 Säkerhet

Det viktigaste vid val av lämpligt kryptosystem är säkerheten. Idén med att förvara dokument i ett centralt arkiv är delvis att kunna erbjuda en hög säkerhet för förvaring av dokument. Således är det även mycket viktigt att säkerheten vid dokumentöverföringen är väldigt hög. Dvs ingen obehörig person skall kunna få tillgång till dokumenten.

4.2.1.2 Snabbhet

Naturligtvis är det mycket önskvärt att överföringsproceduren och därmed krypteringssystemet arbetar snabbt. En överföring av dokument mellan användare och arkivcentral sker dock relativt sällan. De dokument som lagras i arkivet är oftast inga dokument som användaren arbetar med frekvent, utan utgör den typen av dokument som enbart skall förvaras en specifik tid framöver. Eftersom access till arkivcentral i genomsnitt sker sällan är det därför inte av största vikt att snabbheten hos överföringsproceduren är väldigt snabbt. Viktigare är som tidigare nämnts att säkerheten är högre.

4.2.1.3 Enkelhet för användare

Användaren utgör en kund till det företag som tillhandahar arkivcentralen. Därmed är det mycket viktigt för företaget att systemet är lätt att använda för kunden. Är systemet komplicerat för användaren är det troligt att befintliga kunder byter leverantör samtidigt som det kan vara svårt att få nya kunder. Att systemet är enkelt innebär att användaren allra helst inte skall behöva befatta sig med själva krypteringsproceduren. Denna procedur skall skötas automatiskt av överföringssystemet och enbart ingå som ett delsystem i detta. Att användaren ej skall behöva hantera nycklar är ett exempel på hur bidrag kan göras för att kryptosystemet ska vara enkelt att använda för användaren.

4.2.1.4 Enkelhet vid implementation

Det är alltid bra om en funktion är lätt att implementera. Mycket arbete innebär en hög kostnad i form av tid och pengar, men dock utgör denna kostnad troligtvis endast en liten del av den framtida omsättningen av utgifter och inkomster i företaget. En balansgång får göras mellan kostnaderna för utveckling av systemet och vad framtida inkomster av detta system kan tänkas vara. Enkelhet vid implementation av kryptosystemet kan således sägas utgöra ett mindre viktigt krav.

4.2.1.5 Kostnad

Med kostnad menas vad det kostar i tid eller pengar att köpa ett färdigt system eller själv konstruera det. Ofta går det för vissa system att legitimt få tag på färdiga implementationer av kryptosystem gratis, exempelvis via Internet. Det kan vara fördelaktigt ur en säkerhetsaspekt att använda sig av väl publicerade och väl använda system eftersom dessa har utsatts för mycket testning. Att använda sig av välkända standardiserade system utgör även en trygghet för kunden. Om möjligheten finns att

få tag på färdigimplementerade standardiserade system gratis är detta naturligtvis en stor fördel eftersom höga kostnader för att ta fram ett system besparas samtidigt som ett säkert kryptostystem fås.

4.2.2 Distribution av nycklar

Om en osymmetrisk metod väljs för kryptering av data löser sig problemet med nyckeldistribution, dvs nyckeln kan skickas över samma anslutning som det elektroniska dokumentet utan några problem. Däremot måste ett alternativt system för överföring av nycklar beaktas vid ett symmetriskt kryptosystem för att nyckelöverföringen skall uppfylla de krav och egenskaper som önskas. Oavsett vilken typ av kryptosystem som väljs skall nedanstående kriterier för nyckeldistribution vara uppfyllda.

4.2.2.1 Säkerhet

För att säkerheten hos hela dokumentöverföringssystemet skall vara hög är det viktigt att varje komponent i systemet har bra säkerhet. Om nyckeln vid överföringen avslöjas inför en obehörig person faller hela säkerheten vid krypteringen och därmed hela dokumentöverföringen. Hög säkerhet vid distribution bör därför anses som mycket viktig.

4.2.2.2 Snabbhet

Överföring av nyckel kommer troligtvis att utföras ett fåtal gånger beroende på omständigheterna. Om exempelvis situationen uppstår att någon obehörig person på kontoret hos kunden kanske har kommit i kontakt med nyckeln på något vis kan det vara lämpligt att byta nyckel. Eftersom nyckelöverföring sker sällan är det inte av högsta vikt att den måste vara snabb, även om det naturligtvis är önskvärt.

4.2.2.3 Enkelhet för användaren

Som tidigare sagts utgör användaren en kund till det företag som tillhandahar arkivcentralen. Därmed är det mycket viktigt för företaget att systemet är lätt att använda för kunden. Ett önskemål är att användaren ej skall behöva hantera nycklar. Om så ändå är fallet är det viktigt att denna hantering sköts på ett enkelt och snyggt sätt. Att hela systemet ger ett snyggt intryck hos användaren är viktigt för att företaget skall kunna skapa ett bra förtroende hos kunden. Att systemet för nyckeldistribution är enkelt för användaren att hantera är därför väldigt viktigt.

4.2.2.4 Enkelhet vid implementation

Det är alltid bra om en funktion är lätt att implementera av samma orsak som enkelhet vid implementation av ett kryptosystem. Mycket arbete innebär en hög kostnad i form av tid och pengar. Det är bra om nyckelöverföringssystemet är enkelt att implementera men detta kan ändå sägas utgöra ett mindre viktigt krav.

4.2.2.5 Kostnad

Att distribuera nyckel till användarna i systemet är inte gratis. Kostnad beror på vilken typ av metod som väljs för överföring av nyckel. Att skicka nyckel över telenätet kostar exempelvis betydligt mindre än att använda sig av en kurir. Kostnaden för nyckelöverföringen är proportionell mot säkerheten och tiden för överföring. En avvägning får göras vad som anses vara mest viktigt.

4.3 Val av metod

Ett val av generell typ av metod för kryptering samt metod för distribution av nycklar måste göras. Detta val kommer att vara baserat på egenskaper hos de olika typer av system som redovisats tidigare och krav (se kap. 4.2.1 och kap. 4.2.2) på egenskaper hos systemen. Om inte alla krav och önskemål kan uppfyllas hos en metod skall det alternativ väljas som bäst matchar de egenskaper som önskas hos systemet.

Två tabeller har införts för att få en bättre överblick över vilka metoder som har respektive egenskaper. Det finns 2 typer av system för kryptering samt 5 olika metoder för nyckeldistribution. Därför finns det 10 alternativa metoder som skall utgöra en lösning för kryptering och nyckelöverföring. De 5 första metoderna har alla gemensamt att de innehåller ett symmetriskt kryptosystem, medan det i metoderna 6 till 10 används ett osymmetriskt kryptosystem.

Varje egenskap, som finns definierad i den vänstra kolumnen i tabellerna, är kopplad till varje metod. Dvs, för varje metod beskrivs hur väl varje egenskaps uppfylls genom att betygsätta med ’-’, ’0’, eller ’+’. Bedömningen fungerar så att ’+’ indikerar att egenskapen existerar på ett positivt sätt hos metoden, ’-’ indikerar att egenskapen eller kravet existerar på ett negativt sätt i metoden medan ’0’ är ett mellanting, dvs varken bra eller dåligt. Bedömningarna är baserade på de kriterier som tagits fram för det önskade systemet.

Metoder:	Symmetriskt kryptosystem				
	1. Telenätet	2. Brev	3. Epost	4. Besök	5. Telefon
Säkerhet:	+	-	+	+	-
Snabbhet kryptering:	+	+	+	+	+
Snabbhet nyckeldistr.	+	-	0	-	0
Enkelhet för användaren	+	-	-	+	-
Enkelhet vid implement. kryptosyst.	+	+	+	+	+
Enkelhet vid implement. nyckeldistr.	0	+	+	+	+
Kostnad kryptosyst	+	+	+	+	+
Kostnad nyckeldistr.	+	0	+	-	-

Tabell 2: Metoder med symmetriskt kryptosystem

Metod för kryptering och nyckeldistribution

Metoder:	Osymmetriskt kryptosystem				
	6. Telenätet	7. Brev	8. Epost	9. Besök	10. Telefon
Säkerhet:	+	+	+	+	+
Snabbhet kryptering:	-	-	-	-	-
Snabbhet nyckeldistr.	+	-	0	-	0
Enkelhet för användaren	+	-	-	+	-
Enkelhet vid implement. kryptosyst.	-	-	-	-	-
Enkelhet vid implement. nyckeldistr.	+	+	+	+	+
Kostnad kryptosyst	-	-	-	-	-
Kostnad nyckeldistr.	+	0	+	-	-

Tabell 3: Metoder med osymmetriskt kryptosystem

Om tabellerna studeras går det att utläsa att det finns flest fördelar med de metoder som använder sig av symmetriskt kryptering, utifrån de kriterier som tagits fram, varpå det kan vara lämpligt att välja någon av metoderna 1 till 5. Det viktigaste kravet eller egenskapen hos den önskade metoden är som tidigare nämnts att systemet kan visa en hög säkerhet. Därmed kan de alternativ eller metoder som använder sig av osymmetrisk kryptering eller som har låg säkerhet uteslutas. Då återstår endast metod 1, 3 och 4 vilka använder sig av symmetriskt kryptering.

Metod 1 använder sig av distribution av nyckel via uppkoppling över telenätet, metod 3 använder sig av Epost som medel för nyckelöverföring, medan metod 4 använder sig av en nyckelkurir. Den största skillnaden mellan dessa metoder är att metod 4 blir betydligt mer kostsam eftersom ett besök från arkivcentralen måste göras hos användaren för att överlämna installera nyckel.

Med vetskap om dessa egenskaper väljer jag därför att använda mig av metod 1 för att ta fram ett system för kryptering och nyckelöverföring. Denna metod erbjuder en hög säkerhet vid kryptering och nyckelöveföring samtidigt som dess algoritm för kryptering är snabb. Att lyssna av och tolka data som skickas mellan två modem över PSTN är en svår och dyrbar process, vilket bidrar till en hög säkerhet. Nyckelöverföringen blir mycket snabb eftersom den sker direkt över uppkopplingen. Det innebär inte alltför mycket arbete eller problem för användaren att endast mata in sitt lösenord i överföringsprogrammet eftersom detta lösenord är relativt kort och lätt att komma ihåg då användaren själv har bestämt det. Då uppkopplingen används för distribution innebär detta att kostnaden för överföring av nyckel blir väldigt liten, till skillnad från att exempelvis använda sig av en kurir. Systemet för kryptering är enkelt att implementera. Många färdigimplementerade system finns dessutom att hämta

gratis via Internet. Kostnaden för detta system blir därför relativt låg. Möjligheten finns att välja ett känt standardiserat system som är väl testat vad gäller dess säkerhet. Denna metod anser jag passa bäst för de krav som systemmiljön och inblandade personer kräver.

5 Precisering av valda metoder

Problemet med kryptering av elektroniska dokument och distribution av nycklar skall lösas genom att använda en symmetriskt metod för kryptering, samt genom att använda sig av uppkopplingen via telenätet för nyckeldistribution. Det finns för respektive metod olika sätt att implementera dessa metoder. Avsikten med detta kapitel är att finna lämpliga algoritmer och tillvägagångssätt för att praktiskt kunna implementera och realisera lösningsförslagen.

5.1 Precisering av krypteringsmetod

En lämplig algoritm för kryptering av de elektroniska dokumenten måste väljas. Det finns många existerande algoritmer för symmetrisk kryptering av data, eftersom denna metod med två privata nycklar är den äldsta formen av kryptering som använder sig av nycklar. Önskvärt är att välja en algoritm som är väl beprövad och som uppnår en acceptabel säkerhetsnivå. Det är även lämpligt att välja en algoritm som är fri för allmänt bruk och som således kan distribueras och inhämtas på exempelvis Internet utan kostnad.

Tillvägagångssättet för val av metod kommer att ske i olika faser. Förslagsvis tre algoritmer kommer att väljas för närmare studie med kravet att de är väl kända och förhoppningsvis lämpliga ur säkerhetssynpunkt. Olika kriterier för val av algoritm kommer sedan att redogöras för att få fram vilka egenskaper hos algoritmen som är viktiga för valet.

5.1.1 Presentation av lämpliga krypteringsalgoritmer

I detta kapitel kommer ett urval av algoritmer för symmetrisk kryptering av data att presenteras. Det finns, som nämnts tidigare, väldigt många algoritmer för symmetrisk kryptering. Detta urval baserar sig på att de är några av de mest välkända och mest använda algoritmer inom området kryptering. Förutom DES, varianter av DES och IDEA är det i stort sett inte några andra algoritmer, som används för symmetrisk kryptering, som används i någon större utbredning [Med97].

Dessa har olika egenskaper vad gäller exempelvis längden på nycklar, snabbhet, säkerhet, ålder etc. De egenskaper som är relevanta för beslut om vilken algoritm som skall användas vid implementation, kommer att presenteras och diskuteras.

5.1.1.1 DES

DES är en metod eller algoritm som är relativt gammal och väl använd. Den skapades redan 1977 [Rik85]. Vid den tiden fanns ej tillgång till samma hårdvara vad gäller processorkapacitet. Dagens datorer är betydligt snabbare. Algoritmen som använder sig av en nyckel med 56 bitar var fullt tillräcklig ur säkerhetssynpunkt vid den tidpunkt som den skapades. Med dagens datorer innebär dock en nyckellängd på 56 bitar inte att någon högre nivå på säkerheten, varför algoritmen idag kanske kan anses vara lite föråldrad [McK98]. Trots detta används algoritmen fortfarande i många existerande system idag, men inte vid tillverkning av nya system. DES tillhör dessutom en av de snabbaste algoritmerna för kryptering.

DES algoritmen har knäckts genom nyckelforcering. Första tillfället var i juni 1997 då en tävling hade utlysts för att knäcka algoritmen. Det tog 4 månader att hitta rätt nyckel [Rds97]. Den senaste gången någon knäckte algoritmen var i februari 1998. Då

tog det endast 39 dagar att knäcka den krypterade texten [Rds98]. Säkerheten på krypteringen kan liknas vid ett förseglat kuvert som innehåller meddelandet [McK98].

Denna algoritm är relativt simpel i sin funktion och således inte alltför svår att implementera. Eftersom algoritmen dock är välkänd och väl använd är den väl distribuerad och är relativt lätt att få tag på gratis via Internet. Att den har funnits och varit i bruk så länge gör den väldigt pålitlig. Det är möjligt att ganska säkert bedöma hur säker den är.

5.1.1.2 Triple-DES

Triple-DES är ett bra alternativ till DES om en väldigt hög säkerhet krävs. Den använder helt enkelt, som namnet indikerar, DES-algoritmen tre gånger med tre olika nycklar för att kryptera ett meddelande. Med andra ord använder sig Triple-DES algoritmen av en nyckel med längden 168 bitar. Detta anses öka säkerheten kraftigt även om det kanske fortfarande inte kan bevisas [Hus97]. Säkerheten för denna algoritm kan dock liknas vid ett kassaskåp, till skillnad från DES vars säkerhet kunde liknas vid ett förseglat kuvert [McK98].

Algoritmen har inte existerat fullt så länge som DES varför den inte är lika pålitlig i sin funktion. Den har helt enkelt inte hunnit bli utsatt för lika mycket tester som exempelvis DES. Eftersom den enbart består av DES med modifikationen att kryptering sker tre gånger för varje dokument anses den dock vara relativt pålitlig och säkerheten anses kunna uppskattas. En nackdel är att det tar tre gånger längre tid att kryptera en datamängd med denna algoritm.

Fördelen med denna algoritm är att det endast krävs en liten modifikation av den vanliga DES algoritmen för att uppnå denna trefaldiga säkerhet. Detta gör den lätt att implementera i synnerhet om tillgång finns till DES. Den har ytterligare en fördel vad beträffar lagar och förordningar. Enligt amerikansk lag är det olagligt att exportera krypteringsalgoritmer av högre styrka till andra länder. DES med sin 56 bitars nyckel räknas inte som en algoritm av detta slag. Algoritmer som använder nycklar av större längd är däremot otillåtna för export. Triple-DES har praktiskt sett en nycklelängd av 168 bitar men består trots allt av tre delalgoritmer med nyckellängden 56 bitar varför den enligt amerikans lag är laglig att exportera [Med97]. Metoden kan därför ses som väl tillgänglig.

5.1.1.3 Fenced DES

Fenced DES är även den en variant av DES och är skapad av Terry Ritter [Rit98]. Den anses vara mer säker och även snabbare än Triple-DES. Jämfört med DES är den endast 1.2 gånger långsammare men erbjuder en säkerhet som uppskattas vara 4 gånger högre än DES [Rit94].

Algoritmen fungerar som så att den utför fyra DES krypteringar parallellt. Indatan för varje delberäkning är således ett block som är fyra gånger längre än det block DES använder sig av i sina beräkningar i vanliga fall. Utdatan från dessa fyra delkrypteringar påverkar internt varje bit i det stora blocket enligt ett specifikt mönster [Rit98].

Fenced DES är en algoritm som är relativt ny och inte speciellt utbredd eller beprövad. Även om algoritmen utnyttjar den beprövade säkerheten hos DES är det svårt att veta hur säker den egentligen är. Den är dessutom relativt svår att få tag på färdigimplementerad eftersom det inte finns någon utbredd distribution eller användning av denna. Även om denna algoritm är relativt enkel i sin uppbyggnad och

därmed inte utgör någon större svårighet vid implementation är det således svårt att få tag på en specifikation som visar hur algoritmen fungerar.

5.1.1.4 IDEA

IDEA är ett blockchiffer som arbetar på block av storlek 64-bitar. Nyckelstorleken är 128-bitar. Samma algoritm används för både kryptering och dekryptering. Den är enkel att implementera [Sch96]. IDEA använder sig av ett fåtal enkla beräkningar och dess kryptering är snabb och effektiv [Sch96]. Algoritmen anses vara väldigt säker och är patenterad i USA och större delen av Europa. För kommersiella ändamål krävs därför licens [Hus97][Sch96].

Algoritmen togs fram 1992 och är relativt ny men används i stor utbredning eftersom den ingår som komponent i det välkända och väl använda kryptosystemet PGP. Det är svårt att uppskatta hur mycket det går att lita på dess säkerhet med tanke på hur ung algoritmen är, men med tanke på dess stora användning kan den ändå anses ha en någorlunda hög pålitlighet. Dessutom har den genomgått en omfattande kryptoanalys av både akademiker och militära enheter, men ingen har yttrat något om att det skulle finns några brister eller svagheter i algoritmen [Sch96]. Säkerheten hos denna algoritm anses trots allt av många kunna liknas vid säkerheten hos ett kassaskåp [McK98].

IDEA är en väldigt snabb algoritm. Närmare bestämt ungefär dubbelt så snabb som DES [Sch96]. En mätning av dess snabbhet har gjorts av ett företag som heter *r³ security engineering*. Mätningen är gjord på deras implementation i C-kod av algoritmen. Resultatet för kryptering av data på olika datorer visas nedan i tabellen [Sch96].

VAX 8650	430 kbit/sekund
486DX2-66	1700 kbit/sekund
Pentium90	4600 kbit/sekund

Tabell 4 : Snabbheten för IDEA-algoritmen på olika plattformar.

5.1.2 Kriterier för val av algoritm

Valet av vilken algoritm som kommer att användas för implementering av krypteringsalgoritmen kommer att vara baserat på ett antal kriterier. I detta kapitel kommer olika kriterier som spelar in vid val av algoritm att diskuteras. Generellt gäller ungefär samma egenskaper och krav på algoritmen som kraven och önskemålen som ställes vid valet av krypteringsmetod. Kriterier som exempelvis säkerhet och snabbhet är ju krav som måste ställas på samtliga delar av systemet för att hela dokumentöverföringssytemet skall kunna erbjuda en bra säkerhet och snabbhet.

5.1.2.1 Säkerhet

Den viktigaste egenskapen hos dokumentöverföringssystemet är som nämnts förut att säkerheten skall vara hög. Innehållet i de elektroniska dokumenten skall inte kunna läsas av någon obehörig person. Troligtvis kommer större delen av de dokument som finns förvarade i arkivcentralen inte vara av något värde eller intresse för någon annan än den kund som äger dokumentet. Systemet måste ändå kunna erbjuda en bra säkerhetsnivå om så skulle vara fallet att det är av yttersta vikt att ingen obehörig person skall kunna få tag på den hemliga informationen.

Att de elektroniska dokumenten skall skickas över telenätet via modem medför i sig att säkerheten blir relativt hög eftersom det är svårt och kostsamt att lyssna av information som skickas över detta media. Om någon person ändå skulle göra detta krävs en bra algoritm med hög säkerhet för att informationen i dokumenten vid överföringen inte skall kunna hamna i orätta händer.

5.1.2.2 Pålitlighet

Önskvärt är att algoritmen är en känd och väl beprövad algoritm. Om algoritmen är helt ny finns det risk för att den kan innehålla brister som ännu ej har hunnit upptäckas. Samtidigt kan naturligtvis inte algoritmen vara för gammal eftersom den då troligtvis inte uppfyller de säkerhetskrav som ställs. Ett exempel på detta är DES som fortfarande är en mycket bra och pålitlig algoritm till system som inte kräver alltför hög säkerhet. Om ett nytt större system skall upprättas rekommenderas dock att inte använda DES [Hus97]. DES ligger på gränsen till att anses vara en algoritm med för dålig säkerhet.

5.1.2.3 Snabbhet

Det är även önskvärt att algoritmen har egenskapen att den är snabb, dvs att kryptera respektive dekryptera ett meddelande skall gå så snabbt som möjligt. Oftast innebär en säkrare algoritm att den blir något mer komplicerad och således tar längre tid att utföra en kryptering.

Detta är dock en egenskap som kanske inte är av högsta vikt att den uppfylls då överförandet av ett elektroniskt dokument i denna miljö i genomsnitt sker relativt sällan och då storleken på dokumenten oftast inte är av någon större karaktär.

5.1.2.4 Enkelhet vid implementation

Om det går snabbt att implementera systemet och således krypteringsalgoritmen spars både tid och pengar. Därför är det bra om algoritmen är så enkel som möjligt att implementera. Själva funktionaliteten hos applikationen är naturligtvis den viktigaste men om valet står mellan två algoritmer så är det fördelaktigast att den algoritm väljs som är lättast att implementera.

5.1.2.5 Tillgänglighet

Med tillgänglighet menas hur lätt det kan vara att få tag på en specifik krypteringsalgoritm gratis över exempelvis Internet. Om det är möjligt att få tag på en färdigimplementerad algoritm eller källkoden till en del av den kan mycket tid för implementationen besparas. För välkända och väl använda algoritmer brukar det oftast inte vara några större problem att få tag på dessa, såvida det inte finns lagar och restriktioner som förhindrar detta. Det är trots allt mycket onödigt att själv implementera en algoritm om den finns att hämta utan kostnad, färdig och klar för användning.

5.1.3 Val av Krypteringsalgoritm

Ett val av lämplig algoritm för att implementera krypteringsmetoden måste göras. Valet skall vara baserat på de olika kriterier som ställts samt hur väl dessa kriterier stämmer överens med egenskaper hos respektive algoritm. Som redogjorts för tidigare är det generellt säkerheten hos systemet och således algoritmen som är den viktigaste aspekten även om naturligtvis andra egenskaper spelar in. Bakgrundsfakta för varje algoritm har redovisatsi (se kap. 5.1.1) vad beträffar dess egenskaper så som

exempelvis snabbhet, säkerhet etc. En sammanställning av de olika egenskaperna hos de olika algoritmerna har gjorts i följande tabell för att få en bra överblick över situationen.

Metoder:	11. DES	12. Triple-DES	13. Fenced DES	14. IDEA
Säkerhet:	-	+	+	+
Pålitlighet:	+	+	-	0
Snabbhet:	+	-	+	+
Enkelhet vid implement:	+	+	-	+
Tillgänglighet:	+	+	-	-

Tabell 5: Metoder med osymmetriskt kryptosystem

När nya krypteringsalgoritmer tas fram påstås ofta att de är betydligt säkrare än vissa andra specifika existerande sådana. Eftersom det är omöjligt att mäta och specifikt ge ett mått på styrkan eller säkerheten hos en algoritm, är det för nya algoritmer svårt att verifiera att de verkligen är så säkra som de påstås [Rit98]. DES däremot, är en gammal välkänd och väl beprövad algoritm som anses fungerar bra och som det går att lita på, även om den numera, på grund av teknikens utveckling, inte längre kan uppvisa någon högre säkerhet. Att DES är en av de mest kända och beprövade algoritmerna för symmetrisk kryptering gör att det kan vara en bra idé att använda sig av DES som en delkomponent i det nya systemet [Rit98].

Säkerheten är högsta prioritet hos systemet varför DES-algoritmen kan uteslutas direkt. Pålitligheten är även den en ganska viktig faktor eftersom den är sammankopplad med säkerheten. Om algoritmen är opålitlig går det inte att påstå att den är säker. Kvar blir då Triple-DES och IDEA som uppfyller krav om hög säkerhet samt en godtagbar pålitlighet. Nackdelen med IDEA är just att en licens krävs för att få använda den kommersiellt, vilket kostar pengar. Dessutom är den som sagt en relativt ny algoritm som eventuellt kan visa sig inneha svagheter i framtiden.

Med vetskap om allt detta torde Triple-DES vara ett bra alternativ för att implementera krypteringsmetoden. Den är relativt långsam, men det är inte någon större nackdel eftersom en dokumentöverföring i genomsnitt sker så sällan. Den är väl beprövad, eftersom den är uppbygd av DES, och välkänd samtidigt som den kan erbjuda en hög säkerhet. Den är även helt legitimt att fritt använda sig av den för kommersiellt bruk. Således väljs Triple-DES som lämplig algoritm för implementering av krypteringsmetoden.

5.2 Implementation av krypteringsalgoritm

En prototyp av den algoritm som hanterar kryptering av de elektroniska dokumenten med krypteringsmetoden Triple-DES, har tagits fram. Som hjälp för framtagandet och implementationen av denna algoritm har en existerande DES-algoritm använts. Källkoden till denna algoritm är hämtad på Internet [Hus97]. Den modifierades först för att få en bättre funktionalitet, sedan ytterligare för att göra om den till en Triple-DES algoritm.

DES-algoritmen som är hämtad på Internet är skriven i C-kod och avsedd för UNIX-miljö. Källkoden till denna prototyp kan beskådas i bilagorna A till D. Det ursprungliga utseendet hos algoritmen uppvisade flera svagheter och brister i dess beteende och funktionalitet. Exempelvis skrevs den krypterade filen ut på skärmen, vilket inte leder till någon större praktisk användbarhet. I synnerhet då alla 255 tecken som används inte kan skrivas ut visuellt på skärmen. Tanken är att när kryptering sker av ett dokument, i form av en fil, så skall den krypterade datamängden lagras över det okrypterade dokumentet i den ursprungliga källfilen.

En modifikation av algoritmen har således gjorts för att kunna erbjuda denna funktion. Det elektroniska dokumentet läses från den angivna filen och krypteras sedan. En temporär fil skapas för att tillfälligt kunna husera det krypterade dokumentet. När krypteringen är klar flyttas innehållet i den temporära filen över till den ursprungliga källfilen som innehöll det okrypterade dokumentet. Efter att detta har utförts raderas den temporära filen. Förfarandet av de olika stegen är konstruerat så att det alltid existerar en fil innehållandes antingen hela dokumentet i klartext eller hela dokumentet i krypterat format. Detta som en säkerhet om något oförutsett skulle inträffa, så som exempelvis ett strömavbrott. Det vore ett stort missöde om exempelvis ett strömavbrott skulle inträffa när källfilen, innehållandes den ursprungliga klartexten, håller på att överlagras med den krypterande datamängden, utan att någon temporär fil existerar som innehåller hela det krypterade dokumentet. Då skulle dokumentet vara förlorat. Som prototypen fungerar nu kan alltid informationen räddas om något skulle gå fel. Felet måste naturligtvis vara av rimlig proportion.

Att omvandla DES-algoritmen till Triple-DES innebär att längden på nyckeln måste utökas till tre gånger dess ursprungliga längd, dvs från 56 bitar till 168 bitar, eller motsvarande från 8 bytes till 24 bytes. Denna nyckel delas sedan upp i tre nycklar. Således kan de ursprungliga DES-funktionerna användas genom att kryptera det aktuella dokumentet tre gånger i följd med tre olika nycklar. Modifikationen av DES-algoritmen för att få fram en Triple-DES algoritm innebär då inte mycket arbete.

När modifikation har gjorts på prototypen av DES-algoritmen så att önskad funktionalitet och beteende fås, samt även att kryptering med Triple-DES metoden kan erbjudas, ser en körning av programmet i UNIX-miljö ut på följande vis när kryptering skall göras på ett dokument:

```
oden:~/Xjobb/Triple-DES>more dokument
Detta är ett meddelande i klartext, dvs den har ej genomgått någon form av
kryptering.
oden:~/Xjobb/Triple-DES>des -e
Enter name of file: dokument
Enter key:
Enter key again:
Large key: qwertyuiopasdfghjklzxcvb
Session: 1
Session key: qwertyui
Session: 2
```

Session key: opasdfgh

Session: 3

Session key: jklzxcvb

oden:~/Xjobb/Triple-DES>more dokument

Æ×6• h»• ’n*—U‹d*ßàÅ¾• › ÖK¢Yu- e¦^-
†^ =• {TÎ·‡=«p• • e˘ :•_;ùèó®"áOqýx“‹:ÿŒ˜Š• ÑºC³ô

oden:~/Xjobb/Triple-DES>

På följande motsvarande sätt ser det ut när dokumentet skall dekrypteras tillbaka till sitt ursprungliga utseende mha prototypen:

oden:~/Xjobb/Triple-DES>des -d

Enter name of file: dokument

Enter key:

Enter key again:

Key mistyped, try again

Enter key:

Enter key again:

Large key: qwertyuiopasdfghjklzxcvb

Session: 3

Session key: jklzxcvb

Session: 2

Session key: opasdfgh

Session: 1

Session key: qwertyui

oden:~/Xjobb/Triple-DES>more dokument

Detta är ett meddelande i klartext, dvs den har ej genomgått någon form av kryptering.

oden:~/Xjobb/Triple-DES>

Detta är, som nämnts innan, enbart en prototyp av det delsystem som kommer att hantera kryptering av de elektroniska dokumenten. Inmatning av nyckel kommer exempelvis inte att ske för hand av användaren utan kommer att hanteras av applikationen. Användarens kontakt med detta delsystem kommer att vara på en högre nivå, dvs användaren kommer inte att behöva hantera nycklar eller komma i direkt kontakt med själva krypteringsproceduren över huvud taget.

Den nuvarande implementationen är fullt duglig i sin funktionalitet men kanske något långsam. Algoritmen behöver således effektiviseras för att göra beräkningarna snabbare. En effektivisering av algoritmen skulle påverka snabbheten markant då den

inbyggda DES-algoritmen används tre gånger för att kryptera ett dokument. Om dokumentet är stort är det viktigt att implementationen av algoritmen är så effektiv och snabb som möjligt, så att inte kryptering av dokumenten innebär onödigt mycket väntetid för användaren.

5.3 Precisering av metod för nyckelöverföring

Det val av metod som har gjorts, för överföring av nycklar, använder den befintliga uppkopplingen via telenätet. En närmare precisering av förfarandet av hur nyckeln skall förmedlas mellan användare och arkivcentral måste göras. För att skydda nyckeln från att bli exponerad för obehöriga personer då den skickas över uppkopplingen är det troligtvis nödvändigt att använda någon form av kryptering. Det finns två huvudsakliga alternativ till hur detta skall implementeras. De båda alternativa lösningarna kommer att presenteras och diskuteras. Därefter kommer ett val av en av dessa två lösningar att göras. Underlaget till beslutet av vilken lösning som anses lämplig kommer att vara baserat på vilka faktorer som är viktiga för den befintliga problemdomänen med avseende på de speciella krav den ställer, samt vad respektive lösning har för egenskaper och vilka möjligheter den kan erbjuda.

5.3.1 Osymmetrisk kryptering

En möjlighet till att kunna erbjuda en säker lösning för överförandet av nycklar mellan användare och arkivcentral är att använda sig av osymmetrisk kryptering. Dvs, en osymmetrisk krypteringsmetod används för att kryptera respektive dekryptera den nyckel som måste användas för att kunna kryptera eller dekryptera själva meddelandet.

Det finns egentligen bara en algoritm som kan anses erbjuda en acceptabel säkerhet och som är väl beprövad och väl använd. Denna algoritm är RSA [Hus97]. RSA är en relativt säker algoritm och rekommenderas av många då den har varit en väl utbredd och en väl använd osymmetrisk krypteringsalgoritm under en period på ca 20 år [Med97].

Nackdelen med RSA är att den är patenterad inom USA vilket påverkar tillgängligheten av den. Dessutom är det med de amerikanska lagarna om kryptering förbjudet att exportera algoritmer med en nyckellängd som överskrider 56 bitar. Vid användning av RSA rekommenderas att en nyckel måste ha en längd av 512 bitar eller mer för att den skall kunna ha en godtagbar säkerhet [Med97]. Dock kan det finnas internationella implementationer av RSA algoritmer att tillgå för allmänt bruk, även om de kanske inte är alltför lätta att få tag på.

En nackdel med att använda sig av just osymmetrisk kryptering för nyckelöverförandet är att ytterligare en krypteringsalgoritm måste implementeras, även om ett säkert system för nyckeldistribution fås, vilket naturligtvis är positivt.

5.3.2 Symmetrisk kryptering

Om inte osymmetrisk kryptering används för nyckelöverföring måste andra åtgärder tas för att kunna garantera ett säkert dokumentöverföringssystem. För att lättare kunna demonstrera hur ett sådant system ser ut, skall ett tänkt scenario, där användaren kopplar upp sig mot arkivcentralen för att överföra dokument, beskrivas. Händelseförloppet kommer att beskrivas i punktform.

- En användaren eller kund ringer upp arkivcentralen via sitt modem. Användaren ha fått tilldelat sig ett användarnamn samt ett lösenord som han eller hon själv bestämt.
- Vid uppkopplingen till centralen måste kunden logga in för att få tillgång till anslutningen. Detta görs genom att ange användar-ID och lösenord. Dessutom kontrolleras att telefonnumret stämmer överens med användarens. Uppkoppling kan således inte göras genom något annat telefonabonnemang, vilket förhindrar att någon obehörig person, som på något vis har information om både användar-ID och lösenord, loggar in från någon annan dator.
- Lösenordet som skickas över uppkopplingen för verifiering måste skyddas på något sätt för att de inte skall kunna komma i händerna på någon obehörig person. Därför krypteras lösenordet innan det skickas över till arkivcentralen. Denna nyckel för kryptering av lösenordet följer med vid leverans av systemet och matas in vid initiering av systemet. Arkivcentralen har då en kopia av nyckeln och kan således dekryptera och erhålla användarens lösenord. När användaren är inloggad i systemet kan han eller hon byta nyckel för kryptering av lösenord. Denna nyckel behöver ej arkivcentralen fortsättningsvis känna till eftersom den vid verifiering enbart behöver jämföra den krypterade nyckeln med den tidigare översända krypterade nyckeln.
- För att användaren skall kunna skicka eller hämta dokument måste användaren få tillgång till en nyckel som används till att kryptera och dekryptera dokumenten. Detta sker genom att arkivcentralen först krypterar denna nyckel med användarens lösenord med Triple-DES algoritmen. Sedan skickas denna över till användaren som dekrypterar nyckeln med sitt lösenord. Kryptering av nycklar sker således med samma kryperingsalgoritm som krypterar själva dokumenten. Detta är naturligtvis väldigt smidigt.

5.3.3 Val av lämplig teknik för nyckelöverföring

Med vetskap ovanstående information om respektive systems positiva och negativa egenskaper samt tidigare erfarenheter av vad domänen ställer för krav på systemet har valet att använda ett symmetriskt system för nyckelöverföring gjorts.

De krav som ställs på systemet för nyckelöverföringen är generellt att det skall vara säkert, pålitligt, snabbt, samt vara enkelt och billigt att implementera.

Systemet kan anses väldigt säkert såvida överföringen av användarens lösenord till arkivcentralen sker säkert första gången. Detta eftersom överföring av ett nytt lösenord endast sker en gång är det inte troligt att detta hamnar i orätta händer. En kryptering kan dessutom göras av lösenordet innan det skickas över till centralen med en nyckel som finns inbyggd i användarens applikation och således även hos arkivcentralens system. Detta lösenord kan sägas vara den enda svaga punkten hos systemet.

För att användaren skall kunna få tillgång till arkivcentralen krävs det att användaren loggar in sig i systemet vid varje uppkopplingstillfälle. Kunden verifierar sin identitet genom att ange användar-ID samt lösenord. Användar-ID och lösenord krypteras med en nyckel som genereras vid initiering av användarens system. Centralen har ej tillgång till denna nyckel utan jämför istället det krypterade lösenordet och användar-ID med tidigare krypterade lösenord samt användar-ID. Dessutom kontrolleras att telefonnumret för den kund som ringer upp stämmer överens med den sparade informationen om den aktuella användaren.

Vid varje inloggningstillfälle skapas en ny nyckel som skall användas i Triple-DES algoritmen för kryptering av de elektroniska dokumenten som skall överföras mellan användare och arkivcentralen. Denna nyckel skapas hos arkivcentralen, krypteras med användarens lösenord och skickas över till kunden.

Denna lösning för överförandet av nycklar är enkelt att implementera. Samma algoritm, dvs Triple-DES används vid all form av kryptering av data så som exempelvis nycklar och elektroniska dokument. Detta spar tid och pengar till skillnad från om RSA skulle ha använts för överförandet av nycklar, då ytterligare en krypteringsalgoritm hade behövt implementeras.

Triple-DES är, som påpekats tidigare, en säker och pålitlig algoritm. Den är dessutom helt legitim att använda över hela världen. Eftersom Triple-DES är den enda algoritm som används i hela kryptosystemet blir det således inga problem vad beträffar licenser och lagar i olika länder.

6 Slutsats

Detta kapitel kommer att sammanställa och diskutera resultatet av arbetet. Utseendet på gränssnittet mot användaren kommer att exemplifieras och diskuteras. Det kommer även att ges förslag på ytterligare arbete inom detta arbetets område.

6.1 Resultat

Ett förslag på ett delsystem, till hela dokumentöverföringssystemet, som skall hantera kryptering av dokument samt distribution av nycklar, mellan arkivcentral och användare, har tagits fram. Valet av systemen är baserade på de specificerade kraven och önskemålen på systemet för dokumentöverföringen. Alla krav och önskemål har inte kunnat tillgodoses fullt ut på grund av att ett sådant system ännu ej existerar. Exempelvis går det inte att garantera att någon metod för kryptering är helt säker. Dock har system som anses vara de mest lämpade för den aktuella miljö valts.

Som algoritm och metod för kryptering av de elektroniska dokumenten har valts att använda Triple-DES vilken är en symmetrisk krypteringsmetod, dvs den använder samma nyckel för kryptering som för dekryptering. Enligt utvärdering erbjuder Triple-DES ett system som är mycket säkert, pålitligt, snabbt, billigt, samt enkelt att implementera och distribuera.

Tidigare forskning och framtagning av nya algoritmer för kryptering har oftast resulterat i att osymmetriska system rekommenderas, varför dessa används i allt större utsträckning. Detta beror mestadels på att användningen av Internet ökar och där kan det bli problematiskt med nyckeldistributionen om ett symmetriskt system används eftersom det är många användare involverade. Dock fungerar, som påvisats, en symmetrisk krypteringsmetod bättre, i flera avseenden, för denna problemmiljön eftersom den utgör en avskärmad miljö och där varje användare endast har kontakt med arkivcentralen.

Distribution av de nycklar som ska användas vid kryptering av de elektroniska dokumenten, sker genom att kryptera dessa nycklar med Triple-DES kryptering, med användarens lösenord som krypteringsnyckel. På detta sätt blir arbetet med att implementera denna funktion väldigt enkel eftersom samma algoritm som används vid kryptering av dokument kan användas för kryptering av nycklar. Denna metod kan anses relativt säker, i synnerhet då överföringen sker över PSTN. Att använda Triple-DES, dvs DES, innebär att användandet av detta system för nyckledistribution kan ske helt legitimt. RSA exempelvis, som är en osymmetrisk krypteringsmetod är olaglig att använda för kommersiellt bruk inom USA utan licens.

Det slutliga resultatet av arbetet skiljer sig inte så markant från det förväntade resultatet som beskrevs i bakgrunden. Arbetet med att ta fram en specifik algoritm som är lämplig för kryptering av dokumenten, samt specifikation av lämpligt tillvägagångssätt för distribution av nycklar skiljer sig dock något från den förväntade arbetsgången.

Fyra olika typer av krypteringsalgoritmer utvärderades för användning vid kryptering av de elektroniska dokumenten. I bakgrunden uppskattades att ca tre algoritmer skulle komma att utvärderas. Det verkliga tillvägagångssättet skiljer sig således inte så mycket från det uppskattade. I bakgrunden angavs enbart att en mer detaljerad specifikation skulle göras för metoden för nyckeldistribution. Dock utfördes denna precisering på ett liknande sätt som för preciseringen av algoritm för kryptering av dokumenten. Två förslag på system togs fram varefter ett av dessa två system valdes.

Det förväntade resultatet har trots allt uppnåtts. De framtagna förslagen på de delsystem som hanterar dokumentkryptering och nyckeldistribution uppfyller de krav och önskemål som ställs på dokumentöverföringssystemet. Vissa av kraven tillgodoses inte fullt ut men uppfylls ändå i en fullt godtagbar grad.

6.2 Diskussion

Arbetet med att hitta lämpliga metoder för kryptering av dokument och distribution av nycklar för den specifiserade miljön har varit svårare än först uppskattat. Kryptering är ett gammalt område men fortfarande existerar ingen ultimat metod som garanterar en fullt pålitlig säkerhet. Området kryptering kan mer eller mindre liknas vid en djungel. Väldigt många algoritmer inom området finns. De flesta av dessa är mindre beprövade vilket ger en dålig pålitlighet, dvs det är svårt att veta hur säkra dessa algoritmer är. En del av de bästa och säkraste metoderna för kryptering hemlighålls av statliga myndigheter eller innehar förbud mot export till andra länder. De flesta av de övriga algoritmer som anses vara av en mer säker karaktär kräver en licens för att kunna användas i kommersiellt bruk. En privatperson kan dock oftast använda sig av de flesta kryptosystem gratis. En privatperson tycker oftast inte att det är värt att betala för ett sådant system. Samtidigt är det bra för tillverkarna om systemet testas av så många personer som möjligt för att på så vis testa algoritmens pålitlighet.

Statliga myndigheter ser en chans att i framtiden kunna ha tillgång till all krypterad information genom att ha en typ av huvudnyckel som kan låsa upp all krypterad information. Avsikten är att skydda staten och dess människor men samtidigt utgör detta en obehagligt stor makt för staten som då kan läsa all hemlig och personlig information som sänds över exempelvis Internet.

Området kryptering kommer troligtvis utvecklas radikalt inom en relativt snar framtid. I synnerhet då allt mer information, av mer eller mindre känslig och personligt slag, skickas över Internet, vilket är en väldigt öppen arkitektur. Kryptering kommer att krävas för att kunna behålla någon form av personlig integritet.

Problemet i detta arbete innefattar dock inte Internet. De elektroniska dokumenten skall skickas över telenätet vilket utgör en betydligt mer avskärmad miljö. De hemliga dokumenten blir således inte lika exponerade för obehöriga personer. Trots detta behövs någon form av kryptering för att förhindra ett eventuellt försök från någon obehörig person att läsa innehållet i dokumenten. När val av lämpligt kryptosystem skall göras får en avvägning göras mellan vad de dokument som skickas över telenätet är värda, samt vad det kostar att uppnå en viss grad av säkerhet. Ett bra alternativ är att använda en lite äldre och mer beprövad metod som på så vis kan erbjuda en acceptabel säkerhet. Dessa metoder är oftast enkla att får tag i samtidigt som de oftast kan användas utan någon kostnad. DES är ett exempel på en sådan algoritm. För att öka säkerheten men bibehålla samma höga pålitlighet kan Triple-DES användas som då är en variant av DES.

Flera alternativa lösningar finns på det specificerade problemet. Det val av lösning som valts är baserat på att systemet skall vara billigt och enkelt att implementera, samtidigt som systemet skall vara säkert. Därför valdes Triple-DES samt den preciserade metoden för nyckeldistribution som lösning på problemet. Frågan är om det kan finnas andra lösningar till problemet som är bättre. Det är inte helt osannolikt att det finns en krypteringsalgoritm som är väldigt säker samtidigt som den är snabb och fri för kommersiellt bruk utan att någon licens krävs, och som således lämpar sig bättre för problemet. Problemet är då att denna algoritm troligtvis är dåligt beprövad och distribuerad, dvs det den har en dålig pålitlighet. Dålig pålitlighet kan innebära att

en algoritm kan innehålla brister som gör den osäker att använda. Detta är ytterligare en orsak till varför DES är en bra algoritm att använda. DES är en väl beprövad och pålitlig algoritm.

En bättre lösning på problemet av nyckeldistribution kan troligtvis finnas i framtiden. Det kanske smidigaste sättet ur säkerhetssynpunkt är att använda osymmetrisk kryptering för att överföra nycklar mellan arkivcentral och användare. Dock är det ont om bra, säkra och beprövade sådana algoritmer, som dessutom är legitima att använda globalt för kommersiellt bruk. I framtiden ser förhoppningsvis lagar och förordningar annorlunda ut. Önskvärt vore om metoder för kryptering kan studeras globalt till skillnad från idag då mycket av den forskning som finns inom området sker inom ett lands gränser med mycket sekretess. Om ett globalt samarbete fanns skulle det finnas betydligt mer resurser för att ta fram nya, bra och säkra kryptosystem som även blir standardiserade.

Fungerar systemet lika bra för stora respektive små kunder? Systemet är framtaget för att i så stor utsträckning som möjligt kunna tillgodose alla typer av kunder. Därför är de krav som har ställts på systemet relativt generella, exempelvis att systemet skall vara snabbt och säkert. Om en kund är väldigt stor, dvs omsätter mycket dokument av liknande karaktär är det kanske lämpligt att konstruera någon form av automatiseringsprocess som hanterar överföring av dessa dokument till och från arkivcentralen. Denna typen av kunder utgör dock troligtvis extremfall och inte majoriteten av de kunder som kan tänkas komma att använda ett system av denna typ.

Skulle detta framtagna system, som är en lösning på just detta problem, kunna tillämpas på andra problem? Möjligtvis skulle denna lösning kunna fungera på andra miljöer som även de har en stjärnformad struktur. Dock är lösningen baserad på att kommunikationen sker via telenätet. Skulle kommunikation ske över exempelvis Internet är det troligtvis mer lämpligt med ett annat system för nyckelöverföring med tanke på säkerheten. Dessutom har hänsyn tagits till att access till arkivcentralen sker relativt sällan, i det avseende att det exempelvis då inte är av högsta vikt att överföringsprocessen är väldigt snabb.

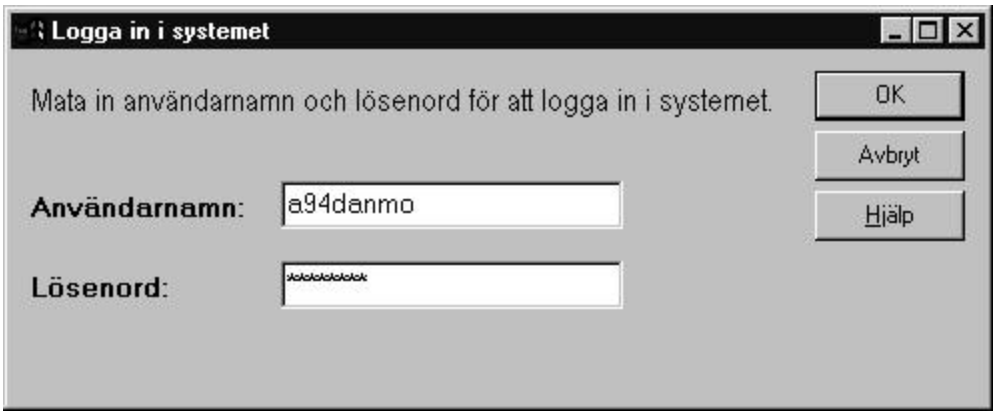
6.3 Gränssnitt mot användaren

Hur skulle ett gränssnitt mot användaren kunna se ut? I detta kapitel diskuteras och ges förslag på hur ett sådant gränssnitt skulle kunna se. Dess utseende är baserat på det valda systemet för kryptering och nyckelöverföring, samt de grundläggande krav som ställs för överföringssystemet.

Ett av kraven som ställs på systemet är att det skall vara enkelt att använda, samt lätt att förstå för användaren. Användaren skall behöva göra minsta möjliga arbete, i form av både fysiskt arbete men också arbete som kräver mycket tankeverksamhet, vid överföring av ett dokument, till eller från arkivcentralen. Exempelvis skall användaren inte behöva handskas med nycklar, som i krypteringsalgoritmen Triple-DES är relativt långa. Systemet som användaren kommer att hantera kommer således inte alls likna prototypen som krypterar dokumenten. Användarens gränssnitt kommer till skillnad från prototypen att vara grafiskt utformad.

Riktlinjer för hur ett användargränssnitt skall utformas har hämtats från [Löw93].

För att kunden skall kunna få tillgång till arkivcentralen för att skicka och hämta dokument, måste användaren först identifiera sig och logga in i systemet. Nedan ges ett exempel på hur detta gränssnittet mot användaren skulle kunna se ut vid detta förfarande.



Figur 15: Användarens gränssnitt för inloggning i systemet

Ovanstående ruta är enkelt utformad och lätt att förstå [Löw93]. Användaren matar in sin användaridentitet och sitt personliga lösenord som användaren själv har valt. För att logga in trycker sedan användaren på knappen märkt ”OK” för att skicka över uppgifterna till arkivcentralen och därmed begära inträde i systemet. Om användaren mot all förmodan inte skulle förstå hur han eller hon skall gå till väga, finns även en knapp märkt ”Hjälp” där utförligare information finns att hämta. När användaren har loggat in skickas en nyckel från arkivcentralen till kunden. Denna nyckel kommer att användas för att kryptera de dokument som skall överföras.

Hur gör då användaren för att genomföra själva överföringen av de elektroniska dokumenten? Återigen är det viktigt att denna process är så enkel och lättförståelig som möjligt för användaren. Nedan visas ett exempel på hur en dialogruta, för överföring av dokument mellan arkivcentral och användare, skulle kunna se ut.

Figur 16: Användarens gränssnitt för överföring av dokument

Utseendet på dialogrutan för denna tjänst liknar den föregående dialogrutan för inloggning i systemet i dess utformning och funktionalitet. Gränssnittet mot systemet skall vara konsekvent och enhetligt utformat, både utseendemässigt och funktionellt, så att det skall vara lättare att förstå och lära sig hantera

dokumentöverföringsprocessen. Även här finns det en knapp märkt ”Hjälp” som användaren kan använda om han eller hon behöver mer hjälp.

Processen för att föra över ett eller flera dokument till och från arkivcentralen blir mycket enkel för användaren, dvs kunden. Endast klickningar med musen utför hela överföringen av dokumenten. Inga nycklar behöver hanteras direkt av användare utan sköts av applikationen. De befintliga dokumenten i arkivcentralen och i användarens dator visas i respektive delruta när dialogrutan först visas. Dessa dokument kan sedan flyttas fram och tillbaka mellan de två delrutorna, som symboliserar användarens dator samt arkivcentralen. För att bekräfta och verkställa överföringen enligt användarens specifikation skall knappen märkt ”OK” tryckas ned. Kryptering av de elektroniska dokumenten görs med den tidigare översända nyckeln innan dokumentet skickas över.

Användaren behöver inte ens veta att kryptering av dokumenten sker vid överföringen eftersom han eller hon inte behöver ta del av den proceduren. Kunden behöver bara veta att själva överföringen sker på ett tryggt och säkert sätt. Det intressanta ur kundens synvinkel är att själva dokumentet flyttas över till eller från arkivcentralen. Han eller hon vet att de dokument som lagras i arkivcentralen ligger i tryggt förvar, vilket är en av avsikterna med att flytta över dokumenten från användarens dator till arkivcentralen. Hur dokumenten sedan förflyttas mellan användaren och det centrala arkivet spelar således inte någon större roll för användaren.

6.4 Fortsatt arbete

Detta arbete tar enbart fram en lösning till en del av det större ursprungliga problemet. Hela problemet består i att finna en lämplig metod för överföring och arkivering av elektroniska dokument för den specificerade miljön. Således återstår mycket arbetet för att få fram en komplett lösning på hela problemet. I detta delkapitel kommer funderingar och förslag på fortsatt arbete med att finna lösningar till problemet att framföras.

Hur skall exempelvis de elektroniska dokumenten lagras i arkivcentralen? Är det nödvändigt ur säkerhetssynpunkt att kryptera dessa dokument? Någon obehörig person kanske, mot all förmodan, får tillträde till arkivcentralen på något sätt. Då kan det vara bra om informationen som finns lagrad där inte finns tillgänglig i klartext. Vilken metod för kryptering är då lämplig för detta ändamål? Eftersom detta är en helt annan miljö, till skillnad från de dokument som skall skickas över telenätet, är det helt andra faktorer som spelar in vid val av lämplig krypteringsmetod.

Vilken metod för nyckelhantering skall i så fall användas om kryptering anses nödvändig? I detta fall är det inte fråga om någon distribution av nycklar, eftersom dokumenten ej ska flyttas till någon annan geografisk plats. Skall samtliga dokument i arkivcentralen vara krypterade med samma nyckel, eller skall det finnas en nyckel för varje dokument, eller alternativt en nyckel för varje användare? Hur bör dessa nycklar förvaras och vem har tillgång till dem? Skall de förvaras i arkivcentralen eller hos användaren?

När användaren skall logga in i systemet måste han eller hon ange användarnamn och lösenord. Hur skall dessa uppgifter lagras i arkivcentralen med tanke på säkerheten.

Vilka säkerhetsåtgärder skall tas mot mjuk- eller hårdvarufel som kan tänkas inträffa i den enhet i arkivcentralen som lagrar dokumenten, som kan orsaka att dokumenten förstörs? Går det att försäkra sig om att inget inträffar som kan skada de elektroniska dokumenten?

Slutsats

Vid hantering av vanliga hårdvarudokument, dvs information tryckt på vanligt papper, gäller för vissa typer av dokument vissa lagar och bestämmelser beträffande arkivering. Exempelvis bestämmelser kring kvaliteten på det papper som dokumentet är skrivet på, samt regler kring utseendet på den lokal där dokumenten skall förvaras. Det finns naturligtvis olika lagar för olika länder. Arkivering av elektroniska dokument är ett relativt nytt begrepp. Finns det speciella regler för denna typen av dokument vid arkivering? Gäller samma regler som för vanliga dokument? Troligtvis är detta ett problem som kommer att belysas mer längre fram i tiden då det uppskattningsvis blir mer vanligt med denna typen av arkivering.

Referenser

Böcker

[Hal92] Halsall, Fred (1992), *Data Communications, Computer Networks and Open Systems*, Third Edition

[Rik85] Riksdataförbundet, SIS-SGT, Sårbarhetsföreningen (1985), *Kryptering i ADB-system, Praktisk hjälpreda för beslutsfattare och systemvetare*, SIS teknisk rapport 312, Utgåva 1

[Bec88] Beckett, Brian (1988), *Introduction to Cryptology*

[Gar95] Garfinkel, Simson (1995), *PGP: Pretty Good Privacy*

[Bra88] Brassard, Gilles (1988), *Lecture Notes in Computer Science, Modern Cryptology*

[Löw93] Löwgren, Jonas (1993), *Human-computer interaction*

Internetsidor

[McK98] McKay, Farrell (april 1998), *Fortify for Netscape*, https://www.fortify.net/

[Jär97] Järlstig, Andreas, Boström, Fredrik, Sjölén, Mattias, (dec 1997), *Säkerhet - En krypterad framtid*, http://www.ntostud.mh.se/~ping9528/rapport.html

[Rds97] RSA Data Security (1997), *DES – RSA Challange Cracked*, http://www.rsa.com/des/

[Rds98] RSA Data Security (1998), *RSA's Secret-Key Challenge Solved by Distributed Team in Record Time*, http://www.rsa.com/pressbox/html/980226.html

[Hus97] Husman, Hans (nov 1997), *Chiffer*, http://www.csd.uu.se/~tv98hah/kryptering/chiffer.html

[Rit98] Ritter, Terry (apr 1998), *Ciphers By Ritter*, http://www.io.com/~ritter/CRYPHTML.HTM

[Rit94] Ritter, Terry (apr 1994), Fenced DES, http://www.io.com/~ritter/NEWS/94042901.HTM

[Med97] Medenis, Michael M. (maj 1997), Security in Teleradiology Systems: Requirements and Proposed Mechanisms, http://www.ece.arizona.edu/~medenis/hw2/sem_pro.htm

[Sch96] Schneier, Bruce (1996), *IDEA*, http://www.r3.ch/products/idea/

Förkortningar

DES	Data Encryption Standard
Fenced DES	Variant av DES
IDEA	International Data Encryption Key
PGP	Pretty Good Privacy
PSTN	Public Switched Telephone Network
RSA	Rivest, Shamir och Adlemans (Osymmetrisk krypteringsmetod)
Triple-DES	Variant av DES där algoritmen används tre gånger
VFAB	Västgöta Företagsarkiv AB
XOR	Exclusive-or

Bilaga A: Prototyp (main.c)

```
/* encryption/decryption algorithm using Triple-DES vith 168 byte key
*/
#include <stdio.h>

char iv[8]; /* Initial vector for CBC mode */
int block;
char filename[50]; /* Name of file to be handled */

main(argc,argv)
int argc;
char *argv[];
{
      int c,cnt,encrypt,decrypt,hexflag,session;
      register int i;
      char
key[8],tkey1[30],tkey[20],tkey2[30],*akey,*akeybig,*getpass();
      extern char *optarg;

      hexflag = block = encrypt = decrypt = 0;
      akey = akeybig = NULL;

      while((c = getopt(argc,argv,"hedk:b")) != EOF){
            switch(c){
            case 'h':
                  /* hexflag++; */
                  break;
            case 'e':
                  encrypt++;
                  break;
            case 'd':
                  decrypt++;
                  break;
            case 'k':
                  /* akeybig = optarg; */
                  break;
            case 'b':
                  /* block++; */
                  break;
            }
      }
      if(encrypt == 0 && decrypt == 0){
            fprintf(stderr,"Usage: des -e|-d\n");
            exit(2);
      }

      /* Get name of file to be handled */
      printf("Enter name of file: ");
      scanf("%s",filename);

      if(akeybig == NULL){
            /* No key on command line, prompt for it */
            memset(tkey1,0,sizeof(tkey1));
            memset(tkey2,0,sizeof(tkey2));
            for(;;){
                  akeybig = getpass("Enter key: ");
                  strncpy(tkey1,akeybig,sizeof(tkey1));
                  akeybig = getpass("Enter key again: ");
                  strncpy(tkey2,akeybig,sizeof(tkey2));
```

```
            if(strncmp(tkey1,tkey2,sizeof(tkey1)) != 0){
                  fprintf(stderr,"Key mistyped, try again\n");
            }else
                  break;
      }
      akeybig = tkey1;
      printf("Large key: %s\n",tkey1);
}

if(encrypt){
      for(session=0;session<3;session++){
            for(i=0;i<8;i++)
                  tkey[i]=tkey1[i+(session*8)];
            akey = tkey;
            printf("Session: %i\n",session+1);
            printf("Session key: %s\n",akey);

            strncpy(key,akey,8);
            /* Set up key, determine parity bit */
            for(cnt = 0; cnt < 8; cnt++){
                  c = 0;
                  for(i=0;i<7;i++)
                        if(key[cnt] & (1 << i))
                              c++;
                  if((c & 1) == 0)
                        key[cnt] |= 0x80;
                  else
                        key[cnt] &= ~0x80;
                  }

            /* Blot out original key */
            i = strlen(akey);
            i = (i < 8) ? i : 8;
            memset(akey,0,i);

            desinit(0);
            setkey(key);

            /* Initialize IV to all zeros */
            memset(iv,0,8);

            doencrypt();
      }

} else{
      for(session=2;session>-1;session--){
            for(i=0;i<8;i++)
                  tkey[i]=tkey1[i+(session*8)];
            akey = tkey;
            printf("Session: %i\n",session+1);
            printf("Session key: %s\n",akey);

            strncpy(key,akey,8);
            /* Set up key, determine parity bit */
            for(cnt = 0; cnt < 8; cnt++){
                  c = 0;
                  for(i=0;i<7;i++)
                        if(key[cnt] & (1 << i))
                              c++;
                  if((c & 1) == 0)
                        key[cnt] |= 0x80;
                  else
```

```
                        key[cnt] &= ~0x80;
                    }

                /* Blot out original key */
                i = strlen(akey);
                i = (i < 8) ? i : 8;
                memset(akey,0,i);

                desinit(0);
                setkey(key);

                /* Initialize IV to all zeros */
                memset(iv,0,8);

                dodecrypt();
            }
        }
}

doencrypt()
{
        char work[8],*cp,*cp1,ch,expres[58];
        int cnt,i;
        FILE *fil;
        FILE *fil2;

        fil = fopen("temp","w");
        fclose(fil);
        fil2 = fopen(filename,"r");
        for(;;){
            if((cnt = fread(work,1,8,fil2)) != 8){
                /* Put residual byte count in the last block.
                 * Note that garbage is left in the other bytes,
                 * if any; this is a feature, not a bug, since
it'll
                 * be stripped out at decrypt time.
                 */
                work[7] = cnt;
            }
            if(!block){
                /* CBC mode; chain in last cipher word */
                cp = work;
                cp1 = iv;
                for(i=8; i!=0; i--)
                    *cp++ ^= *cp1++;
            }
            endes(work);      /* Encrypt block */
            if(!block){ /* Save outgoing ciphertext for chain */
                memcpy(iv,work,8);
            }
            /* Add decrypted data to temp file */
            fil=fopen("temp","a+");
            fwrite(work,1,8,fil);
            fclose(fil);
            /* If at end of file */
            if(cnt != 8){
                fclose(fil); fclose(fil2);
                /* Copy encrypted data into origin file */
                sprintf(expres,"mv temp %s",filename);
                system(expres);
                break;
            }
```

```
	}
}

dodecrypt()
{
	char work[8],nwork[8],ivtmp[8],*cp,*cp1,ch,expres[58];
	int cnt,i;
	  FILE *fil;
	  FILE *fil2;

	  fil = fopen("temp","w");
	  fclose(fil);
	fil2 = fopen(filename,"r");
	cnt = fread(work,1,8,fil2);   /* Prime the pump */
	for(;;){
		if(!block){ /* Save incoming ciphertext for chain */
			memcpy(ivtmp,work,8);
		}
		dedes(work);
		if(!block){ /* Unchain block, save ciphertext for next */
			cp = work;
			cp1 = iv;
			for(i=8; i!=0; i--){
				*cp++ ^= *cp1++;
			}
			memcpy(iv,ivtmp,8);
		}
		/* Save buffer pending next read */
		memcpy(nwork,work,8);
		/* Try to read next block */
		cnt = fread(work,1,8,fil2);
		/* If at end of file */
		if(cnt != 8){     /* Can "only" be 0 if not 8 */
			/* Prev block was last one, write appropriate
number
			 * of bytes
			 */
			cnt = nwork[7];
			if(cnt < 0 || cnt > 7){
				fprintf(stderr,"Corrupted file or wrong
key\n");
			} else if(cnt != 0){
				/* Add decrypted data to temp file */
				  fil=fopen("temp","a+");
				fwrite(nwork,1,cnt,fil);
			}
			fclose(fil); fclose(fil2);
			/* Copy encrypted data into origin file */
			sprintf(expres,"mv temp %s",filename);
			system(expres);
			break;
		} else {
			/* Add decrypted data to temp file */
			  fil=fopen("temp","a+");
			fwrite(nwork,1,8,fil);
				fclose(fil);
		}
	}
}
/* Convert hex/ascii nybble to binary */
int
htoa(c)
```

```
char c;
{
      if(c >= '0' && c <= '9')
            return c - '0';
      if(c >= 'a' && c <= 'f')
            return 10 + c - 'a';
      if(c >= 'A' && c <= 'F')
            return 10 + c - 'A';
      return -1;
}
/* Convert bytes from hex/ascii to binary */
gethex(result,cp,cnt)
register char *result;
register char *cp;
register int cnt;
{
      while(cnt-- != 0){
            *result = htoa(*cp++) << 4;
            *result++ |= htoa(*cp++);
      }
}
#ifdef      DEBUG
put8(cp)
register char *cp;
{
      int i;

      for(i=0;i<8;i++){
            fprintf(stderr,"%02x ",*cp++ & 0xff);
      }
}
#endif
```

Bilaga B: Prototyp (des.c)

```
#define     NULL  0

#ifdef      LITTLE_ENDIAN
unsigned long byteswap();
#endif

/* Tables defined in the Data Encryption Standard documents */

/* initial permutation IP */
static char ip[] = {
      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
};

/* final permutation IP^-1 */
static char fp[] = {
      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
};

/* expansion operation matrix
 * This is for reference only; it is unused in the code
 * as the f() function performs it implicitly for speed
 */
#ifdef notdef
static char ei[] = {
      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
};
#endif

/* permuted choice table (key) */
static char pc1[] = {
      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
};

/* number left rotations of pc1 */
static char totrot[] = {
      1,2,4,6,8,10,12,14,15,17,19,21,23,25,27,28
};

/* permuted choice key (table) */
static char pc2[] = {
      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
};

/* The (in)famous S-boxes */
static char si[8][64] = {
      /* S1 */
      14,  4, 13,  1,  2, 15, 11,  8,  3, 10,  6, 12,  5,  9,  0,  7,
       0, 15,  7,  4, 14,  2, 13,  1, 10,  6, 12, 11,  9,  5,  3,  8,
       4,  1, 14,  8, 13,  6,  2, 11, 15, 12,  9,  7,  3, 10,  5,  0,
      15, 12,  8,  2,  4,  9,  1,  7,  5, 11,  3, 14, 10,  0,  6, 13,

      /* S2 */
      15,  1,  8, 14,  6, 11,  3,  4,  9,  7,  2, 13, 12,  0,  5, 10,
       3, 13,  4,  7, 15,  2,  8, 14, 12,  0,  1, 10,  6,  9, 11,  5,
       0, 14,  7, 11, 10,  4, 13,  1,  5,  8, 12,  6,  9,  3,  2, 15,
      13,  8, 10,  1,  3, 15,  4,  2, 11,  6,  7, 12,  0,  5, 14,  9,

      /* S3 */
      10,  0,  9, 14,  6,  3, 15,  5,  1, 13, 12,  7, 11,  4,  2,  8,
      13,  7,  0,  9,  3,  4,  6, 10,  2,  8,  5, 14, 12, 11, 15,  1,
      13,  6,  4,  9,  8, 15,  3,  0, 11,  1,  2, 12,  5, 10, 14,  7,
       1, 10, 13,  0,  6,  9,  8,  7,  4, 15, 14,  3, 11,  5,  2, 12,

      /* S4 */
       7, 13, 14,  3,  0,  6,  9, 10,  1,  2,  8,  5, 11, 12,  4, 15,
      13,  8, 11,  5,  6, 15,  0,  3,  4,  7,  2, 12,  1, 10, 14,  9,
      10,  6,  9,  0, 12, 11,  7, 13, 15,  1,  3, 14,  5,  2,  8,  4,
       3, 15,  0,  6, 10,  1, 13,  8,  9,  4,  5, 11, 12,  7,  2, 14,

      /* S5 */
       2, 12,  4,  1,  7, 10, 11,  6,  8,  5,  3, 15, 13,  0, 14,  9,
      14, 11,  2, 12,  4,  7, 13,  1,  5,  0, 15, 10,  3,  9,  8,  6,
       4,  2,  1, 11, 10, 13,  7,  8, 15,  9, 12,  5,  6,  3,  0, 14,
      11,  8, 12,  7,  1, 14,  2, 13,  6, 15,  0,  9, 10,  4,  5,  3,

      /* S6 */
      12,  1, 10, 15,  9,  2,  6,  8,  0, 13,  3,  4, 14,  7,  5, 11,
      10, 15,  4,  2,  7, 12,  9,  5,  6,  1, 13, 14,  0, 11,  3,  8,
       9, 14, 15,  5,  2,  8, 12,  3,  7,  0,  4, 10,  1, 13, 11,  6,
       4,  3,  2, 12,  9,  5, 15, 10, 11, 14,  1,  7,  6,  0,  8, 13,

      /* S7 */
       4, 11,  2, 14, 15,  0,  8, 13,  3, 12,  9,  7,  5, 10,  6,  1,
      13,  0, 11,  7,  4,  9,  1, 10, 14,  3,  5, 12,  2, 15,  8,  6,
```

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

      /* S8 */
      13,  2,  8,  4,  6, 15, 11,  1, 10,  9,  3, 14,  5,  0, 12,  7,
       1, 15, 13,  8, 10,  3,  7,  4, 12,  5,  6, 11,  0, 14,  9,  2,
       7, 11,  4,  1,  9, 12, 14,  2,  0,  6, 10, 13, 15,  3,  5,  8,
       2,  1, 14,  7,  4, 10,  8, 13, 15, 12,  9,  0,  3,  5,  6, 11
};

/* 32-bit permutation function P used on the output of the S-boxes */
static char p32i[] = {
      16,  7, 20, 21,
      29, 12, 28, 17,
       1, 15, 23, 26,
       5, 18, 31, 10,
       2,  8, 24, 14,
      32, 27,  3,  9,
      19, 13, 30,  6,
      22, 11,  4, 25
};
/* End of DES-defined tables */

/* Lookup tables initialized once only at startup by desinit() */
static long (*sp_)[64];       /* Combined S and P boxes */

static char (*iperm)[16][8];  /* Initial and final permutations */
static char (*fperm)[16][8];

/* 8 6-bit subkeys for each of 16 rounds, initialized by setkey() */
static unsigned char (*kn)[8];

/* bit 0 is left-most in byte */
static int bytebit[] = {
      0200,0100,040,020,010,04,02,01
};

static int nibblebit[] = {
       010,04,02,01
};
static int desmode;

/* Allocate space and initialize DES lookup arrays
 * mode == 0: standard Data Encryption Algorithm
 * mode == 1: DEA without initial and final permutations for speed
 * mode == 2: DEA without permutations and with 128-byte key
(completely
 *            independent subkeys for each round)
 */
desinit(mode)
int mode;
{
      char *malloc();

      if(sp_ != NULL){
            /* Already initialized */
            return 0;
      }
      desmode = mode;

      if((sp_ = (long (*)[64])malloc(sizeof(long) * 8 * 64)) ==
NULL){
```

```
            return -1;
      }
      spinit();
      kn = (unsigned char (*)[8])malloc(sizeof(char) * 8 * 16);
      if(kn == NULL){
            free((char *)sp_);
            return -1;
      }
      if(mode == 1 || mode == 2)    /* No permutations */
            return 0;

      iperm = (char (*)[16][8])malloc(sizeof(char) * 16 * 16 * 8);
      if(iperm == NULL){
            free((char *)sp_);
            free((char *)kn);
            return -1;
      }
      perminit(iperm,ip);

      fperm = (char (*)[16][8])malloc(sizeof(char) * 16 * 16 * 8);
      if(fperm == NULL){
            free((char *)sp_);
            free((char *)kn);
            free((char *)iperm);
            return -1;
      }
      perminit(fperm,fp);

      return 0;
}
/* Free up storage used by DES */
desdone()
{
      if(sp_ == NULL)
            return;     /* Already done */

      free((char *)sp_);
      free((char *)kn);
      if(iperm != NULL)
            free((char *)iperm);
      if(fperm != NULL)
            free((char *)fperm);

      sp_ = NULL;
      iperm = NULL;
      fperm = NULL;
      kn = NULL;
}
/* Set key (initialize key schedule array) */
setkey(key)
char *key;              /* 64 bits (will use only 56) */
{
      char pc1m[56];          /* place to modify pc1 into */
      char pcr[56];           /* place to rotate pc1 into */
      register int i,j,l;
      int m;

      /* In mode 2, the 128 bytes of subkey are set directly from the
       * user's key, allowing him to use completely independent
       * subkeys for each round. Note that the user MUST specify a
       * full 128 bytes.
       *
```

```
       * I would like to think that this technique gives the NSA a
real
       * headache, but I'm not THAT naive.
       */
      if(desmode == 2){
            for(i=0;i<16;i++)
                  for(j=0;j<8;j++)
                        kn[i][j] = *key++;
            return;
      }
      /* Clear key schedule */
      for (i=0; i<16; i++)
            for (j=0; j<8; j++)
                  kn[i][j]=0;

      for (j=0; j<56; j++) {        /* convert pc1 to bits of key */
            l=pc1[j]-1;       /* integer bit location  */
            m = l & 07;       /* find bit        */
            pc1m[j]=(key[l>>3] &    /* find which key byte l is in */
                  bytebit[m]) /* and which bit of that byte */
                  ? 1 : 0;    /* and store 1-bit result */
      }
      for (i=0; i<16; i++) {        /* key chunk for each iteration
*/
            for (j=0; j<56; j++)    /* rotate pc1 the right amount */
                  pcr[j] = pc1m[(l=j+totrot[i])<(j<28? 28 : 56) ? l:
l-28];
                  /* rotate left and right halves independently */
            for (j=0; j<48; j++){   /* select bits individually */
                  /* check bit that goes to kn[j] */
                  if (pcr[pc2[j]-1]){
                        /* mask it in if it's there */
                        l= j % 6;
                        kn[i][j/6] |= bytebit[l] >> 2;
                  }
            }
      }
}
/* In-place encryption of 64-bit block */
endes(block)
char *block;
{
      register int i;
      unsigned long work[2];        /* Working data storage */
      long tmp;

      permute(block,iperm,(char *)work);  /* Initial Permutation */
#ifdef LITTLE_ENDIAN
      work[0] = byteswap(work[0]);
      work[1] = byteswap(work[1]);
#endif

      /* Do the 16 rounds */
      for (i=0; i<16; i++)
            round(i,work);

      /* Left/right half swap */
      tmp = work[0];
      work[0] = work[1];
      work[1] = tmp;

#ifdef LITTLE_ENDIAN
```

```
      work[0] = byteswap(work[0]);
      work[1] = byteswap(work[1]);
#endif
      permute((char *)work,fperm,block);  /* Inverse initial
permutation */
}
/* In-place decryption of 64-bit block */
dedes(block)
char *block;
{
      register int i;
      unsigned long work[2];  /* Working data storage */
      long tmp;

      permute(block,iperm,(char *)work);  /* Initial permutation */

#ifdef LITTLE_ENDIAN
      work[0] = byteswap(work[0]);
      work[1] = byteswap(work[1]);
#endif

      /* Left/right half swap */
      tmp = work[0];
      work[0] = work[1];
      work[1] = tmp;

      /* Do the 16 rounds in reverse order */
      for (i=15; i >= 0; i--)
            round(i,work);

#ifdef LITTLE_ENDIAN
      work[0] = byteswap(work[0]);
      work[1] = byteswap(work[1]);
#endif

      permute((char *)work,fperm,block);  /* Inverse initial
permutation */
}

/* Permute inblock with perm */
static
permute(inblock,perm,outblock)
char *inblock, *outblock;           /* result into outblock,64 bits
*/
char perm[16][16][8];               /* 2K bytes defining perm. */
{
      register int i,j;
      register char *ib, *ob;       /* ptr to input or output block
*/
      register char *p, *q;

      if(perm == NULL){
            /* No permutation, just copy */
            for(i=8; i!=0; i--)
                  *outblock++ = *inblock++;
            return;
      }
      /* Clear output block    */
      for (i=8, ob = outblock; i != 0; i--)
            *ob++ = 0;

      ib = inblock;
```

```
	for (j = 0; j < 16; j += 2, ib++) { /* for each input nibble */
		ob = outblock;
		p = perm[j][(*ib >> 4) & 017];
		q = perm[j + 1][*ib & 017];
		for (i = 8; i != 0; i--){   /* and each output byte */
			*ob++ |= *p++ | *q++;   /* OR the masks together*/
		}
	}
}

/* Do one DES cipher round */
static
round(num,block)
int num;                        /* i.e. the num-th one   */
unsigned long *block;
{
	long f();

	/* The rounds are numbered from 0 to 15. On even rounds
	 * the right half is fed to f() and the result exclusive-ORs
	 * the left half; on odd rounds the reverse is done.
	 */
	if(num & 1){
		block[1] ^= f(block[0],kn[num]);
	} else {
		block[0] ^= f(block[1],kn[num]);
	}
}
/* The nonlinear function f(r,k), the heart of DES */
static
long
f(r,subkey)
unsigned long r;        /* 32 bits */
unsigned char subkey[8];      /* 48-bit key for this round */
{
	register unsigned long rval,rt;
#ifdef	TRACE
	unsigned char *cp;
	int i;

	printf("f(%08lx, %02x %02x %02x %02x %02x %02x %02x %02x) = ",
		r,
		subkey[0], subkey[1], subkey[2],
		subkey[3], subkey[4], subkey[5],
		subkey[6], subkey[7]);
#endif
	/* Run E(R) ^ K through the combined S & P boxes
	 * This code takes advantage of a convenient regularity in
	 * E, namely that each group of 6 bits in E(R) feeding
	 * a single S-box is a contiguous segment of R.
	 */
	rt = (r >> 1) | ((r & 1) ? 0x80000000 : 0);
	rval = 0;
	rval |= sp_[0][((rt >> 26) ^ *subkey++) & 0x3f];
	rval |= sp_[1][((rt >> 22) ^ *subkey++) & 0x3f];
	rval |= sp_[2][((rt >> 18) ^ *subkey++) & 0x3f];
	rval |= sp_[3][((rt >> 14) ^ *subkey++) & 0x3f];
	rval |= sp_[4][((rt >> 10) ^ *subkey++) & 0x3f];
	rval |= sp_[5][((rt >> 6) ^ *subkey++) & 0x3f];
	rval |= sp_[6][((rt >> 2) ^ *subkey++) & 0x3f];
	rt = (r << 1) | ((r & 0x80000000) ? 1 : 0);
	rval |= sp_[7][(rt ^ *subkey) & 0x3f];
```

```
#ifdef      TRACE
      printf(" %08lx\n",rval);
#endif
      return rval;
}
/* initialize a perm array */
static
perminit(perm,p)
char perm[16][16][8];               /* 64-bit, either init or final
*/
char p[64];
{
      register int l, j, k;
      int i,m;

      /* Clear the permutation array */
      for (i=0; i<16; i++)
            for (j=0; j<16; j++)
                  for (k=0; k<8; k++)
                        perm[i][j][k]=0;

      for (i=0; i<16; i++)          /* each input nibble position */
            for (j = 0; j < 16; j++)/* each possible input nibble */
            for (k = 0; k < 64; k++)/* each output bit position */
            {   l = p[k] - 1; /* where does this bit come from*/
                  if ((l >> 2) != i)  /* does it come from input
posn?*/
                  continue;   /* if not, bit k is 0    */
                  if (!(j & nibblebit[l & 3]))
                  continue;   /* any such bit in input? */
                  m = k & 07; /* which bit is this in the byte*/
                  perm[i][j][k>>3] |= bytebit[m];
            }
}

/* Initialize the lookup table for the combined S and P boxes */
static int
spinit()
{
      char pbox[32];
      int p,i,s,j,rowcol;
      long val;

      /* Compute pbox, the inverse of p32i.
       * This is easier to work with
       */
      for(p=0;p<32;p++){
            for(i=0;i<32;i++){
                  if(p32i[i]-1 == p){
                        pbox[p] = i;
                        break;
                  }
            }
      }
      for(s = 0; s < 8; s++){             /* For each S-box */
            for(i=0; i<64; i++){          /* For each possible input
*/
                  val = 0;
                  /* The row number is formed from the first and last
                   * bits; the column number is from the middle 4
                   */
```

```
                  rowcol = (i & 32) | ((i & 1) ? 16 : 0) | ((i >> 1)
& 0xf);
                  for(j=0;j<4;j++){ /* For each output bit */
                        if(si[s][rowcol] & (8 >> j)){
                         val |= 1L << (31 - pbox[4*s + j]);
                        }
                  }
                  sp_[s][i] = val;

#ifdef      DEBUG
                  printf("sp_[%d][%2d] = %08lx\n",s,i,sp_[s][i]);
#endif
            }
      }
}
#ifdef      LITTLE_ENDIAN
/* Byte swap a long */
static
unsigned long
byteswap(x)
unsigned long x;
{
      register char *cp,tmp;

      cp = (char *)&x;
      tmp = cp[3];
      cp[3] = cp[0];
      cp[0] = tmp;

      tmp = cp[2];
      cp[2] = cp[1];
      cp[1] = tmp;

      return x;
}
#endif
```

Bilaga C: Prototyp (misc.c)

```
#ifndef OSK
″
/* Set block of memory to constant */
″
memset(blk,val,size)
″
register char *blk;
″
register char val;
register unsigned size;
{
      while(size-- != 0)
            *blk++ = val;
}

/* Copy block of memory */
memcpy(dest,src,size)
register char *dest,*src;
register unsigned size;
{
      while(size-- != 0)
            *dest++ = *src++;
}

/* Compare two blocks of memory */
memcmp(a,b,size)
register char *a,*b;
register unsigned size;
{
      while(size-- != 0)
            if(*a++ != *b++)
                  return 1;
      return 0;
}
#endif
```

Bilaga D: Prototyp (getpass.c)

```
#include <stdio.h>
#include <signal.h>
#include <sgtty.h>

#define     TTY   "/dev/tty"  /* Change to "con" for MS-DOS */

#define     SIG   void            /* Change to run on sun */

/* Issue prompt and read reply with echo turned off */
char *
getpass(prompt)
char *prompt;
{
      struct sgttyb ttyb,ttysav;
      register char *cp;
      int c;
      FILE *tty;
      static char pbuf[128];
      SIG (*signal())(), (*sig)();    /* change for sun */
/*    int (*signal())(), (*sig)(); */ /* orginal */

      if ((tty = fdopen(open(TTY, 2), "r")) == NULL)
            tty = stdin;
      else
            setbuf(tty, (char *)NULL);
      sig = signal(SIGINT, SIG_IGN);
      ioctl(fileno(tty), TIOCGETP, &ttyb);
      ioctl(fileno(tty), TIOCGETP, &ttysav);
      ttyb.sg_flags |= RAW;
      ttyb.sg_flags &= ~ECHO;
      ioctl(fileno(tty), TIOCSETP, &ttyb);
      fprintf(stderr, "%s", prompt);
      fflush(stderr);
      cp = pbuf;
      for (;;) {
            c = getc(tty);
            if(c == '\r' || c == '\n' || c == EOF)
                  break;
            if (cp < &pbuf[127])
                  *cp++ = c;
      }
      *cp = '\0';
      fprintf(stderr,"\r\n");
      fflush(stderr);
      ioctl(fileno(tty), TIOCSETP, &ttysav);
      signal(SIGINT, sig);
      if (tty != stdin)
            fclose(tty);
      return(pbuf);
}
```