Produktivitet - en motpol till säkerhet.

(HS-IDA-EA-00-610)

Maria Dalhage (a95marda@student.his.se)

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

Examensarbete i informationssystemutveckling, 10 p

vårterminen 2000.

Handledare: Anders Malmsjö

Tack till

Peter Bengtsson som hjälpte mig med min lastmätning samt besvarade tekniska frågor.

Intygssida

Denna rapport, Produktivitet - en motpol till säkerhet, är skriven av Maria Dalhage 2000-06-26.

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.

_____

Sammanfattning

Huvudfrågan i arbetet är hur produktivitet och säkerhet kombineras i de säkerhetslösningar som existerar idag samt huruvida högre säkerhet leder till ökad komplexitet. Det tillvägagångssätt som använts består av en lastmätning av Secure Socket Layer (SSL), och Pretty Good Privacy (PGP), samt en teoretisk jämförelse av säkerhetslösningarna PGP, SSL och Secure Electronic Transaction (SET). Detta innebär att PGP som har jämförelsevis hög säkerhet när det kommer till kryptering även är den säkerhetslösning är mest resurskrävande. SSL däremot är mindre säkert med betydligt lägre resursanvändning. Generellt kan man med facit i hand säga att den lösning som är säkrast även är den mest komplexa. Den säkerhetsaspekt lastmätningen bygger på är främst krypteringsalgoritm och nyckellängd. Även om dessa aspekter är allmänt vedertagna i säkerhetstänkande måste även hänsyn tas till säkerhetstänkandet som helhet. SET visar ett sätt att kringgå den klassiska motsättningen mellan säkerhet och produktivitet genom att begränsa användningsområdet, men detta på bekostnad av flexibilitet och ger vidare potential för interoperabilitet.

Nyckelord: Säkerhet, Produktivitet, PGP, SSL, SET, Kryptering, Linux, Windows 2000, Lastmätning

Innehållsförteckning

1 Inledning

Detta arbete kommer att behandla frågeställningar rörande produktivitet, effektivitet och säkerhet. Produktivitet och säkerhet är två begrepp som kan hamna i motsatsförhållande och min avsikt är här att belysa några aspekter av denna motsättning.

Det en användare uppfattar som effektivitet har hittills stått i motsats till allt vad säkerhetstänkande beträffar (Thylander, 2000). Det kan handla om möjligheten att surfa fritt på Internet, använda ICQ (I seek you), kolla sin externa postbox på hotmail eller dylik. Dessa är exempel på aktiviteter många användare önskar kunna utföra när de befinner sig på sin arbetsplats. Att en användare ser dessa som produktiva tjänster gör dem icke desto mindre till möjliga säkerhetsluckor till arbetsplatsen. Även andra tjänster som mer direkt kan ses som arbetsrelaterade kan innebära säkerhetsrisker. Exempel på sådana tjänster kan vara att kunna läsa sin e-post via Internet om man är ute och reser, att kunna komma åt sina hemliga arbetsdokument hemifrån eller från Internet, i stort sett all form av uppringt hemarbete, då sk Random Access System (RAS)-servrar[1] sällan uppfyller de säkerhetsmål många säkerhetschefer sätter upp (Thylander, 2000). Liknande tjänster uppskattas av många användare och de anser ofta att de höjer produktivitetsgraden på arbetsplatsen (Thylander, 2000). Det är denna motsättning man på allt mer innovativa sätt försöker komma runt men ändå verkar användarens mål ständigt stå i konflikt med säkerheten på företaget.

Strävan efter elektroniska säkerhetslösningar är stor och säkerhetskraven stärks då mer och mer känslig information lagras och transporteras elektroniskt (Lindberg och Ohlsson, 2000).

Säkerhet handlar om att skydda sig mot de hot och risker som finns. Ett säkert system är ett system som eliminerar dessa hot och risker. Det finns olika verktyg som begränsar riskerna. Det är dock viktigt att klargöra att inget verktyg klarar av alla säkerhetsproblem och det krävs att den totala säkerhetslösningen innehåller alla förutsättningar som behövs för att säkra systemet.

Idag består nästan varje arbetsplats av lokala nätverk med en koppling ut mot Internet. I ett lokalt nätverk finns många hål att täppa igen så att ingen obehörig trafik gör intrång. Detta innebär att de lokala nätverken bl a måste skydda sig med brandväggar och krypterad trafik. Detta har inverkan på prestandan då brandväggar kontrollerar all trafik in och ut från nätverket. Kryptering kräver datorkraft och gör att datorernas processorer får massor med extraarbete och därigenom minskad produktivitet. Införandet av säkerhetssystem innebär också mer arbete och nya rutiner såväl för administratörer som ledning och användare. För att inte göra nätverket oanvändbart, men säkert, måste vi hitta en jämn balans mellan säkerhet och produktivitet. Detta görs lämpligen genom att definiera en hotbild, vad ska vi skydda oss mot och hur. Utifrån detta kan sedan en lämplig säkerhetslösning utarbetas. Exempelvis kan det förutsättas att Riksbanken har betydligt större säkerhetskrav än en privat revisionsbyrå, eftersom Riksbankens tillgångar vida överstiger små företags och därför utgör ett begärligare mål. Båda eftersträvar dock maximal prestanda men samtidigt maximal säkerhet för att skydda sig mot tänkbara hot. Det rör sig hela tiden

[1] RAS server programvara finns med i NT server paketet. RAS servern ger möjlighet för servern att fungera som en modempool för nätverk.

om en avvägning mellan hur säkert ett system ska vara och hur produktivt det ska vara att använda. En viktig faktor är givetvis prislappen på systemet – ibland så kan man bli tvingad att välja en mindre säker lösning eftersom att det skulle bli för dyrt att kunna hantera den graden av säkerhet som man egentligen ville ha.

Rapporten börjar med att definiera vad jag menar med produktivitet och säkerhet, vilka säkerhetsrisker som finns samt hur de kan avhjälpas. Med detta som bas så ämnar jag sedan gå vidare till olika typer av kryptering som belyser en viktig aspekt hos flera säkerhetslösningar. Utifrån det studeras sedan några olika säkerhetssystem i olika miljöer och analyseras utifrån ovanstående givna kriterier.

2 Bakgrund

I motsatsförhållandet mellan produktivitet och säkerhet måste dessa två aspekters olika definitioner klargöras. I båda fallen handlar det om två abstrakta begrepp som används både inom och utanför användningen av datorer. Detta avsnitt ska belysa vad jag anser menas med effektivitet samt produktivitet både gällande användarens syn och utifrån en mer teknisk infallsvinkel. Även en djupare analys av säkerheten kommer också att göras.

2.1 Effektivitet

Samuelsson (1978) menar att effektivitet är ett mått på hur väl ett mål uppfylls eller hur bra resultat man får med givna förutsättningar. Andra begrepp som hjälper till att definiera effektivitet är förenkling (Lund et al., 1992), automatisering (Eriksson, 1998) samt standardisering (Preece et al., 1994)

Ett system som avhjälper onödig komplexitet är en del av ett effektivt system. Det system som stödjer det procedurella minnet, den kunskapsnivå som finns i det undermedvetna, kräver inte att användaren måste begrunda sina handlingar utan att de sker ”av sig självt” (Lund et al., 1992). Exempel på detta är den erfarna datoranvändaren som låter fingrarna spela över tangentbordet utan att denna verkligen ”tänker” på var de olika tangenterna är placerade. Genom av avlasta minnet i den utsträckning det är möjligt så kan användaren handla på rutin vilket går snabbare därför ses som mer effektivt. Detta kan ses som en form av förenkling om det tidigare systemet var komplext.

Ett exempel på automatisering är EDI, Electronic Data Interchange. Detta är en form av elektronisk handel där köpares och säljares datasystem kan kommunicera med varandra automatiskt, utan mänsklig inblandning (Ericsson, 1998) Att automatisera kan dessutom ses som ett sätt att öka säkerheten, då en av de största säkerhetsriskerna – människan – fjärmas från systemet.

Åkerman (2000), säger att genom att standardisera system och program underlättas hantering på så sätt att rutiner och uppgifter ter sig på likartade sätt. Han säger även att standarden, i detta fall gällande säkerhetslösningar och program, leder till interoperabilitet.

2.2 Produktivitet

Samuelsson (1978) nämner begreppet deleffektivitet, även kallat produktivitet. Denna typ av produktivitet är alltså sambandet mellan kostnad och storleken hos resultatet. Enligt Samuelsson (1978) är strävan efter ett förbättrat system till minskade resurser ett led i produktiviteten. Han särskiljer på effektivitet och produktivitet. Inom systemteorin menar Samuelsson på att effektivitet belyser hur väl systemets mål uppfylls medan produktivitet visar på hur väl systemet utnyttjar sina resurser. Samuelssons definition av deleffektivitet, dvs produktivitet är det som belyses i detta arbete.

2.2.1 Kvantitativa produktivitetsaspekter

Det är svårt att bedöma vad varje enskild användare lägger i ordet produktivitet. När jag nu diskuterar produktivitet kommer jag att bedöma detta utifrån kvantitativa produktivitetsaspekter såsom minne, processtid, diskutrymme och bandbredd. Dessa resurser är viktiga för prestanda och om ett program använder för mycket resurser så

tar det längre tid för programmet att utföra sin uppgift om programmet klarar av det överhuvudtaget. Produktiviteten mäts i detta fall genom att betrakta hur lite resurser ett program klarar sig med samt hur väl de utnyttjas. Dessa aspekter tas i beaktande då de även kan ses som en del i användarproduktivitet, detta eftersom att dröjsmål irriterar användarna.

2.2.1.1 Primärminne

Med minne avses här enligt Silberschatz och Galvin (1994) primärminne, det minne i vilket data och kod måste finnas vid användning. Primärminne, också kallat RAM[2]-minne brukar oftast ställas i kontrast till sekundärminne - med vilket man brukar åsyfta hårddiskar och andra liknande lagringsmedier. Mängden primärminne som ett program använder kan vara relevant för hur lång tid det tar för programmet att utföra en given uppgift. Anledningen till att mängden minne som används är kritisk för tidsåtgången är att om primärminnet tar slut blir operativsystemet tvunget att lagra delar av primärminnet på sekundärminnet. Detta är tidskrävande eftersom exekvering av data sker i primärminnet och inte i sekundärminnet. Om datamängden då är lagrat i sekundärminnet måste utrymme frigöras i primärminnet så att den efterfrågade mängden data ännu en gång hamnar i primärminnet (Silberschatz och Galvin et al, 1994). Tiden det tar att komma åt någonting i primärminnet mäts vanligen i ns (nanosekunder) medan tiden det tar att hitta någonting i sekundärminnet mäts i ms (millisekunder). Skillnaden i söktid, den tid det tar att lokalisera ett givet dataelement, är alltså ungefär en faktor 10^6 (Brorsson, 1999). Kontentan av ovanstående är att så fort det använda minnet överstiger mängden primärminne så riskerar man att få allvarliga prestandaförluster.

2.2.1.2 Sekundärminne

Det lagringsutrymme, som ett program använder, kan delas upp i den mängd själva programmet tar upp, och den lagringsmängd som programmet använder när det jobbar. Detta kan t ex vara temporära filer, en mellanlagringsbuffer (cache) eller loggfiler. Lagringsutrymme är oftast en kritisk resurs såtillvida att om lagringsutrymmet tar slut så slutar många program att fungera vilket läsaren säkerligen själv fått uppleva. Slut på primärminne eller processkapacitet leder till att saker går långsammare - slut på sekundärminne leder oftast till att de slutar att fungera (Silberschatz och Galvin, 1994).

2.2.1.3 Processtid

Processtid syftar på två olika aspekter av ett programs tidsåtgång enligt Silberschatz och Galvin (1994). Dels hur lång tid det tar totalt att utföra en uppgift, dels hur stor del av den tillgängliga processkapaciteten som behöver tas i anspråk för att genomföra någonting. De hävdar även att ett program kan behöva mindre kapacitet än vad som finns tillgängligt beroende på att ett normalt program ägnar större delen av tiden åt att vänta på att olika data skall finnas tillgängliga. Detta kan t ex gälla program som reagerar på användaren och som då kan ägna mycket tid åt att vänta på musklick, tangenttryckningar osv (Silberschatz och Galvin, 1994).Vid mätning av använd processtid, mäter man t ex hur stor del av var sekund som används aktivt till databearbetning. Dessvärre är det så att processtidsprofilen för ett program kan variera ganska kraftigt under dess körning (Bengtsson, 2000). Istället identifieras olika relevanta förlopp där programmets åtgång i processtid varierar (Beizer, 1990).

[2] Random Access Memory

2.2.1.4 Bandbredd

Termen bandbredd används för att benämna den totala överföringskapaciteten i ett datornätverk vilken vanligtvis mäts i bitar per sekund (Halsall, 1993). Ett program som kommunicerar över ett nätverk har en tillgänglig bandbredd som beror på vad det är för trafik i övrigt och hur stor volym den har. Redan vid ganska låg belastning, ungefär 40% kan man börja märka av transmissionsfördröjningar (Halsall, 1993). Detta beror på vilka typer av nätverk som finns på vägen, men den bakomliggande orsaken är att data skickas i paket som kan komma när som helst och om två olika paket råkar hamna på samma del av ett nätverk samtidigt får man fördröjningar [3] (Halsall, 1993).

För många program gäller att bandbredden på nätverket oftast är en begränsande faktor då den tillgängliga bandbredden inte räcker till för att hålla transmissionen sysselsatt hela tiden (Silberschatz och Galvin, 1994). Nu är det så att bandbreddsbehov liksom processtidsbehov varierar med tiden, men lösningen är densamma - man studerar behovet under olika intressanta förlopp (Silberschatz och Galvin, 1994).

2.2.1.5 Operativsystem

Ett operativsystem är en kärna[4] som tillsammans med en mängd program och kod interagerar med omvärlden - användare, kontrollerar operativsystem/hårdvara och tillhandahåller ytterligare tjänster för andra program (Silberschatz och Galvin, 1994).

Det är pga operativsystemets interaktion med program, som det är av intresse att närmare förklara vad som är ett operativsystem och vad som skiljer de två operativsystemen åt som används i detta arbete. Denna jämförelse mellan Debian GNU/Linux och Windows presenteras i Bilaga 1.

Gränsen för vad som tillhör och inte tillhör ett operativsystem är dock inte alltid särskilt skarp. Vad kärnan består av kan skilja sig mellan operativsystem (Silberschatz och Galvin, 1994).

2.3 Säkerhet

Att känna sig trygg och säker är viktigt för oss. Vi har vissa säkerhetsrutiner som vi ser som sunt förnuft. Vi låser våra ytterdörrar när vi går hemifrån så att ingen ska ta sig in i vårt hem och stjäla våra ägodelar. Vi låser bildörrar för att göra det krångligare för någon annan att stjäla den, signerar brev och avtal med vår personliga signatur så att den som läser detta brev eller handhar vårt avtal ska veta att det är just vi som skrivit under. Vi klistrar igen våra kuvert för att avskräcka den alltför nyfikne att ta del av information som inte är behörig för denne. Vi skaffar oss en försäkring så att vi får ersättning för skador som uppkommer på oss själva eller våra ägodelar. När det kommer till fysiska ägodelar, såsom byggnader, kapital mm eller är en fråga om personlig integritet har vi nått långt i vårt säkerhetstänkande. Men när det gäller olika typer av information har vårt säkerhetstänkande inte nått fullt så långt.

Som alltid när det kommer till nya tekniker är det svårt att veta vilka faror som finns och vilka säkerhetsaspekter som ska införas. När vi åker bil använder vi bilbälte av

[3] Fördröjningarna beror alltså inte direkt på hur stor del av kapaciteten som utnyttjas utan på hur sannolikt det är att olika paket skall mötas

[4] Kärnan är den första och primära processen i ett operativsystem. Kärnans huvuduppgift är att betjäna andra program.

säkerhetsskäl, men när vi däremot nyttjar elektroniska kommunikationsmedel så är de flesta relativt omedvetna om de risker som existerar. Många sätter alltför stor tilltro till tekniken och använder den utan eftertanke.

Freese och Holmberg (1993) kategoriserar olika hot och faror som ofta förekommer inom tele- och datasäkerhet.

Informationsstöld

- Hacking
- Linjeavlyssning
- Buggning
- Röjande signaler, RÖS
- Terminalavläsning
- Feladressering

Manipulering av information

- Hacking
- Övertagande av linje
- Återutsändning
- Falsk information

Förlust av information

- Hacking
- Stöld av utrustning
- Oönskade dataprogram, tex datavirus
- Brand
- Sabotage

Avbrott

- Kabelavgrävning
- Åska, vatten och brand
- Sabotage

Nedan kommer en utförligare beskrivning från Freese och Holmberg (1993) där en del av ovanstående begrepp förklaras närmare:

Genom att använda en linjeavlyssnare, en s k protokollanalysator som är ett mätinstrument för feldetektering och underhåll på ledningar inom datakommunikation går det att lyssna av datalinjer. Linjeavlyssnaren kopplas in på en ledning och all information som förs över ledningen kopieras. Det går även att använda en tranciever

som skruvas fast på ledningen för att kunna lyssna av information Kryptering förhindrar inte att man kopierar det som skickas, men väl att man kan förstå den skickade informationen.

RÖS, röjande signaler, är den elektromagnetiska strålning som en dator sänder ut och vilken med lämplig utrustning kan fångas upp. Denna utrustning är både avancerad och dyr, men t ex militären eller säkerhetstjänster kan ha tillgång till den. Det går att skydda sig mot denna typ av attacker, men det är ganska dyrt – att röjningsskydda en enskild dator kostar tiotusentals kronor. Röjningsskyddet går i princip ut på att bygga in alla delar av datorn i burar av metallnät som fångar upp och sprider signalerna.

Röjningsskydd förekommer så gott som endast där extremt hög säkerhet krävs – detta kan vara militära anläggningar, vissa banksystem eller extra känsliga maskiner hos företag.

En av de större farorna är dock inte extern avlyssning, utan avlyssning av s.k. trojaner och virus. En trojan är ett program som utger sig för att göra en sak men vars egentliga syfte är något annat, det egentliga syftet kan exempelvis vara att läsa av tangentbordet när en användare skriver in sina lösenord och sedan skicka dem till den som gjorde trojanen.

Virus är däremot replikerande program som försöker att gömma sig för användaren, till skillnad ifrån trojaner som inte gömmer sig alls. Virus kan ha olika syften, att förstöra data, att bara sprida sig eller att ta reda på känslig information. Virus har dock ofta bieffekter som gör dem mer benägna att förstöra data än trojaner. Ett virus infiltrerar operativsystemet för att gömma sig och om viruset samtidigt ändrar operativsystemets beteende för övriga program så löper man en mycket större risk att data förstörs eller försvinner.

Efter ovanstående listning av Freese och Holmberg (1993), vill jag dessutom inflika att datasäkerhet handlar inte enbart om att skydda informationen mot användares misstag och obehörigas nyfikenhet, det är lika mycket en säkerhet för användaren som på detta sätt gör sig fri från misstankar när fel inträffar.

Då läsaren får alla hot listade framför sig kan det uppfattas som en självklarhet att försöka att skydda sig ifrån dessa. Ändå har jag fått den uppfattningen när jag kommit i kontakt med olika företag att säkerhetstänkandet inte är speciellt tillfredsställande. Ett problem kan vara att användarna upplever säkerhetsreglerna som komplexa och som ett hinder i deras arbete. Detta skapar en negativ attityd till att säkra sig. Det är därför viktigt att välja informationsskyddet på ett sådant sätt att användarna märker så litet som möjligt av det i deras dagliga arbete, men blir medvetna om varför säkerheten krävs och stödjer dessa rutiner. Freese och Holmberg (1993) hävdar att utbildning och motivering är viktiga aspekter kring säkerhet. Feloperationer, slarv och okunskap resulterar i att säkerhetssystemen inte fungerar. Användare med olika kunskapsnivåer har skilda förutsättningar att använda de säkerhetssystem som finns på ett produktivt sätt. Det är viktigt att se till att kunskapsnivån hos användarna motsvarar vad som behövs för att uppnå den säkerhet man eftersträva. Detta är en fråga där ansvaret till stor del kan läggas på företagsledningar och chefer som genom att driva dessa frågor kan höja säkerheten (Freese och Holmberg 1993).

2.3.1 Policy

Datasäkerhet idag fokuserar mycket på tekniska lösningar men innan man får ett fungerade säkerhetssystem bör man se till att man har en genomtänkt säkerhetspolicy (Lindberg och Ohlsson, 2000). Svensk Standard SS 62 77 99-1 (SIS, 1999) är ett

framtaget regelverk som kan användas som underlag och stöd för utveckling av en organisations säkerhetsföreskrifter. Denna policy tar upp följande:

- Informationssäkerhet
- Riskanalys
- Riskhantering
- Säkerhetspolicy
- Säkerhetsorganisation
- Klassificering och kontroll av tillgångar
- Personal säkerhet
- Fysisk och miljörelaterad säkerhet
- Styrning av kommunikation och drift
- Styrning av åtkomst
- Avbrottsplanering
- Efterlevnad/uppföljnings rutiner

Genom en diskussion med representanter från olika företag såsom WM-data, SJ-data, och ISG-systems AB, har jag fått uppfattningen att problemet idag med hantering av säkerhetspolicy är att den inte efterlevs. Företag spenderar stort kapital för att utfärda ett policydokument. Dock saknas energi till att följa upp om det som togs fram efterlevs överhuvudtaget. Läser man igenom ovanstående punkter framgår det tydligt att säkerhet minst lika mycket är en policy- och ledningsfråga som en fråga om teknik. Det är viktigt att se säkerheten i ett system som en kontinuerlig process och inte som ett fast tillstånd. Arbetet med att analysera och upprätthålla eller förbättra säkerheten måste alltid pågå. Schneier (1995) har sagt:

> ”Security is a process, not a product”

2.3.2 Kryptering och dekryptering

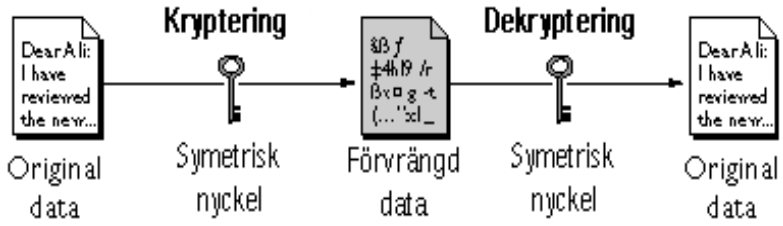
Detta avsnitt baseras på Johansson (1998). Tekniska säkerhetslösningar bygger till större delen av olika krypteringslösningar. Krypteringstekniken klarar av att säkra en del systemområden, men långt ifrån alla områden. De viktigaste tillämpningar av krypteringstekniken är :

- *Konfidentialitet.* Detta betyder att viss information inte ska kunna läsas av obehöriga.
- *Dataintegritet*. Om information ändras på ett otillbörligt sätt så går det att upptäcka. En mottagare kan kontrollera om ett meddelande manipulerats sedan det lämnade avsändaren.
- *Autenticiering,* eller *verifiering* av identitet handlar om att kunna fastställa identiteten hos en användare eller apparat. Detta för att parterna ska vara säkra på vem de kommunicerar med.
- *Icke-förnekelse* eller *non-repudiation.* En utförd elektronisk transaktion ska inte kunna förnekas av någon av de inblandade parterna. Detta gör det t ex möjligt för företag att binda en kund vid ett köp som denne gjort.

En kryptering är enkelt uttryckt en algoritm (metod, tillvägagångssätt) för att omvandla ett meddelande från en form till en annan. Detta på ett sådant sätt att man måste ha tillgång till en bit information till, nyckeln, för att omvandla tillbaka meddelandet till ursprungsformen. Om nyckeln är kort, så kan den ofta gissas – eller så kan man prova alla möjliga nycklar. Detta är anledningen till att man vill använda långa nycklar då detta försvårar att någon ’knäcker’ koden. Långa nycklar leder dock till att det kräver mer arbete att kryptera/dekryptera eftersom att hela nyckeln måste användas, annars så har man ju egentligen en kortare nyckel.

Inom kryptering idag är det RSA som lägger standarden för vilka krypteringsalgoritmer som används. Med deras algoritmer måste en asynkron kryptering ha en nyckellängd av minst 512-bitar medan en synkron kryptering endast behöver ha en nyckellängd på 64-bitar. Asymmetrisk kryptering är således mer resurskrävande för systemet. Man talar i dag om symmetrisk- och asymmetrisk kryptering.

2.3.2.1 Symmetrisk kryptering



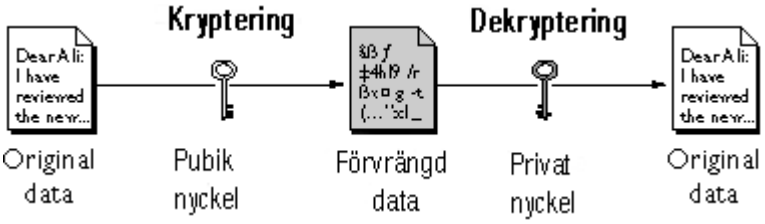
Figur 1. Bilden visar symmetrisk kryptering (Netscape, 2000)

Detta avsnitt är baserat på (Johansson, 1998). Med symmetrisk kryptering använder man sig av en nyckel för både kryptering och dekryptering. Denna teknik kan utföras med hög prestanda, den kräver inte särskillt mycket resurser. Detta gör att användaren inte upplever denna säkerhetshantering som ett hinder (Netscape, 2000). Vad som ytterligare är positivt med denna teknik är att användarna får en indirekt autenticiering av var den krypterade datan kommer ifrån eftersom det endast kan komma från någon som har tillgång till krypteringsnyckeln. Detta bör endast vara ett fåtal. Denna strategi innebär att användarna måste lita på varandra så att de inte delar ut krypteringsnyckeln till osäker källa eller förlorar den utan att veta om det, eller att någon har kopierat den så att en tredje part kan använda den. Detta är inte bara en säkerhetsrisk, det är även opraktiskt då hanteringen inte är flexibel. Det är en omständig process med nyckelbyte.

För att vara säker krävs strikta rutiner för hanteringen av symmetriska nycklar. Den idag mest använda symmetriska krypteringen heter DES, vilket står för Data Encryption Standard. DES knäcktes för första gången 1997 och man har därför börjat att utveckla nya symmetriska krypteringsstandarder som även i framtiden kan uppfylla de krav man har på säkerhet och produktivitet. Någonting som har satt ytterligare press på nya krypteringar är att det idag finns projekt som t ex Distributed.net som ägnar sig åt att knäcka koder. distributed.net prövar idag 132 miljarder nycklar per sekund (Distributed.net, 2000).

2.3.2.2 Asymmetrisk Kryptering

Detta avsnitt baserar sig på Johansson (1998). Med asymmetrisk kryptering använder man nyckelpar bestående av en privat och en publik nyckel. Den privata nyckeln används oftast som dekrypteringsnyckel och skall bevaras strikt privat. Den publika nyckeln används oftast för kryptering och den kan spridas ut till andra. Genom att bara en person behöver ha den privata, hemliga nyckeln förenklas administrationen av krypteringsnycklar avsevärt. Den privata hålls hemlig hos användaren och den publika görs tillgänglig för andra genom någon central administration.



Figur 2. Bilden visar asymmetrisk kryptering (Netscape, 2000)

Det som skiljer asymmetrisk kryptering från symmetrisk kryptering åt är nyckellängderna. Jämfört med symmetrisk kryptering kräver asymmetrisk mer beräkningar – längre krypteringsalgoritmer. Den privata nyckeln består av par med två stycken primtal, den publika nyckeln är produkten av dessa primtal. I asymmetrisk kryptering måste således nyckellängderna vara långa för att man inte skall kunna gissa vilka primtal som används. Detta innebär minskad produktivitet i hanteringen. Det krävs mer datakraft för beräkningsprocessen. Detta gör att asymmetrisk kryptering inte alltid är tillämpbart för större mängder data. Det är dock möjligt att använda publik kryptering för att skicka en symmetrisk nyckel, vilken i sin tur kan användas till att kryptera data. Det går även att hantera dessa nycklar spegelvänt, dvs att endast den privata nyckeln används som krypteringsnyckel och den publika som dekrypteringsnyckel. Eftersom den privata nyckeln är strikt personlig kan den användas som en digital signatur. Denna lösning har använts för att på ett effektivt sätt få in autenticiering och kryptering för elektronisk handel och annan kommersiell handel som använder kryptering.

2.3.3 Digital Signatur

Detta avsnitt baserar sig på Johansson (1998). För att kunna hantera informationen över t ex Internet och vara säker på dess ursprung räcker inte kryptering till. Det krävs även en autenticiering på att avsändaren är den person denne utger sig för att vara. För att hantera detta problem används digitala signaturer. Som nämndes tidigare kan en privat nyckel användas som digital signatur

Istället för att kryptera själva datamängden skapar den signerande mjukvaran en envägs hash[5] av den och använder sedan den privata nyckeln för att kryptera hashvärdet (Netscape, 2000). Anledningen till att använda sig av ett hashvärde är att slippa kryptera hela datamängden med den privata nyckeln. Eftersom att den publika nyckeln sedan kan användas för att dekryptera information som krypterats med den

[5] Hashing är en metod som bygger på en hashalgoritm vilken skapar ett värde som beror på en hela datamängden i meddelandet (Elmasri et al, 1994).

privata, så har man nu signerat. Den krypterade hashen tillsammans med information om vilken hashalgoritm som använts utgör en digital signatur Ett annat sätt att identifiera sig elektroniskt är att använda certificat

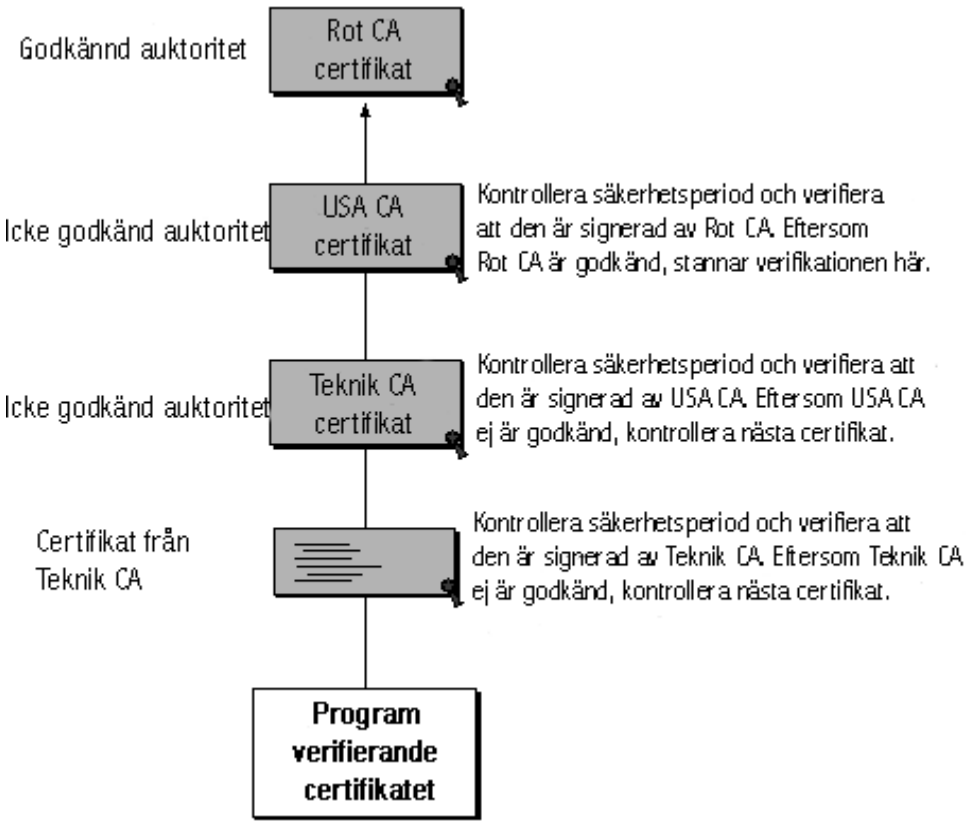
2.3.4 Elektroniska certificat

Ett certifikat är ett elektroniskt dokument som används för att identifiera individer, servrar, företag eller andra entiteter som har behov av elektronisk identifiering. Det kan jämföras med ett pass eller ett körkort och utgör en internationell erkänd identitetshandling. Varje certifikat utfärdas av en auktoriserad utgivare, en certifikat autenticierare (CA). Denna CA garanterar certifikatens giltighet och kan verifiera att de användare som vill använda certifikat utfärdade från denna CA är godtagbara (Johansson, 1998).

Följande certifikat är vanligast enligt Netscape (2000):

CA certifikat. Används för att identifiera en CA. Man kallar dessa för rotcertifikat då detta certifikat utgör stommen i ett certifikat utfärdat av denna CA.

Exempel: Ett företag kan ha ett certifikat som säkerställer företagets identitet. Detta certifikat utgör då roten i detta företags CA. Då sedan de anställda får certifikat utfärdade av CA:n så säkerställs dels deras identitet genom personliga uppgifter och dels genom att de är giltiga användare som är utfärdade av en giltig CA. Identifiering ligger således både i personliga uppgifter och i företagets rotcertifikat.



Figur 3. Bilden visar verifierat certifikat genom CA-kedja (Netscape, 2000)

Klient SSL certifikat. Används för att identifiera klienter för servrar. Vanligast är att Certifikatet används för att säkerställa en persons identitet, t ex en anställd, i ett företag eller en bankkund. Certifikatet är utfärdat av ett företags CA.

Exempel: En bank ger en kund ett klient SSL certifikat vilket gör att bankens servrar kan verifiera denne kund som en användare genom att identifiera denne och kan således ge kunden tillgång till sina konton över t ex Internet. Dessutom kan ett företag ge en anställd ett klient SSL certifikat för att denne skall komma åt sina egna skyddade användarkonton och ge den anställde tillgång till de servrar denne behöver i sitt arbete.

Server SSL certifikat. Används för att identifiera servrar för klienter. Server autenticiering kan användas utan klient autenticiering. Certifikatet är utfärdat av ett företags CA.

Exempel: Internetsiter som hanterar elektronisk commerce stödjer oftast serverbaserad autenticiering. Det innebär att den person som vill handla över Internet med kreditkort gör detta genom en session mellan klientcertifikat och servercertifikat. Detta gör att kunden är säker på mottagarens identitet och att känslig information som kortnummer hålls hemlig för övrig internettrafik.

S/MIME certifikat. Används för att signera och kryptera e-post. Precis som med klient SSL certifikat fastställer ett sådant certifikat en persons identitet. Certifikatet är utfärdat av ett företags CA.

Exempel: Inom ett stort företag kan det finnas behov av att säkerställa identiteten hos e-postanvändare då känslig information skickas. Detta för att säkerställa att sändaren verkligen är den denne utger sig för att vara.

2.3.4.1 Hantering av certifikat

Då stommen i en säkerhetslösning oftast grundar sig på digitala certifikat måste det finnas en garanti för att dessa certifikat kan anses vara giltiga. Som visats ovan är ett företags anställdas certifikat utfärdade av företaget själv. Men för att dessa certifikat kan anses giltiga måste företagets rotcertifikat vara giltigt. Detta certifikat ansöks om hos ett fåtal företag som är erkända rotcertifikatutfärdare. För att få ett rotcertifikat måste registreringsbevis för företaget erläggas så att företagets existens kan verifieras (Netscape, 2000).

2.3.4.2 LDAP katalogen

För att kunna hantera certifikatanvändning på ett flexibelt sätt krävs att dessa kan erhållas från en central plats. För att lösa detta använder man sig av en global katalogtjänst. Denna tjänst kallas LDAP (Lightweight Directory Access Protocol) och ger en stor flexibilitet inom certifikathantering. Tjänsten består av en global standard över organistationsträdsstrukturer, vilket kan jämföras med vanliga katalogträd i ett filsystem. Genom att använda applikationer på en server kan man skapa ett katalogträd som t ex kan motsvara ett organisationsträd. För att använda detta för certifikat lägger man upp certifikaten i denna trädstruktur så att de som behöver komma åt certifikaten kan hämta dem genom att söka sig genom företagets organisationsträd. Detta görs genom anrop till den server som innehåller katalogtjänsten (Netscape, 2000).

2.3.5 Public Key Infrastructure (PKI)

Det vanligaste begreppet inom dagens krypteringslösningar är PKI, Public Key Infrastructure. Med PKI menas den utrustning och de rutiner som behövs för att hantera publika krypterings- och signaturnycklar samt digitala certifikat (Netscape, 2000).

PKI-lösningar gör följande (Johansson, 1998):

- Den utfärdar digitala certifikat.
- Den gör certifikat tillgängliga för användaren.
- Den återkallar certifikat.
- Den förnyar certifikat vars giltighetstid gått ut.
- I vissa fall lagrar den kopior av nycklar, för den händelse att de går förlorade hos användaren.

PKI baseras på asymmetrisk kryptering och ses som det generella verktyget att använda för autenticiering, godkännande (authorization), kryptering och nyckelhantering i olika Internet och intranet applikationer inklusive säker meddelandehantering och e-handel. Allt fler företag använder sig av digitala certifikat med motivationen att minska kostnaderna och öka produktiviteten (Johansson, 1998). Detta genom att företagets interna processer förbättras. Pappersformulär som kräver underskrift av den anställde ersätts med digitala signaturer. Dessutom, med den ökade säkerheten som PKI med digitala certifikat erbjuder, kan företagen erbjuda tjänster som annars skulle vara omöjliga av säkerhetsproblem, t ex elektronisk handel (Netscape, 2000).

Ett stort problem är att vi i dagens läge har en uppsjö av PKI- produkter som inte är kompatibla med varandra, de kan med andra ord inte läsa varandras certifikat eller spärrlistor. Detta ställer till problem för användarna. I ett system där det inte råder interoperabilitet uppstår avbrott som saktar ner systemet och hindrar tillgängligheten till det. Detta leder till tidsförluster och att tilltron till systemet minskar. Produktiviteten samt effektiviteten uteblir helt enkelt (Computer Sweden, 2000).

De säkerhetslösningar jag betraktar i arbetet är PGP, SSL och SET. Dessa säkerhetslösningar angriper säkerhetsproblemet på olika sätt och ger en god överblick över de tekniker som till stor del används idag.

2.3.5.1 Secure Socket Layer (SSL)

Följande information i detta avsnitt är hämtat från Johansson (1998). SSL-protokollet används för att autenticiera, det vill säga verifiera identiteten hos en klient och en server och etablera en krypterad förbindelse över Internet mellan dem. SSL utnyttjas av andra protokoll, exempelvis HTTP och FTP, för att skydda information som sänds mellan en klient och en server. Protokollet bygger på användning av både symmetrisk och asymmetrisk kryptering.

Det skrevs av Netscape och utnyttjar teknik utvecklad av RSA Data Security. SSL finns idag integrerat i produkter för kommunikation över Internet från många olika företag. Protokollet kan till exempel användas för att skydda ett kontokortsnummer när det skickas från en kund till ett säljande företag.

I den första fasen av SSL autenticieras servern. På klientens begäran skickar servern över sitt digitala certifikat till klienten, som verifierar det. I nästa steg skickar servern

ett kort meddelande med en digital signatur, utförd med dess privata nyckel. Klienten använder serverns publika nyckel för att kontrollera signaturen. Servern autenticieras av klienten. Klienten sänder därefter ett meddelande som innehåller en nyckel för symmetrisk kryptering till servern, krypterad med serverns publika nyckel. Den symmetriska nyckeln används sedan för att kryptera all information som skickas mellan klienten och servern.

Den andra fasen av SSL, som inte är obligatorisk, består i att klienten autenticieras av servern. Den görs genom att klienten får svara på ett slumptal från servern. Servern skickar ett meddelande till klienten, som sätter en digital signatur på det och skickar tillbaka det tillsammans med sitt digitala certifikat. Servern kan genom att kontrollera den digitala signaturen vara säker på klientens identitet.

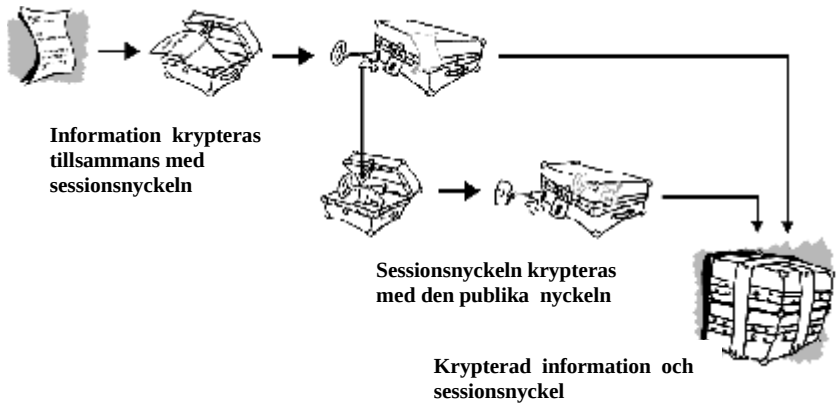
För att kunna utnyttja SSL måste åtminstone servern förses med ett digitalt certifikat. Då kan servern autenticieras av klienten. Om den andra fasen ska användas, där klienten autenticieras av servern, måste klienten ha ett certifikat (som användaren skaffar). För att verifiera ett digitalt certifikat måste den publika nyckeln för den CA som utfärdat certifikatet användas. I dagens webbläsare finns de publika nycklarna för flera kända CA:er inlagda för att webbläsaren ska kunna verifiera certifikat utgivna av dessa.

Möjligheten att autenticiera klienten används ännu så länge inte i någon större omfattning och det är just detta som får många att kritisera säkerheten med SSL. Utan autenticiering går det visserligen att hålla information konfidentiell, men identiteten hos den som skickar information till en server kan inte verifieras.

2.3.5.2 Pretty Good Privacy (PGP)

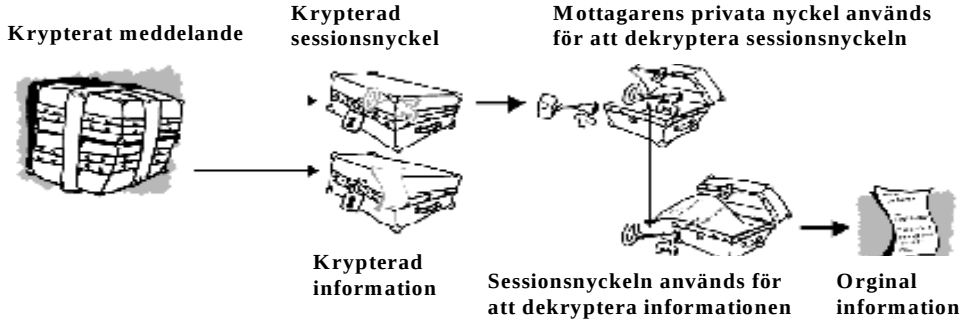
Informationen i detta avsnitt är hämtat från PGPinternationals officiella hemsida (PGPi, 2000). Pretty Good Privacy (PGP) kombinerar konventionell kryptering och PKI. PGP är ett hybridkryptosystem, vilket betyder att det hanterar både symmetrisk och asymmetrisk kryptering. När en användare krypterar text med PGP komprimeras först informationen. Detta minskar transmissionstid och diskutrymme och viktigast av allt, det stärker krypteringen. Detta då de flesta tekniker för krypteringsanalys utnyttjar mönster i den krypterade informationen för att knäcka krypteringsalgoritmen. Genom att komprimera informationen minskas dessa mönster.

PGP skapar en sessionsnyckel vilket är en hemlig engångs nyckel. Denna nyckel utgörs av ett slumpmässigt tal som genereras av rörelserna från användarens pekdon och den frekvens som användaren trycker ner tangenterna i. Sessionsnyckeln används tillsammans med en konventionell krypteringsalgoritm för att kryptera informationen. När informationen blivit krypterad krypteras sessionsnyckeln till mottagarens publika nyckel. Denna publika krypterade sessions nyckel skickas tillsammans med den krypterade informationen till mottagaren.



Figur 4. Figuren visar PGP kryptering (PGPi, 2000)

Dekrypteringen sker på motsatt vis. Mottagarens kopia av PGP använder sin privata nyckel för att återskapa den temporära sessionsnyckeln, som sedan PGP använder för att dekryptera den konventionella krypterade informationen.



Figur 5. Figuren visar PGP dekryptering (PGPi, 2000)

Denna kombination av bekvämligheten med publik nyckelhantering och konventionell krypterings höga hastighet. Kombinationen av de båda krypterings metoderna förenar bekvämligheten med publik nyckelhantering med den symmetriska krypteringens höga hastighet. Symmetrisk kryptering är ungefär 1000 gånger snabbare än kryptering med publik nyckel. Publik nyckelkryptering däremot har bl a stora fördelar med sin distribution av krypteringsnycklar. I kombination kan både prestanda och nyckeldistribution förbättras utan att offra säkerhet.

2.3.5.3 Secure Electronic Transaction (SET)

Informationen i detta avsnitt kommer från den officiella hemsidan för SET. SET är en standard som möjliggör säkra kontokortsbetalningar på Internet. SET är ett program som är tänkt att fungera som en helhetslösning för kontokortsbetalning. Det är en mer specialiserad lösning som i princip fungerar som en virtuell elektronisk plånbok. SET har utvecklats av bl a Visa, Master Card, IBM, Netscape, Microsoft och RSA Data Security. Standarden syftar till att verifiera, godkänna och skydda säljare, köpare och banken när en kortbetalning genomförs på Internet. SET används idag mest av banker samt ett fåtal nätbutiker, de flesta använder dock fortfarande någon form av SSL-lösning. Genom kryptering garanteras och skyddas köparens och säljarens identitet samt den betalningsinformation som sänds dem emellan.

SET standarden måste garantera att innehållet i meddelandet inte förändras under en transaktion mellan sändare och mottagare. Betalningsinformation från kortinnehavaren till försäljaren inkluderar orderinformation, personlig data och betalningsinstruktioner. Om någon del av en transmission ändras kommer inte betalningen att kunna genomföras på ett korrekt sätt.

SET använder kryptering för att säkra meddelandens konfidentialitet. I SET krypteras data med en slumpmässig genererad symmetrisk krypteringsnyckel. Denna nyckel krypteras med meddelandemottagarens publika nyckel. Detta kallas för det digitala kuvertet av ett meddelande och skickas till mottagaren tillsammans med det krypterade meddelandet. Efter att ha mottagit det digitala kuvertet, dekrypterar mottagaren det och använder sin privata nyckel till att skaffa den slumpmässigt genererade symmetriska nyckeln och använder sedan denna nyckel till att dekryptera meddelandet.

På grund av det matematiska egenskaperna hos den publika och privata nyckeln kan data som krypterats med den ena nyckeln dekrypteras med den andra. Detta gör det möjligt för avsändaren att använda sin privata nyckel till att kryptera meddelandet. Mottagaren kan verifiera vem som sänt meddelandet genom att dekryptera avsändarens publika nyckel.

Användaren har två asymmetriska nyckelpar. Ett par, för utbyte av nycklar, som används för kryptering och dekryptering. Ett annat par för skapande och verifiering av digitala signaturer. I skapandet av digitala signaturer har den publika och privata nyckeln ombytta roller då den privata nyckeln används till kryptering/signering och den publika nyckeln används till dekryptering/verifiering. Innan två parter kan använda publik nyckelkryptering för att utföra handel, vill båda parter vara säker på att den andra parten är autenticierad. För att säkra alla parters identitet använder SET certifikat som utfärdats av tredje part, en certifikat autenticierare (CA). Eftersom användaren har två nyckelpar har denne även två certifikat som verifierar vardera nyckelpar. Båda certifikatet skapas och signeras samtidigt.

2.4 Sammanfattning - baserat på genomgången av säkerhet och produktivitet.

Jag har redogjort för produktivitetsbegreppet och vad jag menar med produktivitet i detta sammanhang. Jag har även visat på de hot och faror vilka förekommer i samband med säkerhet. Efter detta har jag kommit fram till att kryptering är en av de mest fundamentala tekniska aspekterna i säkerhetslösningar, och även att användarna är viktiga element i ett säkert system även om de kanske inte alltid inser detta själva.

I och med kryptering har jag tagit upp fundamentala begrepp såsom digitala certifikat, autenticiering, symmetrisk och asymmetrisk kryptering samt några olika krypteringslösningar. Dessa lösningar är idag de mest använda, och den normale användaren kommer till och från i kontakt med dem när hon t.ex. sköter sina betalningar via Internet eller kontrollerar sin e-post. De olika säkerhetslösningarna kräver även olika mycket av användarens uppmärksamhet, alltifrån aktivt tankearbete till ingenting alls. Vidare är de olika säkerhetslösningarna olika i hur stark kryptering de använder och hur säkra de kan anses vara i olika sammanhang. Både användarmedverkan och krypteringsstyrka är aspekter som är intressanta ur resurssynpunkt, i det här fallet användarnas tid och systemresurser hos de maskinvaror på vilka säkerhetssystemen implementerats och körs. Systemresurser

såsom minne, processtid och bandbredd är vitala för om systemen blir alltför långsamma kommer de vare sig att kunna hantera de mänger data som är önskvärt eller kunna motivera användarna till att använda dem.

2.5 Källkritik

Detta avsnitt är avsett för att ge läsaren en möjlighet att bedöma materialet i denna uppsats utifrån objektivitet och trovärdighet. Patel och Davidsson (1994) nämner följande kriterier som intressanta:

- När och var ett dokument tillkommit?
- Vilket syfte hade författaren med dokumentet?
- Var upphovsmannen en person med kännedom inom området eller var han lekman?

När det gäller tillkomsten av samtliga dokument är detta återgivet i källreferensen. Tillkomsten är av intresse då den speglar dels dokumentets aktualitet, dels i vilken miljö dokumentet skapats i. Syftet med dokumenten har sin relevans genom att man på detta sätt får en inblick i om meningen är av exempelvis undervisningssyfte, eller ren propaganda. Detta för att kunna avgöra dokumentets objektivitet. Författarens kunnighet inom området kan också anses relevant. Inte för att en kunnig person på ett bättre sätt sprider sin kunskap objektivt, men kvaliteten och tillförlitligheten till materialet ökar.

Jag ämnar här att redogöra för de typer av källor jag använt mig av så att läsaren kan förstå hur jag källkritiskt uppfattat dem. De olika typer av källor jag använt mig av i arbetet är böcker, tidskrifter samt internetadresser. Jag har även kontaktat olika företag i syfte att komplettera och klargöra den information jag skaffat mig. De intervjuer som gjorts bifogas som Bilaga 2 - 4.

2.5.1 Böcker

Många av de böcker jag valt att referera till, såsom exempelvis Silberschatz och Galvin är böcker som använts av mig i mina studier på högskolan. Detta ger en trygghet eftersom böckerna då ges status som god källa. Facklitteratur och studentlitteratur ger en trygghetskänsla då innehållet har granskats och blivit godkännt av ett förlag.

2.5.2 Tidskrifter

Vad gäller tidskrifter tar jag endast fackpress i beaktande då övriga typer av tidsskrifter ej är relevanta för detta arbete, dels pga arbetets tekniska natur men även eftersom kvaliteten inte kan garanteras. Kvaliten garanteras inte bara för att det är fackpress- även här kan subjektiva utlåtanden och felaktigheter förekomma. Därför är det av intresse att ta reda på hur publiceringsprocessen ser ut. Efter kontakt med bl a Bäckström från Computer Sweden fick jag veta att all information i omgångar korrekturläses av tekniskt sakunniga innan redigering. Chefredaktören har själv teknisk bakgrund. Som en av de största datortidningarna i Sverige kan är det viktigt för företaget att ha en objektiv profil. Naturligtvis kan information i form av intervjuer vara subjektiva men det är upp till läsaren att bedöma den intervjades objektivitet.

2.5.3 Internetadresser

Denna källa är den som kan föranleda mest misstänksamhet gällande dokumentets giltighet, eftersom vem som helst kan lägga upp en hemsida och säga vad som helst utan att det behöver finnas ett uns sanning i det. För att då säkra läsaren om att infomrationen i arbetet har substans har jag använt mig av Web-adresser såsom officiella hemsidor för t ex Microsoft, SSL, PGP och SET. Syftet med hemsidorna är visserligen att på ett fördelaktigt sätt presentera sina produkter men även att besvara frågor på ett objektivt sätt. Detta då det ligger i deras intresse att produkten fungerar.

Vad gäller andra typer av Internet-källhänvisningar kan vi som exempel se till Slashdot som är kombinerar nyhetsservice och forum. Slashdot är dessutom tekniskt inriktat och flertalet av de som läser artiklarna är väl insatta i ämnena som disskuteras. Varje artikel som postats på Slashdot får runt 150-300 KB kritik och kommentarer, vilket motsvarar ungefär 25-50 A4 sidor text. Förutom att artiklarna mestadels är skrivna av fackfolk, får de alltså en grundlig granskning av läsarna själva. De initiala misstankar man kan få av att författarna oftast postar under pseudonym och att texten ligger på Internet behöver inte nödvändigtvis vara välgrundade. Faktum är att det med största sannolikhet är så att artiklar i tidningar och böcker får betydligt mindre kritik, vilket visserligen inte betyder att de är sämre – men väl att om en felaktig artikel postats så uppdagas detta strax. Dock behöver detta inte innebära att alla synpunkter och kommentarer är objektiva – det råder säkerligen en del mer eller mindre ogrundade åsikter på dessa sidor. Detta är dock ett problem som gäller alla källor. Det finns alltid en risk att författarens egna åsikter och synpunkter färgar materialet.

3 Problemställning

Samtidigt som jag belyst att säkerhet är nödvändig för produktivitet menar jag även att dessa två står i ett motsatsförhållande till varandra.

I detta arbete kommer jag att försöka besvara frågorna:

- *Hur kombineras säkerhet och produktivitet inom området tekniskdatasäkerhet?*
- *Verkar högre säkerhet leda till att systemet blir svårare att använda?*

Den första frågan kommer ej att redogöras för i allmänhet, utan speciellt för säkerhetslösningarna PGP, SSL och SET.

I och med att utvecklingen går framåt och starkare kryptering krävs, innebär detta att kraftigare maskiner behövs för att hantera av de matematiska algoritmer som används vid kryptering (PGPi, 2000). Detta är oftast inget problem för den enskilde användaren därför att denne inte reagerar speciellt mycket om krypteringen tar en hundradelssekund eller en sekund. Det är inget som stör dennes arbete i t ex ett PKI-system som är ämnat att användas för säker kommunikation mellan människor. Om man istället jämför med ett handelssystem som hanterar daglig börshandel där det skickas tusentals transaktioner under vissa tidsbegränsade tidpunkter, är kravet oerhört stort för att inte säga avgörande på att säkerheten i systemet inte får påverka hanteringen /produktiviteten negativt (Lindberg och Ohlsson, 2000; Ohlsson, 2000). Om den gör det måste man välja en mindre säker lösning som går fortare, dvs svagare krypteringsalgoritmer som inte tar så mycket processortid att genomföra. Man vill givetvis ha en så säker lösning som möjligt Detta innebär att det som efterfrågas är en så säker lösning som möjligt vilken ändå klarar av de krav man har på prestanda och pris – speciellt är produktivitet, i detta fall utnyttjande av systemresurser såsom minne, processtid och bandbredd, vitalt då en mindre produktiv lösning ger ett högre pris för samma prestanda.

Säkerhet handlar även om att inte förlora information (Freese och Holmberg, 1993). En säkerhetslösning är inte bättre en den svagaste länken. I detta fall får man nog se till människan som den svaga länken (Lotsson 2000). Säkerhetsdosor kan försvinna. Genom slarv och dåliga lösenord kan intrång ske på systemet. När det blir mycket att hålla reda på så tycker de flesta att det blir komplext. Å andra sidan, känner vi oss säkra på systemet, är vi kanske beredda att avstå från lite produktivitet. Är det så att produktiviteten kommer med ökad säkerhet?

3.1 Avgränsning

Det problemområde som jag valt är att jämföra olika säkerhetslösningar med avseende på hur väl de löser de problem de är avsedda att åtgärda samt hur produktiva de är. De säkerhetslösningar som jag valt att jämföra här är SSL (Secure Socket Layer) och PGP (Pretty Good Privacy) – dessa system är i grunden rätt olika och avsikten är att ge en bättre överblick än om man hade valt att jämföra likartade system. De aspekter hos systemen som jämförs är:

1. Hur stark säkerhetslösningen är (Krypteringsstyrka).
2. Hur resurseffektiv den är i fråga om primär/sekundärminne, processtid samt bandbredd.
3. Interoperabilitet och kompatibilitet – vilka system den finns till och hur väl olika implementationer fungerar tillsammans.

4. Användningsförfarande - hur säkerhetslösningarna används av användarna.

När det kommer till krypteringsstyrkan är frågeställningen relevant då större säkerhet uppnås med längre nycklar, vilket även kräver mer systemresurser. Krypteringsstyrkan är inte den enda säkerhetsaspekten som är relevant men den är viktig.

Resursproduktiviteten är direkt relevant för min primära frågeställning då jag jämför förhållandet mellan produktivitet och säkerhet. Detta för att sedan ta ställning till om högre säkerhet verkar leda till att systemet blir svårare att använda.

Interoperabilitet och kompatibilitet är mer indirekta produktivitetsaspekter som inte direkt berör minne, processtid och bandbredd utan snarare ser till huruvida systemet ens går att använda.

Användningsförfarandet är av intresse då jag avser utröna huruvida ökad säkerhet verkar leda till svåranvändbarhet. Leder säkerhetsaspekten till att användaren får fler rutiner i sitt datorutövande?

Dessa frågor hjälper till att besvara huvudfrågorna i detta arbete. Det måste dock poängteras att huvudfrågorna i allmän mening skulle kunna besvaras på något annat vis, men för att knyta an till datorsäkerhet såsom kryptering - en tämligen teknisk lösning - torde resultatet av denna frågeställning vara tillfredställande.

En teknisk analys ligger i tiden då en distribuerad miljö, som Internet, inte har samma möjlighet till att utbilda eller få användaren att följa säkerhetsrutiner och protokoll på samma sätt som en vanlig arbetsplats (Thylander, 2000). Det användaren kommer i kontakt med och som går att påverka är gränssnittet eller bakomliggande teknik såsom säkerhetslösningarna PGP, SSL och SET. Därför är det intressant att se till de tekniska aspekterna då gränssnittet, oavsett hur det är utformat, inte kan förbättra den tekniska prestandan.

Produktivitet mäts också i ekonomiska termer varpå ett företag, genom att välja lämplig, anpassad säkerhetslösning kan utnyttja befintliga resurser och på så sätt öka maskinernas livslängd. Om ett företag väljer en säkerhetslösning med lång krypteringstid kan företaget tvingas skaffa snabbare datorer. Denna lösning blir då kostsam. Alternativt behålla de gamla datorerna och få ett system som tar väldigt lång tid, vilket inte kan ses som produktivt.

Tekniken är det grundläggande. Vilket val som görs mellan säkerhet och produktivitet/ effektivitet är beroende av de resurskrav lösningen har. Det spelar ingen roll hur säkerhetsmedveten användaren än är om inte krypteringen eller de tekniska resurserna är tillräckliga.

Med dessa resonemang i bagaget inleder jag nästa avsnitt, nämligen metoddelen.

4 Metod

Produktiviteten går att belysa utifrån två aspekter. Den första behandlar användarens användning av systemet och den andra aspekten berör systemresurser som exempelvis minne, processtid, diskutrymme och bandbredd.

Denna syn på produktivitet ska sättas i relation till vad vi menar med säkerhet. Även här kan begreppet avse minst två olika infallsvinklar, dels den tekniska säkerheten, dels säkerhetssynen ur användarens perspektiv.

När det kommer till den normale användarens kunskaper rörandes säkerhetsaspekter är den tekniskt tämligen svag. Och när en användare får en fråga rörandes produktivitet så handlar det mestadels om sätt att underlätta användarens arbetsuppgifter. Denna infallsvinkel är visserligen intressant, men kräver mycket tid och resurser för att genomföra på ett tillfredsställande sätt. Vidare är det få användare som har en god kunskap om säkerhetsfrågor och vid en utfrågning skulle antagligen de tekniska aspekterna mer eller mindre utelämnas till förmån för mer elementära åtgärder som t ex att inte skriva upp sina lösenord.

4.1 Fallstudie

Användaren är dock den för vilken man gör en bedömning av huruvida systemet är både säkert och produktivt eftersom användaren utnyttjar systemet för att kunna utföra sina sysslor. För att ta reda på hur systemet uppfattas av användaren kan vi använda oss av nån typ av fallstudie. Med fallstudie menas en undersökning som görs på en mindre avgränsad grupp. I detta fall är det av intresse att se till situationen, där olika säkerhetslösningar jämförs. Vi utgår i detta fall från ett helhetsperspektiv för att få så täckande information som möjligt (Patel och Davidson, 1994).

Detta kan göras genom att be ett antal personer lösa uppgifter eller använda sig av olika typer av säkerhetssystem i en relativt naturlig miljö. För att sedan få jämförbara uppgifter kan man antingen räkna antal fel som personerna gör på olika system, eller jämföra hur lång tid det tar för dem att lösa uppgifterna. Detta ger resultat vilka exempelvis kan användas då man studerar produktivitet respektive effektivitet. Denna jämförelse blir intressant då ett system där användarna lätt gör fel kan betraktas som icke produktivt. Man kan även studera användarnas hanterande av systemet dels genom att iaktta dem, antingen personligen eller med hjälp av en videokamera. Man be användarna att lösa problemen i grupp eller enskilt. Man kan även mäta olika parametrar hos systemet då användaren löser uppgifterna.

Fördelen med att få fram kvantitativa uppgifter är att det är lätt att göra en jämförelse mellan olika lösningar. Det går att få fram nån form av rangordning. Nackdelen är att den typen av mätvärden fortfarande inte på alla sätt belyser användarens synpunkter på systemet. Att det är snabbt kan man mäta kvantitativt, men att tillmäta ett värde till egenskaper som ’enkel’, ’intuitiv’, ’krånglig’, ’lättanvänd’ och ’användbar’ är inte alls lika enkelt då dessa är subjektiva eller kvalitativa aspekter. Dessutom brukar människor göra implicita uttalanden, på så sätt att även det som är outtalat ska förstås. (Pohl, 1994)

Genom att iaktta användarna ser man dem operera i sin naturliga miljö. Man kan be dem ”tänka högt” för att få inblick i vad de anser vara svårt eller otydligt och man ser hur de hanterar olika situationer. Problemet när användaren sitter själv är att hon tenderar att tystna vid problem, genom låta dem samarbeta för de en diskussion med

varandra. Det är viktigt att tänka på att alla iakttagelser där försökspersonen vet att hon är iakttagen kan leda till onaturligt beteende och blockering p g a nervositet.

4.2 Experiment

Experiment är enligt Patel & Davidson (1994) en beteckning på en undersöknings upplägg där man studerar några enstaka variabler och försöker få kontroll över annat som kan påverka dessa variabler. Genomgående för de många olika sätten att lägga upp experiment på är att man försöker kontrollera alla faktorer som kan påverka den eller de oberoende variablerna och den beroende variabeln. De viktigaste faktorerna att kontrollera är individfaktorer och situationsfaktorer.

Det vi söker i denna metod är någon form av korrelation mellan produktivitet och säkerhet. Olika typer av experiment skulle t ex kunna omfatta intrångsförsök, benchmarking samt lastmätning.

För att det ska bli tekniskt möjligt att jämföra säkerhetslösningarna behövs det ett operativsystem att arbeta emot. Olika operativsystem är t ex Windows NT, Open BSD, Solaris och Linux. Genom att installera olika säkerhetslösningar på dessa går det att se om det föreligger någon skillnad i produktivitet mellan de olika systemen. Installationen i sig kan vara intressant vid bedömning av hur jobbigt det är att sätta upp systemet samt vid en bedömning av hur lätt systemet är att konfigurera. Vi kan jämföra olika säkerhetslösningar som löser samma problem. På detta sätt kan man även tänka sig att det går att jämföra om produktiviteten är beroende av operativsystem eller om det handlar om vald säkerhetslösning. Operativsystemet kan vara en viktig faktor i diskussionen runt produktivitet. Vad som menas med detta är att om inte operativsystemet, i detta fall, accepterar vald säkerhetslösning fungerar inte lösningen alls (Åkerman, 2000). Operativsystemet kan även i sig göra produktiviteten högre, bl a genom att vissa OS har inbyggt stöd för exempelvis dedicerad[6] krypteringshårdvara (Silberschatz och Galvin 1994).

4.2.1 Attack mot säkerhetssystemen

Problemställningen kan besvaras genom att jämföra olika säkerhetslösningar utifrån de kriterier vi ställt på produktivitet för att sedan jämföra hur lätt det är att göra intrång i systemet.

Denna metod kan ske genom någon form av cracking[7]. Dock kräver denna metod omfattande och djupa kunskaper i operativsystem, kryptering, protokoll, avlyssning osv för att kunna genomföra på ett tillfredsställande sätt (Freese och Holmberg, 1993).

4.2.2 Benchmarking

Benchmarking är en form av produktivitetsmätning där man för ett program (eller system) mäter hur mycket tid det använder för att utföra olika uppgifter. Tidsåtgången jämförs sedan med samma mätningar för andra program eller samma program under andra betingelser. Nackdelen med benchmarking är att det inte tar hänsyn till hur mycket resurser som används för att utföra en uppgift. Två program som tar lika lång tid på sig att utföra en uppgift kan ha vitt skilda lastprofiler – detta betyder inte att benchmarking är meningslöst, men man måste vara medveten om vad det är man egentligen mäter (Hockney, 1995).

[6] Hårdvara som endast utför en viss uppgift såsom kryptering.
[7] En cracker är en person som bryter sig in i system i motsats till en hacker som är en duktig programmerare.

4.2.3 Lastmätning

Lastmätning är besläktat med benchmarking, men det man mäter är inte tid, utan resursåtgång. Man ser alltså inte till hur lång tid en process tar, utan till vilka resurser som krävs. Givetvis spelar tiden in även här, men inte på ett lika direkt sätt som i benchmarking. En lastmätning är vanligtvis något svårare ett genomföra än benchmarking då det är fler faktorer som man måste börja med att ta hänsyn till, men det kan i gengäld resultera i ett mer användbart resultat eftersom att en lastmätning bättre beskriver hur beteendet förändras vid hög last (t.ex. mätt i transaktioner/s) – lastmätning kan på sätt och vis ses som en utökad form av benchmarking. Detta är av speciellt intresse då man betraktar inte bara klientsystem, utan även serversystem. För serversystem är det viktigt att man inte får en lägre produktivitet än man räknat med då detta kan vara kritiskt och med lastmätning så får man en bättre bild av hur tidsåtgången ökar med ökat antal transaktioner än man får med benchmarking. (Hockney, 1995)

5 Val av metod

Samtliga metoder som jag tagit upp kan belyser frågeställningarna rörande produktivitet och säkerhet. Vissa av dessa metoder är dock lämpligare än andra. Fallstudie väljer jag bort som metod då denna metod dels är oerhört resurskrävande på så sätt att man förutom att analysera datan även måste ägna en hel del tid och material och användarstudier både vid utformning och genomförande av dessa. Då detta arbete dessutom är ganska tekniskt vinklat är dessutom en fallstudie kanske inte helt lämplig då dessa bättre lämpar sig för processer och förändringar (Patel och Davidson, 1994). Att jämföra säkerhetsaspekterna hos de olika säkerhetsalternativen, PGP, SSL och SET rent praktiskt skulle bli svårt – detta eftersom att det skulle kräva djupgående kunskaper i kryptering, programmering och säkerhet för att t ex attackera systemen och jämföra hur väl de står emot dessa attacker. Istället väljer jag att på teoretisk väg, via litteraturstudier jämföra dessa. För att undersöka produktiviteten väljer jag en mer pragmatisk inriktning nämligen att lastmäta olika säkerhetslösningar som jag kommer att installera under operativsystemen Debian GNU/Linux och Windows 2000. Lastmätning, i detta fall en jämförelse av de tekniska produktivitetsaspekterna minne, processtid, bandbredd och diskutrymme ses som en variation på benchmarking men skiljer sig åt då lastmätningen tar hänsyn till fler parametrar och ger ett mer användbart mått på systemens beteende (Hockney, 1995).

5.1 Lastmätning

Att mäta ett programs resursanvändande går ut på att göra en uppskattning av dess resurskrav och beteende under olika omständigheter (Hockney, 1995). Styrkan i denna metod är att man får ett resultat som enkelt går att analysera. Att göra en exakt bedömning av resursåtgången är i de flesta fall inte möjligt, men detta utesluter dock inte att man ändå kan få en god förståelsemodell av vad ett program behöver och hur det uppför sig. Detta beror på att man inte kan eliminera alla övriga faktorer som bidrar med brus till mätningen. Dessa kan t ex vara olika saker operativsystemet gör. När man sedan skall ta sig an att göra själva mätningarna, är detta ett mer eller mindre komplext problem (Hockney, 1995). För en del typer av resurser, t ex sekundärminne så kan en mätning bli relativt enkel medan den för t ex processtid eller bandbredd kan bli ganska komplex. Komplexiteten i detta ligger dels i att det är svårt att få exakta siffror – man kan inte mäta utan att påverka resultatet – dels i att andra processer och eventuell mätning av deras resursutnyttjande kan inverka genom att det kan vara svårt att effektivt avgränsa vilka processer som använder vilka delar av resurserna (Hockney, 1995). Som tur är så finns det bra verktyg för att mäta många typer av resursanvändning, att veta hur mycket resurser av olika slag som går åt är ju praktiskt i många sammanhang. Några exempel på lastmätningsverktyg följer:

5.1.1 Lastmätningsverktyget Top

Informationen i detta avsnitt är hämtat från Top(2000). Top är ett verktyg för att mäta processtids och primärminnes användning. Top används primärt i UNIX system. Top rapporterar hur mycket minne ett program använder, samt hur stor del av den totala processkapaciteten som det använder. Top rapporterar även hur mycket minne som finns ledigt och hur mycket processkapacitet som ej används - dessa värden kan vara av intresse om direkta mätningar visar sig svåra att genomföra.

5.1.2 Aktivitetshanteraren

Informationen i detta avsnitt är baserar sig på Microsoft (2000). Till Windows 2000 medföljer även där olika verktyg för övervakning och administration av systemet, däribland aktivitetshanteraren som ger ungefär samma information som Top[8] men används även för att t ex avsluta processer. Aktivitetshanteraren kan även generera enkla grafer tyvärr saknas dock möjlighet att logga resultaten – man blir helt enkelt tvungen att manuellt ta tid och notera resultaten.

5.1.3 Resursutnyttjande

När man nu skall jämföra hur två program beter sig, är det bara att starta dem och se hur mycket resurser de använder, eller? Tyvärr så är det inte riktigt så enkelt. Till att börja med så ger olika operativsystem och maskinvaror olika förutsättningar för de program man vill testa och de kan bete sig ganska olika beroende på vilket OS det körs under. Detta problem kommer man runt genom att testa programmen på samma maskiner och OS för var och en av programmen. På så sätt får man en mer relativ mätning. Som tidigare nämts i detta arbete, är det inte nödvändigt med mer precisa resultat för att få en känsla för hur programmen beter sig och hur mycket resurser de kräver. Dessutom blir mätvärderna så pass distinkta att en jämförande analys kan göras. För den typ av jämförelse jag vill göra, utifrån processtid samt minneshantering, ger lastmätningsverktygen Top och aktivitetshaneraren mig fullgoda resultat. De faktorer jag nämner nedan är för att uppmärksamma läsaren på andra faktorer som också har en betydelse i detta sammanhang. De faktorerna är:

- vilken CPU (Central Processing Unit) som används, (Aceshardware, 2000)
- vilken kringhårdvara som finns, (Ars Technica, 2000)
- vad för typ av buss systemet har, (Aceshardware, 2000)
- vad för typ av minne som används, (Aceshardware, 2000)
- vilken frekvens minnet klarar, (Aceshardware, 2000)
- hur minnet sitter (s k interleaving), (Aceshardware, 2000)
- om man har en eller flera CPU:er, (Ars Technica, 2000)
- hur maskinvaran är konfigurerad, (Ars Technica, 2000)
- vad som eventuellt finns buffrat av OS:et, (Silberschatz och Galvin, 1993)
- vilka inställningar man har gjort i OS:et, (Silberschatz och Galvin, 1993)
- vilka program som körs samtidigt (inklusive olika systemtjänster). (Silberschatz och Galvin, 1993)

Förutom detta, kan olika resurser vara beroende av varandra vilket kan ge upphov till svårtolkade resultat om mätningar exempelvis görs på system med olika mängder primärminne, menar Silberschatz och Galvin (1994). Ytterligare en faktor som försvårar är att man ofta själv inte kan avgöra vilka faktorer som är väsentliga för ett programs prestanda och därför inte vet om det råkar vara någon felkonfigurering i OS:et, ett problem med någon specifik maskinvara, eller om programmet i sig självt orsakar den last som man mäter (Silberschatz och Galvin 1994). Att det är svårt att jämföra olika system beror först och främst på att olika egenskaper eventuellt inte går

[8] Vilka processer som finns samt lite om vad de har för sig i fråga om minnesanvändning och CPU tid.

att jämföra direkt. Detta är som i det klassiska exemplet med att jämföra äpplen och päron där det ena inte kan uttryckas med hjälp av det andra. Två fundamentalt olika lösningar kan lösa samma problem, men det betyder inte nödvändigtvis att hur de löser problemen är helt jämförbara (Hockney, 1995). Det man får nöja sig med i så gott som samtliga fall är att man får fingervisningar om programmens relativa meriter. Ovanstående resonemang har gett kunskap om vad som kan påverka resultaten och därför kan dessa tolkas utifrån vetskapen att det kan finns mycket brus i mätresultaten. Det finns två huvudsakliga typer av brus eller mätfel, slumpmässiga och systematiska fel. Slumpmässiga fel blir enligt CGS (centrala gränsvärdessatsen) och de stora talens lag normalfördelade om tillräckligt många mätningar görs. (Carlsson et al., 1997). För att få god bruskontroll genomför jag många mätningar, dvs 30 stycken per program. Hur skall jag då genomföra en lastmätning? Jo, programmen körs och sedan rapporteras hur mycket resurser som de resursanvändningsprogram använder för olika scenarier.

De scenarier jag använder mig av i mätningen är följande (Notera att alla scenarier inte är applicerbara på alla program):

1. Programmet aktivt men sysslolöst.
2. Programmet sysselsatt med kryptering.

De egenskaper som mäts är processtidskrav och primärminneskrav. Kraven som ställs på tillgänglig bandbredd undersöks ifrån ett rent teoretisk perspektiv då detta är den aspekt som bäst lämpar sig för sådan analys. Detta pga att det är svårare att finna/göra en testmiljö för bandbreddsmätningar[9] samt att det är färre parametrar man behöver ta hänsyn till (Halshall, 1993). Resultaten jag får fram via lastmätningen av de olika säkerhetsprogrammen jämförs sedan med resultaten av motsvarande alternativ då säkerhetslösningarna utesluts. Detta görs separat för de olika operativsystem som undersöks.

Därefter jämförs resultaten mellan de olika operativsystemen och sätts i perspektiv till den uppmätta absoluta lasten. Det uppskattade bandbreddskravet är per definition detsamma oavsett vilket operativsystem som används eftersom detta beror på protokollet och inte på hur det som skall skickas sammanställs (Bengtsson, 2000). Slutligen så görs en uppskattning av de olika säkerhetssystemens totala lastkostnad i relation till deras säkerhetsmeriter, detta med avsikten att dels försöka ge en helhetsbild av potentiella användningsområden och meriterande egenskaper, men främst för att göra en bedömning av sambandet mellan säkerhet och produktivitet. Säkerhetsmeriterna som tas i beaktande är krypteringsstyrka, nyckellängd, krypteringsalgoritm och användningsområden. Dessa jämförs med produktivitetsmåtten flexibilitet, bundenhet, transparens, interoperabilitet, resursanvändning och resurskrav. Säkerhets- och produktivitetsmeriterna beskrivs kortfattat nedan:

- Krypteringsstyrka. En uppskattning av hur stark krypteringen är baserat på nyckellängd och algoritm. Detta beskriver kort hur säker man kan vara på att krypteringen inte knäcks. (Sjögren, 1996)
- Nyckellängd. Hur långa nycklar krypteringen använder, detta reflekterar till viss del hur säker krypteringen är – men även hur mycket resurser som behövs för att

[9] För detta krävs för mycket hårdvara.

kryptera eftersom att en längre nyckel oftast kräver mer beräkningar. (Sjögren, 1996)

- Krypteringsalgoritm. Vilken eller vilka krypteringsalgoritmer som användes i de test jag skall göra. Detta ger en indikation på hur säker och resursintensiv krypteringen är eftersom att olika krypteringsalgoritmer kräver olika mycket beräkningar och minne för att kryptera. (Johansson, 1998)
- Användningsområden. Vad en säkerhetslösning kan användas till i fråga om kryptering, signering, autenticiering och Icke-förnekelse. Dessa parametrar utgör vad som menas med god kryptering. (Johansson, 1998)
- Flexibilitet. Hur anpassningsbar en säkerhetslösning är. Går det att anpassa den till just de behov man har? Flexibiliteten avser mer specifikt anpassning i användning och ska inte förväxlas med interoperabilitet vilket avser om systemet fungerar på och mellan olika operativsystem m m. (Freese, 1993)
- Bundenhet. Är säkerhetslösningen normalt bunden till en fysisk maskin eller inte. Att binda ett säkerhetssystem till en fysisk maskin görs normalt av säkerhetsskäl. Visserligen kan man se bundenhet som ett specialfall av flexibilitet – men det särskiljer sig just på grund av säkerhetsaspekten. (SET, 2000)
- Transparens. Hur mycket märker man av att använda säkerhetssystemet. (SSL, 2000)
- Interoperabilitet. Om systemet går att använda tillsammans med olika programvaror, operativsystem och nätverk I den här rapporten betraktas endast operativsystem. (Åkerman, 2000)
- Resursanvändning. Vilka resurser som verkligen används av säkerhetssystemet i fråga om minne, processtid och bandbredd. (Hockney, 1995)
- Resurskrav. Vilka minimikraven i fråga om resurser är för systemet. (Hockney, 1995)

5.2 Val av scenarier

I det första scenariot är programmet aktivt men det utför ingenting. Detta tillstånd är av intresse då en jämförelse går att göra, med när programmet är aktivt, och på så sätt se hur mycket resurser själva krypteringsarbetet kräver.

I det andra scenariot är programmet aktivt med att läsa, kryptera och skriva data. På detta sätt går det att se hur mycket resurser programmet kräver när det arbetar och man kan även se om detta varierar beroende på vad för data som ska krypteras.

5.3 Val av testdata

Då misstankar föreligger om att resurskraven för kryptering kan variera med indata så prövas två olika typer av data . Jag har valt filer bestående av data med hög respektive låg entropi. Med entropi menas just i detta fall någon form av informationstäthet[10] på så sätt att data med låg entropi har låg täthet och mycket struktur medan data med hög entropi har hög täthet och lite struktur. (Sjögren, 1996) Data med låg entropi är lätt att komprimera, vilket resulterar i data med hög entropi. Vid kryptering vill man i regel få så hög entropi som möjligt i den krypterade datan för att försvåra t ex statistiska

[10] Informationsmängd per meddelandeenhet.

angrepp på krypteringen. Eftersom att komprimering ökar entropin används detta ibland som ett försteg till kryptering (Sjögren 1996).

Hur lätt det är att komprimera data beror på hur hög entropi den har. Därför blir det intressant att prova båda typerna av data. De olika data jag valt är en textfil som har låg entropi samt en bild (JFIF/JPEG) vilken har hög entropi (Sjögren, 1996).

5.4 Val av OS

De två operativsystem vilka jag valt är Windows 2000 samt Debian GNU/Linux. Dessa två har jag valt utifrån operativsystemens framtidsutsikter. Windows 2000 är efterföljaren både till Windows NT4 och Windows 98. Detta torde resultera i att det blir det mest använda operativsystemet om två till tre år om man jämför med tidigare versioner som Microsoft släppt. Konkurrenten Linux är för närvarande det snabbaste växande operativsystemet och som börjat stjäla betydande marknadsandelar från Microsoft (ZDNet, 2000). Linux är till skillnad från Windows 2000 helt gratis (Linux.org, 2000). Dessutom skiljer sig operativsystemen åt då Linux är baserat på UNIX. Denna jämförelse blir intressant ur interoperabilitetssynpunkt. (Åkerman, 2000) Dessutom kan en jämförelse göras för att försöka se om prestandan av säkerhetslösningarna är mer beroende av operativsystemet än av systemet självt.

5.5 Val av säkerhetslösningar

Det främsta kriteriet för val av säkerhetslösningar är deras användningsområden. Man talar bl a om olika grader av krypteringssäkerhet såsom snabb (osäker) kommersiell, och militär (hög) (PGPi, 2000)

SSL och PGP, är de olika protokollen som jämförs via en lastmätning. SSL sätter upp en krypterad, signerad kommunikationskanal mellan två datorer för säkra transaktioner och säker överföring av data. SSL är inte riktigt lika flexibelt som PGP (Johansson, 1998). PGP krypterar, signerar och verifierar godtycklig data som sedan kan skickas iväg eller lagras lokalt för senare användning. PGP är utan tvivel den absolut äldsta säkerhetslösningen av de tre (PGPi, 2000).

Sammanfattningsvis har jag valt att jämföra de resurser såsom processtid, minne och bandbredd som alla är relaterade till produktivitet. Vidare avser jag göra en litteraturstudie av säkerheten. Jämförelsen av produktivitetsresurserna sker m h a en så kallad lastmätning där jag använder verktyget Top i operativsystemet Debian GNU/Linux och Aktivitetshanteraren i Windows 2000. Dessa mätningar och litteraturstudien avser belysa säkerhet kontra produktivitet såtillvida att man först får en överblick över de olika systemens säkerhet men även vilka resurskrav de har och hur de utnyttjar de tillgängliga resurserna. Med detta som bas förs sedan en diskussion kring resultaten. Styrkan i testen är att man får resultat som enkelt går att analysera, svagheten är att det finns brus vilket utesluter riktigt noggranna jämförelser. Vidare så beror bruset på vilka variabler man kan kontrollera och hur goda uppskattningar man kan göra av dessa. Lastmätning är även en föredömligt enkel metod i förhållande till hur precisa resultat den rent teoretiskt skulle kunna ge (Hockney, 1995).

6 Genomförande av lastmätningen

Det är i detta skede som de olika säkerhetslösningarna testas. Jag börjar med att beskriva lastmätningsresultaten utifrån säkerhetslösningarna PGP, SSL.

De resurser jag mätt i Debian GNU/Linux är storlek (använt minne primär och sekundär), samt processtid där jag valt att redovisa medelvärdet av den använda processtiden dvs tiden mätt i relation till hur mycket process tid som totalt finns tillgängligt, detta för att försöka minska påverkan från slumpmässiga felkällor i enlighet med CGS som beskrivits tidigare i arbetet. Slutligen har jag även valt att redovisa den totala processtiden som använts då detta reflekterar hur mycket tid som verkligen gick åt till bearbetning av data i motsats till hur lång tid programmet tog att köra. Detta är av speciellt intresse om man vill betrakta flera simultana transaktioner.

6.1 PGP

PGP testades först under Debian GNU/Linux och sedan under Windows 2000. För krypteringen användes en 2048 bitars RSA nyckel.

6.1.1 Debian GNU/Linux

För dessa test användes en Laptop med en Pentium CPU på 133 MHz och 16 MB primärminne. Först gjordes ett test av PGP då programmet laddades in i minnet men då det inte utförde någon kryptering. Detta för att ge en referensmätning att jämföra med då jag senare genomförde en mätning av lasten under pågående kryptering.

Tabell1. Tabellen visar resultaten från referensmätningen

Storlek (KB)	**%CPU**	**Total tid (s)**
1080	0	0

PGP använde i det här fallet alltså 1080 KB primärminne.

6.1.1.1 Textfil

Det första krypteringstestet med PGP gick ut på att kryptera en fil med låg entropi. Som fil valdes en text i formatet RTF[11]. Filen beskriver hur gemensamma egenskaper hos Linux baserade operativsystem och storleken var 1 002 047 bytes.

En relativt stor fil valdes för att underlätta lastmätningen då man i detta fallet har en längre tidsperiod under vilken man kan mäta resursåtgången och därvid få en bättre bild av hur lastprofilen ser ut. Efter mätningarna bildades medelvärden av de uppmätta storheterna förutom minnesåtgång och total tid, detta i syfte att i största möjliga mån eliminera brus ur resultaten. För minnesåtgången valdes istället maxvärdet då detta bättre visar på vad som krävs och för den totala tiden gjordes ingen bearbetning alls.

[11] Rich Text Format

Tabell 2. Tabellen visar resultatet av mätningen

Storlek (KB)	**%CPU**	**Total tid. (s)**
1152	70	4

Mätningarna utfördes med intervall på 1s.

6.1.1.2 Bild

Det andra krypteringstestet med PGP gick ut på att kryptera en fil med hög entropi. Som fil valdes en JPEG komprimerad bild. Bilden är ett montage av planterna i solsystemet och filstorleken är 1 144 864 bytes.

Även här valdes en relativt stor fil med tanke på att förenkla mätningarna då man slipper mäta bråkdelar av sekunder.

Tabell 3. Tabellen visar resultatet av mätningen

Storlek (KB)	**%CPU**	**Total tid. (s)**
1584	80	13

Mätningarna utfördes med intervall på 1s

6.1.2 Windows 2000

För detta test användes en dator med en Intel Pentium-III CPU på 450 MHz och 64 MB primärminne. Eftersom att aktivitetshanteraren i Windows saknar möjlighet att logga mätningar, blir resultaten här något mer imprecisa. För de olika testen användes samma testdata som under Debian GNU/Linux. Även namnen på de uppmätta storheterna skiljer sig mellan Debian GNU/Linux och Windows, men jag har här valt att använda samma namn rakt igenom i presentationen av resultaten.

Notera speciellt att då mätningarna i Debian GNU/Linux och Windows 2000 gjordes på olika maskiner så är det den relativa prestandan som är intressant. De absoluta prestandasiffrorna är alltså inte direkt tillämpbara i en jämförelse utan man blir först tvungen att bearbeta datan – genom att t ex se till ration mellan processtid för en krypterad och en okrypterad transaktion.

6.1.2.1 Textfil

Krypteringen av textfilen gick till på i princip samma sätt som under Debian GNU/Linux.

Tabell 4. Tabellen visar resultaten av mätningen

Storlek (KB)	**%CPU**	**Total Tid. (s)**
1808	95	1

Resultatet stämmer väl överrens med vad jag förväntade mig, vilket är i princip samma beteende som under Debian GNU/Linux. Detta eftersom att PGP bör vara mest begränsat av det tillgängliga processkapaciteten då det använder RSA-krypteringen vilken tidigare konstaterats kräver mycket beräkningar (Johansson, 1998).

6.1.2.2 Bild

Även här såg resultaten ut som förväntat och bilden tog ca tre gånger så lång tid att behandla som texten trots dess marginellt större längd.

Tabell 5. Tabellen visar resultaten av mätningen

Storlek (KB)	**%CPU**	**Total Tid. (s)**
1840	95	3

Efter att ha genomfört lastsmätningen finner man att PGP, då det är inaktivt, använder 1080 KB minne, detta till uppstart av programmet, och ingen processtid överhuvudtaget.

Vid en jämförelse mellan att kryptera texten och bilden visade det sig att bilden i alla avseenden krävde mer resurser att kryptera. Detta ter sig fullt naturligt då PGP komprimerar datan innan den krypteras och en fil med hög entropi (bilden) är svårare att packa – texten kan alltså krympas mer innan den krypteras och man får således en väsentligt mycket mindre mängd data att kryptera i slutändan.

PGP uppvisade nästan identiska resultat oavsett vilket operativsystem det kördes under, den enda skillnaden bestod i den totala tidsåtgången för att genomföra krypteringen. Den relativa processtiden var nästan exakt tre gånger längre för bilden än för texten i båda fallen, PGP använde all tillgänglig processkapacitet och minnesåtgången var marginellt högre för bilden än för texten.

6.2 SSL

Om man jämför SSL med PGP så finner man snart att SSL har en betydligt kortare nyckelängd. SSL använder sk sessionsnycklar som kastas bort efter att man avslutar kommunikationen. SSL kan dock betraktas som en lättviktskryptering. När jag genomförde testen på SSL valde jag att testa en SSL klient – i det här fallet Netscape Navigator 4.72. Som testdata valdes en befintlig SSL krypterad websida[12]. Eftersom det tar alltför kort tid att läsa en enstaka websida, har lasten här mätts under det att en och samma sida har lästs om och om igen. För att ge ett mer rättvist resultat så kringgicks cachen[13] i Navigator då sidan lästes okrypterat. När man läser ett krypterat dokument läggs detta av säkerhetsskäl inte i cachen (Netscape, 2000), vilket är varför detta gjordes.

[12] Då det skulle innebära alltför mycket arbete med att installera och konfigurera en webserver med stöd för SSL.
[13] Ett mellanlagringsutrymme där webbsidor, bilder och annan data lagras för att man inte skall behöva hämta dem från servern var gång.

6.2.1 Debian GNU/Linux

För dessa test användes en Laptop med en Pentium CPU på 133 MHz och 16 MB primärminne. Som SSL klient användes Netscape Navigator 4.72 för ett i386 Linux system.

Som första test gjordes en lastmätning under det att en sida lästes utan kryptering och ett medelvärde av lasten togs fram. Då SSL ej komprimerar det som skickas, är det här inte meningsfullt att titta på olika typer av data**.** (SSL, 2000)

Tabell 6. Tabellen visar resultaten för en okrypterad sida

Storlek (KB)	**%CPU**	**Total tid. (s)**
9072	40	10

Tabell 7. Tabellen visar resultaten för en krypterad sida

Storlek (KB)	**%CPU**	**Total tid. (s)**
8900	35	10

6.2.2 Windows 2000

För detta test användes en dator med en Intel Pentium-III CPU på 450 MHz och 64 MB primärminne. Testen utfördes på samma sätt som under Debian GNU/Linux.

Aktivitetshanteraren i Windows har här tvingat mig att göra en något grövre mätning än i Debian GNU/Linux då kontinuerliga förlopp blir svårare att mäta utan tillgång till loggning**.**

Tabell 8. Tabellen visar läsning av okrypterad data

Storlek (KB)	**%CPU**	**Total tid. (s)**
10576	Låg aktivitet	2

Tabell 9. Tabellen visar läsning av krypterad data

Storlek (KB)	**%CPU**	**Total tid. (s)**
12148	Medelhög aktivitet	5

SSL uppvisade både fler och färre skillnader än väntat. Fler såtillvida att resultaten skiljer sig åt mellan de olika operativsystemen. Färre i det fall att det under Debian GNU/Linux ej stod att finna några signifikanta skillnader mellan att läsa krypterad och okrypterad data med SSL.

Jämför man däremot att läsa krypterad och okrypterad data under Windows 2000 så ser man direkt att det finns en väl märkbar skillnad i last. Att läsa krypterad data kräver nästan 2 MB mer minne och processtidsåtgången är, även om den fortfarande inte är direkt hög, märkbart högre.

Vad man kan sluta sig till av detta är dels att operativsystemen i detta fallet spelar en större roll för prestandan samt att mätningarna är mer bruskänsliga – vilket betyder att resursåtgången för SSL är liten.

7 Analys

7.1 Jämförelse mellan PGP och SSL

Det som kan sägas efter mätningarna är att PGP kräver mycket mer processtid än SSL. PGP använder mycket CPU-tid under korta perioder när det är aktivt i jämförelse med SSL som inte nämnvärt nyttjar processtid överhuvudtaget. Detta är väntat då krypteringsnyckeln i PGP är mycket längre än krypteringsnyckeln i SSL.

PGP påverkas inte av vilket OS som det körs under, medan SSL uppvisade resultatskillnader beroende av OS. Av detta kan slutsatsen dras att PGP som har den säkraste krypteringslösningen även kräver mest resurser.

I analysen visas en tabell över de redan etablerade säkerhetslösningarna PGP och SSL. SET, vilket är en nyare säkerhetslösning, kommer att jämföras då denna lösning är intressant eftersom att den skiljer sig ifrån de tidigare lösningarna på så sätt att det är en helhetslösning och då den anses ha en högre grad av säkerhet. De aspekter av SET som inte kommer att jämföras är resurskrav såsom minne, processtid etc då dessa inte implementerats i experimentet.

Tabell 10. Tabellen visar en sammanställning av säkerhets och produktivitetsvariabler

	PGP	**SSL**	**SET**
Flexibilitet	Mycket	Ganska	Inte alls
Nyckellängd	2048	128	1024
Krypteringar	RSA	RSA, DES	RSA, DES
Uppskattad krypteringsstryrka	Extremt hög (Paranoid)	Måttlig (Kommersiell[14])	Hög (Militär)
Bunden/Obunden	Datorobunden	Datorobunden	Datorbunden
Transparens	Mestadels manuellt	Helt automatiskt	
Användningsområde	Kryptering, signering och autenticiering och icke-förnekelse.	Kryptering och autenticiering endast för nätverks-kommunikation.	Virtuellt kontokort endast, blott och bart. Krypering, signering, autenticiering och icke-förnekelse.
Interoperabilitet	Linux, Windows 2000	Linux, Windows 2000	Windows 2000 endast
Resursanvändning	Så mycket processtid som möjligt, lite minne (<2MB).	Lite processtid, lite minne.	-
Resurskrav	Låga	Låga, kräver dock en klient[15]	Måttliga krav

[14] Detta baseras på att nycklarna används som engångsnycklar.

Flexibilitet

Detta är en uppskattning av hur lösningen kan anpassa sig efter olika behov. Med det menas på vilket sätt lösningen kan användas. Grundtanken med PGP är att användaren själv skall kunna välja hur säkerhetsproblemen skall lösas (PGPi, 2000). Detta genom att själv välja nyckellängd, krypteringsalgoritmer och nyckelhantering. I mitt experiment använde jag en enkel RSA kryptering med en nyckellängd på 2048 bitar, vilket är den vanligaste nyckellängden och algoritmen (PGPi, 2000), även om PGP stödjer flertalet olika både symmetriska och assymetriska krypteringar med varierande nyckellängd från 56 bitar för DES till maximalt 16384 bitar för RSA (PGPi, 2000).

SSL är flexiblelt i det att den teoretiskt sett kan kryptera all typ av nätverkstrafik. Dock är den största begränsningen hos SSL att den *endast* kan kryptera nätverkstrafik (SSL, 2000)**.** SSL kan inte heller utföra signering, utan endast autenticiering vilket ytterligare begränsar användningsområdena (SSL, 2000)**.**

SET kan inte överhuvudtaget ses som flexibelt då detta är en helhetslösning som endast är avsedd för kontokortshandel på Internet (SET, 2000).

Nyckellängd och uppskattad krypteringsstyrka

Endast i PGP kan nyckellängden variera. SSL har en nyckellängd på 128 bitar denna nyckellängd har tidigare ”knäckts” (distributed.net, 2000), vilket innebär att säkerheten inte är tillräcklig för känslig data som behöver förbli hemlig även efter att den har använts. SET har en längre nyckelländ på 1024 bitar. Detta räknades som militär säkerhet 1994, och omnämns fortfarande som det (PGPi, 2000). Om man dock ser till Moore:s lag vilken säger att datorernas kapacitet fördubblas och priset halveras var 18:e-24:e månad innebär detta att datorkapaciteten och verktygen blir mer avancerade och att det som för sex år sedan sågs som enormt säkert inte är lika säkert idag (Ars Technica, 2000).

Krypteringar

RSA är den krypteringsalgoritm som säkerhetslösningarna använt sig av i experimentet. Dock är det, som tidigare nämnts, så att RSA och DES är de enda algoritmer som SSL och SET använder. PGP hanterar även ett stort antal andra krypteringar såsom IDEA och CAST, men eftersom PGP är så pass flexibelt så får användaren välja vilken kryptering som PGP ska använda(PGPi, 2000).

Bunden/Obunden

SET är den enda av dessas tre säkerhetslösningar som är datorbunden (SET, 2000). Med detta menas att användaren installerar ett certifikat för var maskin som hon skall kunna använda SET ifrån. Detta gäller visserligen till viss del för PGP också, då vissa nyckeltyper hanteras som filer, men de behöver dock inte installeras på de system där de skall användas – det räcker med att de finns tillgängliga t.ex. på en diskett (PGPi, 2000). För SSL krävs ingenting alls mer än att en SSL klient existerar (SSL, 2000).

Transparens

Transparens innebär i detta fallet att man inte behöver märka av säkerhetslösningarna när man arbetar med dem. SSL är den enda av lösningarna som passerar obemärkt. (Netscape, 2000) PGP- lösningen baserar sig på flexibilitet, vilket i detta fall även innebär manuell hantering (PGPi, 2000). Det finns program som använder sig av PGP

[15] SSL är ej fristående utan kräver t ex en webläsare med SSL stöd.

i bakgrunden och man kan på grund av detta eventuellt påstå att PGP har en viss transparens, men dock inte i sitt grundutförande (PGPi, 2000).

Användningsområde

Både PGP och SET stödjer de krav som ställs på god krypteringsteknik. Samtliga lösningar stödjer kryptering och autenticiering. Dock så stöds inte verifiering och icke-förnekelse av SSL (SSL, 2000). I SET hanteras detta m h a certifikat och signering (SET, 2000). I PGP hanteras det främst av signering (PGPi, 2000).

Interoperabilitet

Med detta menas hur väl lösningen kan anpassas till andra lösningar och andra system. I min jämförelse uppvisar SET störst brister då detta säkerhetsprogram inte kan köras på andra plattformar än Windows (SET, 2000). Detta är inte särskilt produktivt om exempelvis användaren endast har tillgång till Unix-miljö. SSL är väl spritt och finns till de flesta plattformar med någon typ av web-läsare (Lynx, 2000). PGP finns till 11 olika platformar och kräver nästan ingen konfigureration av systemet (PGPi, 2000).

Resursanvändning

Här valdes att inte jämföra med SET eftersom att ingen lastmätning gjordes på det då SET ej finns till Debian GNU/Linux. Det som kan nämnas om PGP och SSL är att PGP använder mer resurser än SSL om det finns tillgängligt, enligt min mätning. Detta är inte svårt att förstå om man ser till nyckellängderna och krypteringsalgoritmerna. En god approximation torde vara att SET befinner sig någonstans emellan PGP och SSL vad det gäller processtid eftersom att den använda nyckellängden gör det. När det gäller minnesåtgången förbrukar troligen SET mer än både SSL och PGP då programmet har ett mer omfattande användargränssnitt.

Resurskrav

De krav som behövs för att lösningen ska kunna användas skiljer sig mellan det olika säkerhetslösningarna. Detta har med interoperabilitet och flexibilitet att göra. PGP har som tidigare nämnts stora möjligheter till att kunna användas med få resurser (PGPi, 2000). SSL kräver inte heller särskilt mycket när det kommer till de olika kvantitativa resurser som tidigare diskuterats i detta arbete (SSL, 2000). Dock kräver SSL en klient som kan behöva mer eller mindre resurser – vanligtvis mer än PGP. SET har visserligen inte särskilt höga krav men däremot många om man jämför med PGP och/eller SSL (SET, 2000).

8 Litteraturstudie av bandbredd

En prestandaaspekt som kan vara av intresse för applikationer i nätverk är hur mycket bandbredd applikationen använder (Silberschatz och Galvin 1994). Med bandbredd menas här överföringskapacitet på ett nätverk, lämpligen mätt i bitar per sekund. När det kommer till kryptering och bandbreddsbehov, är detta speciellt intressant eftersom att krypterad data inte kan komprimeras (Sjögren, 1996). För fasta nätverksanslutningar är detta vanligtvis inte så intressant eftersom att de sällan komprimerar data, dock finns det åtminstone en vanlig typ av nätverksanslutning som gör detta - nämligen modem (Halshall, 1993)**.** Visserligen är det så att lasten är mest kritisk hos servrar, där det inte finns någon anledning att inte ha en fast nätverksanslutning, men att lasten är mest kritisk för servrar betyder ju inte att bandbreddsbelasningen är helt ointressant för användare - speciellt inte med tanke på att alla moderna datorer hanterar *multitasking* - att utföra flera olika uppgifter simultant (Silberschatz och Galvin, 1993)**.** En användare kan ju vilja göra andra saker samtidigt med den krypteradenätverkstrafik jag här tar upp.

8.1 PGP

PGP komprimerar all data innan den krypteras, vilket har tre principiella fördelar (Sjögren, 1996). Den första fördelen är att komprimering av data inte bara gör den mindre, den ökar även entropin hos datamängden vilket är en fördel ur krypteringssynpunkt. Den andra fördelen som Sjögren (1996) också nämner är att den datamängd som ska krypteras blir mindre. Detta kan vara av intresse om man använder krävande krypteringsalgoritmer med långa nycklar - någonting som PGP utan tvekan gör (PGPi, 2000)**.** Slutligen blir den krypterade datamängden också mindre- vilket för lagring innebär mindre sekundärminnesbelasning, och för kommunikation mindre bandbreddsbehov. Detta betyder att om man har en icke-komprimerande nätverksansluting är det alltså produktivare, ur bandbreddssynpunkt, att kryptera datamängden med PGP än att skicka den i klartext. Har man däremot en komprimerande nätverksansluting blir produktiviteten i värsta fall marginellt sämre men aldrig mycket sämre än jämfört med att skicka klartext (PGPi, 2000). Även i fallet med en komprimerande nätverksanslutning kan man eventuellt vinna i prestanda även om det knappast blir märkbart mycket. Vinsten i det fallet beror på vilken komprimeringsalgoritm som råkar vara bäst för den mängd data man vill komprimera (Comp.compression FAQ, 2000).

8.2 SSL

SSL komprimerar inte den data som skall skickas utan krypterar den endast (SSL, 2000). Detta kräver visserligen mindre processtid än att först komprimera och sedan kryptera, men den krypterade data kommer alltid att ta upp lika mycket utrymme, ändå (Sjögren, 1996). För icke-komprimerande nätverksanslutningar betyder detta att produktiviteten är exakt densamma som för att skicka datamängden som klartext. För komprimerande nätverksanslutningar betyder det dock att man aldrig kan uppnå en högre produktivitet än för en icke-komprimerande nätverksanslutning med samma kapacitet (PGPi, 2000). Då SSL är avsett för krypterad direktkommunikation, t ex när man läser en websida, är det dessutom så att den data som skickas vanligtvis har relativt låg entropi och därför är lätt att komprimera (Sjögren, 1996). Detta i motsats till om man med t ex ftp hämtar redan komprimerad data som har hög entropi vilket gör att krypteringen inte spelar någon roll för bandbreddsbehovet (PGPi, 2000). Kontentan av detta är alltså att SSL faktiskt ökar bandbreddsbehovet - kanske inte för

alla användningsområden, men väl för den absolut vanligaste, vilket tveklöst är att kryptera http-trafik (SSL, 2000).

8.3 Jämförelse mellan PGP och SSL

Kryptering förhindrar att data komprimeras, vilket kan leda till större bandbreddsbehov. PGP kringgår detta genom att själv komprimera mängden data innan den krypteras – dock fungerar denna strategi sämre på data som redan har hög entropi (Sjögren, 1996). För data med hög entropi kommer PGP att kräva mer processtid än en icke-komprimerande lösning med samma kryptering. Vad gäller SSL så komprimeras mängden data överhuvudtaget inte, vilket resulterar i att man inte kan minska bandbreddsbehovet med komprimeringsstrategier på nätverksnivå (Sjögren, 1996). Om man ser till den typen av data som vanligen skickas, nämligen text, framgår det snart att PGP är överlägsen eftersom text lämpar sig väl för komprimering. I de fall man vill kryptera data med hög entropi kräver PGP mer processtid, men för bandbreddsbehovet så spelar detta inte någon roll överhuvudtaget (PGPi, 2000).

9 Resultat

Frågan om hur säkerhet och produktivitet kombineras inom området datasäkerhet torde vara besvarad i och med jämförelsen mellan de tre säkerhetslösningarna PGP, SSL och SET i analysdelen samt med den kompletterande diskussionen rörande bandbredd.

Det som finns att tillägga för att återigen sammanfatta resultaten är följande. PGP med 2048 bitars kryptering är som fristående program den säkerhetslösning som kräver mest resurser, i alla fall vad gäller processtid. Lastmätningen visar på att PGP oavsett operativsystem använder ganska lite minne men så gott som all tillgänglig processtid. SSL använder relativt lite av båda, men kräver dessutom en klientapplikation som kan ha varierande krav. PGP är flexibelt, kraftfullt och säkert, men mer krävande att använda i motsats till SSL som är nästan helt transparent. Bandbreddsutnyttjandet är, för viss data effektivare för PGP då denna i motsats till SSL komprimerar data. För okomprimerbar data är det inte någon skillnad.

Verkar högre teknisksäkerhet leda till att systemet blir svårare att använda d v s leder säkerhetsaspekten till att användaren får fler rutiner i sitt datorutövande som leder till att systemet blir svåranvänt?

Resultatet visar att PGP som potentiellt är den säkraste krypteringslösningen är också den mest komplexa. Säkerheten hos PGP kommer inte endast ifrån att den kan hantera långa nycklar, utan även från det faktum att man kan kombinera de olika typerna av krypteringar. Dessutom går det att använda olika nycklar/krypteringsalgoritmer där de passar bäst. Denna flexibilitet i användandet av PGP kan leda till produktivitet då man oftast kan göra det man avser med PGP - om man besitter tillräckligt goda kunskaper. För den ovane användaren är PGP inte en lätthanterlig lösning – därav komplexiteten.

SSL är mer inriktat på produktivitet än på säkerhet såtillvida att SSL både är oerhört transparent och dessutom direkt anpassat för nätverkskommunikation. Dock har SSL betydligt lägre säkerhetsnivå än PGP. SSL hanterar transparent krypterad kommunikation på ett sätt som inte stör användaren i sina uppgifter, men detta delvis på bekostnad av säkerhet.

SET, som inte är särskilt transparent, men trots detta tämligen lättanvänt, har hög säkerhet. Dock medger SET inte att det gör någonting annat än just det som det är avsett för – nämligen att utföra elektroniska kontokorttransaktioner. Detta visar på att transparens, som är en väsentlig produktivitetsfaktor, påverkar säkerheten. Den största bristen hos SET är utan tvivel att certifikaten måste installeras på de maskiner som SET skall användas på. Om certifikatinstallation och avinstallation är alltför tidskrävande kommer detta att leda till att det helt enkelt ses som produktivare att låta certifikaten vara installerade på alla maskiner man kan vilja använda SET på. Att låta certifikaten vara installerade på flera maskiner kommer givetvis drastiskt att minska säkerheten och här har vi återigen ett exempel på motsättningen mellan produktivitet och säkerhet.

Svaret på frågan, huruvida högre säkerhet leder till att systemet blir svårare att använda, torde vara: Ja, ökad säkerhet leder till ett mer svåranvänt system om man ser till de aspekter som tas upp i detta arbete i dagsläget. Dock finns andra faktorer som inverkar, vilket kan ses på SET som i och för sig är säkert, men är begränsat till en enda uppgift.

10 Diskussion

Även om det omnämnts på otaliga ställen i arbetet att kryptering, i sig, inte räcker för att upprätthålla en fullgod säkerhet så tål detta att upprepas ännu en gång. Kryptering är en viktig och vital del av elektronisk säkerhet, men det finns fortfarande andra aspekter att ta hänsyn till som är minst lika viktiga. Detta har Bruce Schneier illustrerat med följande liknelse:

"Using PGP with an extremely strong key is akin to protecting your house with a fence composed of a single picket a mile high. No one will get through that picket - but they can just walk around it." (Ars Technica, 2000)

Det Schneier menar med detta är att det är alldeles för lätt att ha en övertro till kryptering och teknik i allmänhet. Konsekvenserna av detta är att användaren inte ser sambandet med säkerhet och policy, att bristande säkerhetsrutiner gör hela systemet osäkert. Nu vill jag inte påstå att den lastmätning som gjorts skulle vara i onödan, för den har dels styrkt resonemanget kring produktivitet och säkerhet samt visat på potentiellt intressant prestandaskillnader, utan endast att användningen av ett verktyg, som t ex PGP, är minst lika viktig som dess säkerhet. Säkerhetsrutiner som innebär att systemet blir mer svårhanterligt kan bli en börda för användarna. Om en organisation övergarderar sig (Freese och Holmberg 1993) tas onödigt mycker resurser i anspråk – vilket även gäller användarna av systemet som indirekt uppmuntras till att kringgå säkerhetsrutinerna för att uppnå en högre produktivitet i sitt arbete. En lägre säkerhetsnivå kan i dylika fall på sikt vara säkrare då den lättare kan motiveras till att efterföljas, eller, som Freese och Holmberg (1993) skriver:

"Ganska många av de rutiner som införs i dagens samhälle för att säkra information eller andra värden är alltför inflexibla. Vilket skapar en negativ inställning till säkerhet som sådan." (s.63)

Det är viktigt att när man väljer en säkerhetslöning, först se till att den dels löser de problem man avser lösa, dels att alla tekniska krav är uppfyllda – både vad gäller säkerhet och produktivitet. Dessa mer tekniska aspekter, såsom krypteringsalgoritm, bandbreddsbehov, minnesanvändning, processtidsanvändning o dyl är alltså nog så viktiga och bör dessutom komma på ett tidigare stadium än användandet. Att fokusera på användaren utan att först se till detta leder till att man i värsta fall kan studera användadet av en lösning till fel problem eller användandet av en otillräcklig lösning. Bara genom att se till den gjorda lastmätningen framgår dessutom att det inte bara är lösningen som sådan som inverkar, utan även vilket operativsystem och vad för hårdvara som används.

Som nämnts i avsnittet om lastmätning, är den mätning jag gjort ganska approximativ till sin natur, och en mer grundlig undersöking av olika krypteringslösningars kvantitativa egenskaper kan vara av intresse. Detta kräver dock omfattande och djupa kunskaper i datalogi, programmering och komplexitetsanalys (Rosen, 1995). Lastmätning är en metod som ganska väl täcker in de behov jag hade för att genomföra min analys av de olika säkerhetslösningarns resursbehov. Precisionen i resultaten är ganska låg, men stödjer ändå min analys på ett tillförlitligt sätt. Metoden får tas för vad den är – en kvantitativ uppskattning av säkerhetslösningarnas resursförbrukning.

Andra förslag till arbeten som kan vidareutveckla de frågor och undersökningar som gjorts i detta arbete kan vara:

Att genomföra en mer heltäckande eller kompleterande undersökning där användarnas roll i säkerheten tas i beaktande då säkerhet är mer än kvantitativa aspekter.

Att göra vidare studier av de tekniska aspekterna hos säkerhet och produktivitet, speciellt i relation till egenskaper som flexibilitet då vi hos SET har sett att hög säkerhet kan kompenseras med specialisering för att på så sätt erhålla en relativt låg komplexitet.

11 Slutsatser

När det gäller säkerhet handlar det inte bara om kryptering och verifiering utan ett säkert system måste se till helheten. Detta är en konsekvens av många olika standarder där olika protokoll och certifikat inte kan läsas av varandra. Här följer huvudpunkterna av det som skrivits i rapporten.

- Produktivitet och säkerhet står i motsatsförhållande.
- Snabbhet och transparens leder till lägre säkerhet.
- Hög säkerhet leder till komplexitet och högre tidsåtgång.
- Det är viktigt att välja säkerhetslösning efter behov.
- Komplexitet hos säkerhetslösningar kan kompenseras t ex genom att begränsa funktionaliteten.
- För att täcka upp alla säkerhetsbehov kan man behöva flera säkerhetslösningar vilket i sin tur kan leda till interoperabilitetsproblem.
- Ökad förståelse för tekniken bakom säkerhetssystem kan ge ett bättre underlag för användningsstudier.
- Sambandet mellan säkerhet och produktivitet tål att undersökas närmare, speciellt i relation till andra faktorer som t ex flexibilitet.

Tekniken stödjer användarna och möjliggör säkerhet, men den sätter även vissa käppar i hjulet för dem då säkra lösningar tenderar till komplexitet. Detta i sin tur leder till att användarna vill ha enklare lösningar, och då är alternativen att sänka säkerheten eller begränsa användningsområdena för säkerhetssystemet. Ett alltför begränsat säkerhetssystem begränsar dock även användaren vilket kan leda till en negativ syn på säkerhet.

Referenser

Beizer, B. (1990) *Software testing techniques*. New York: International Thomson Publishing Inc.

Brorsson, M. (1999) *Datorsystem Program- och maskinvara*. Lund: Studentlitteratur.

Carlsson, P. Johansson, S. (1997) *Modern elektronisk mätteknik.* Stockholm: Liber.

Comp.compression FAQ. (2000) *The comp.compression FAQ*. [online]. Available From: comp.compression [Datum: 2000-05-17]

De Gelas, J. (2000) *The Athlon: Seventh generation or not?* [online]. Aces Hardware. Available from: http://www.aceshardware.com [Datum: 2000-05-17].

De Gelas, J. (2000) *The Secrets of High Performance CPUs, Part 1-5.* [online]. Aces Hardware. Available from: http://www.aceshardware.com [Datum: 2000-05-17].

Elmasri, R. Navathe, S. (1994) *Fundamentals of database systems, 2nd edition*. Redwood City: Addison-Wesley.

Ericsson, H. (1998) *Elektronisk handel krymper avstånd.* SAF Tidningen, Nr 4, sid 17-19.

FreeBSD. (2000) *Learning about UNIX*. [online]
Available from: http://www.freebsd.org/projects/newbies.html [Datum: 2000-05-17].

Freese, J., Holmberg, S. (1993) *Data osäkerhet* 3e upplagan. Stockholm: Affärsinformation AB.

FSF. (2000) *The Free Software Foundation*. [online]. Available From: http://www.fsf.org [Datum: 2000-05-17]

Halsall, F. (1993) *Data communications, computer networks and open systems.* New York: Addison-Wesley.

Hockney, R. (1995) W. *The Science of Computer Benchmarking* London: SIAM Society for Industrial & Applied Mathematics

Johansson, R. (1998) *Kryptering nyckeln till säkerhet.* Stockholm: Svenska attachéer utlandsrapporter.

LeFebvre, W. (2000) *The Top Homepage*. [online] Available From: http://www.groupsys.com/topinfo [Datum: 2000-05-17].

Lund, L.G., Montgomery, H. Waern, Y. (1992) *Kognitiv psykologi.* Lund: Studentlitteratur.

Lynx (2000) Lynx. [online]. Available From: http://lynx.browser.org [Datum: 2000-05-17]

Microsoft (2000) Microsoft Corp. [online]. Available From: http://www.microsoft.com [Datum: 2000-05-17]

Netscape (2000) *Introduction to Public Key Cryptography*. [online]. Netscape. Available From:
http://developer.netscape.com/docs/manuals/security/pkin/contents.htm
[Datum: 2000-05-17]

OpenBSD (2000) *Security*. [online]. Available From:
http://www.openbsd.org/security [Datum: 2000-05-17]

PGP International (2000) PGPi. [online]. Available From: http://www.PGPi.org [Datum: 2000-05-17]

Patel, R., Davidsson, B. (1994) *Forskningsmetodikens grunder. Att planera, genomföra och rapportera en undersökning*. Lund:Studentlitteratur

Pohl, K. (1996) *Requirements Engineering: An Overview*, CREWS Report Series CREWS-96-02

Preece, J. (1994) *Human-Computer Interaction.* New York: Addison-Wesly publishing.

Rijk, C. (2000) *Inside Tech: The UltraSPARC III*. [online]. Aces Hardware. Available from: http://www.aceshardware.com [Datum: 2000-05-17].

Rosen, K. H. (1995) *Discrete mathematics and its applications, 3rd edition*. Singapore: McGraw-Hill international editions.

Ryu, W. (2000) *Pickt Fence Privacy*. [online]. Ars Technica. Available From: http://www.arstechnica.com [Datum: 2000-05-17]

The SET Standard Book 1 Business Description
http://www.setco.org/download.html/#spec

Schneier, B. (1995) *Applied cryptography.* London: John Wiley and Sons Ltd

Silberschatz, A., Galvin, P. (1994) *Operating system concepts*, 4th edition. New York: Addison-Wesley.

SIS. (1999) *Svensk Standard SS 62 77 99 utgåva 1*

Sjögren, J. (1996) Talteori och Kryptografi. Skövde: Högskolan I Skövde

Solar. (2000) *Open Source == Faster Bugfixes*
http://slashdot.org/articles/00/01/17/1355245.shtml

SSL. (2000) *The SSL Standard*. [online] Available From: http://www.ssl.com [Datum: 2000-06-01]

Stokes, J. (2000) *Apple and PPC*. [online]. Ars Technica. Available From: http://www.arstechnica.com [Datum: 2000-05-17]

Stokes, J. (2000) *The Future of RAM*. [online]. Ars Technica. Available From: http://www.arstechnica.com [Datum: 2000-05-17]

Stokes, J. (2000) *The G4 and the K7: an architectural look at two post-RISC processors*. [online]. Ars Technica. Available From: http://www.arstechnica.com [Datum: 2000-05-17]

Torvalds, L. (2000) *linux-2.2.14.tar.bz2*. [online]. Available From: http://www.kernel.org/pub/linux/kernel/v2.2/linux-2.2.14.tar.bz2 [Datum: 2000-06-01]

ZDNet. (2000) *Java Linux Duopoly*. [online] Available From: http://www.zdnet.com/enterprise/stories/main/0,10228,2581961,00.html [Datum: 2000-05-17]

Åkerman, C. (2000) PKI: Misspassning bromsar e-handeln. *Datorteknik 3.0*, Nr. 5, sid. 12-13.

Samtal

Bengtsson, P. Projektplanerare/Programmerare. ISG-Systems AB. Höganäs. 2000-05-01

Bäckström, K. Redigerare. Computer Sweden. Stockholm. 2000-09-04

Lindberg, S. Säkerhetskonsult. WM Data. Stockholm. 2000-05-17.

Ohlsson, F. Säkerhetskonsult. WM Data. Stockholm. 2000-05-17.

Thylander, M. Funktionsansvarig Win/Nt drift. SJ Data. Stockholm. 2000-05-05.

Bilaga 1. Utmärkande egenskaper som är typiska för Unix OS, samt Windows OS.

Avsnittet om Unix OS baserar sig på FreeBSD (2000).

Debian GNU/Linux:

Debian GNU/Linux är ett operativsystem baserat på Linux. Med detta menas att Debian GNU/Linux baseras på Linuxkärnan, men många av systemverktygen kommer ifrån ett projekt som heter GNU (vilket står för GNU is Not Unix) (FSF, 2000).

Debian är, liksom samtliga övriga Linuxbaserade OS, ett UNIX liknande operativsystem. UNIX är ett OS - eller snarare en familj med OS - med rötter som går långt tillbaka i tiden. Utmärkande för UNIX är följande (FreeBSD, 2000):

- **Filsystemet som gränssnitt mot enheter.**

Detta syftar till att alla UNIX OS baserar nästan all kommunikation med olika enheter på filer. En fil i UNIX är mer än bara en samling data med ett namn - en fil kan t ex representera en hårddisk, grafikminnet, en serieport eller i princip vad som helst.

- **Ett globalt filträd.**

I UNIX system så finns det exakt ett filträd som börjar med katalogen /. Många andra operativsystem (t ex WindowsNT) gör olika filträd för olika partitioner[16], hårddiskar och medier.

- **Kommandoskalet som primärt gränssnitt mot användaren.**

Oavsett vilken variant av UNIX man använder eller vad för skal man senare installerar, så har det textbaserade kommandoskalet alltid varit det gränssnitt som operativsystemet först presenterat för användaren/administratören.

- **Läslig konfigurationsdata.**

Det har varit mer eller mindre tradition i UNIX-världen att konfigurationsfiler skall vara läslig text så att de kan förstås och ändras av en insatt användare.

- **Nätverks och fleranvändarstöd**.

UNIX är gjort för flera samtidiga användare och för att sitta i ett nätverk. I den allra enklaste formen skulle detta kunna vara en server med ett antal terminaler direkt kopplade mot sig.

- **Källkodskompatibilitet.**

En styrka hos UNIX OS är att ett program skrivet för en variant oftast går att kompilera med inga eller smärre modifikationer under en annan variant.

Windows

Windows 2000 och Windows NT är olika versioner av samma operativsystem. Windows och Debian GNU/Linux ha olika tankesätt bakom designen. I alla Linuxbaserade OS är operativsystemet väl avskiljt från alla användargränssnitt (Torvalds, 2000), utvecklingen är distribuerad och allt är tillgängligt för alla som vill se hur komponenterna är konstruerade. I Windows har Microsoft integrerat

[16] Del av hårddisk

användargränssnittet i operativsystemet, all utveckling är centraliserad och hemlig (Microsoft, 2000).

Även ur säkerhetssynpunkt är de två operativsystemen olika. UNIX operativsystem har ofta kritiserats för att vara osäkra. (OpenBSD, 2000) Den troligaste orsaken till detta är att många UNIX OS normalt är konfigurerade så att de automatiskt startar många tjänster som kan innehålla säkerhetsluckor. (OpenBSD, 2000)

Windows NT är i motsats till Debian GNU/Linux säkerhetscertificerat, men certificeringen gäller endast om maskinen i fråga inte är ansluten till något nätverk (Microsoft, 2000).

Bilaga 2. Intervju med Lindberg och Ohlsson. WM-data

.

Lindberg och Ohlsson arbetar säkerhetskonsulter på WM-data i Stockholm.

Fråga:

Vad innebär det att jobba med datasäkerhet?

Lindberg och Ohlsson:

Då marknaden idag skriker efter säkerhetslösningar, mycket på grund av att det har blivit ett modebegrepp har vi mycket att stå i. och när en kund kontaktar oss vet de ofta inte vad de vill ha. De vill bara ha säkerhet och tror att det räcker med lite kryptering av mail så är allt frid och fröjd. Vad vi sysslar med behandlar så mycket mer än så.

Fråga:

Det är alltså en stor efterfrågan på tekniska säkerhetslösningar?

Lindberg och Ohlsson:

Ja, efterfrågan är stor eftersom säkerhetskraven stärks eftersom vi mer och mer lagrar och transporterar känslig information elektroniskt. Datasäkerhet idag fokuserar mycket på tekniska lösningar men som jag precis påpekade så bör man se till att man har en genomtänkt säkerhetspolicy innan man börjar ta fram ett tekniskt säkerhetssystem.

Fråga:

Vad är det som är viktigt med att först ha en säkerhetspolicy?

Lindberg och Ohlsson:

Många tror att det räcker med att ha ett krypteringssystem för t ex personalens mail så är allt bra. Men det handlar först och främst om att skydda sig från det största hotet, nämligen den mänskliga faktorn. Att skydda sig från att radera fel fil eller att skicka ett mail med känslig information till fel mottagare. För att motverka att saker som dessa inträffar kan det hjälpa med att ha säkerhetsföreskrifter om hur vissa saker och rutiner ska gå till. Det är i vissa fall inte ens nödvändigt att ta fram tekniska lösningar. Ibland räcker det med att ta fram riktlinjer för hantering av information ska gå till.

Fråga:

Om det nu är så viktigt med en säkerhetspolicy. Hur vet man då att man har en heltäckande sådan?

Lindberg och Ohlsson:

Vi använder oss av en fastställd standard som heter Svensk Standard SS 62 77 99-1 och ges ut av SIS. Det är ett sammanställt regelverk som ska täcka allt inom en organisation. Den är till stor hjälp att ha som underlag och stöd när man tar fram en policy för en organisation. Man minskar då risken att glömma någon rutin eller liknande.

Fråga:

Kärnan i mitt examensarbete handlar om säkerhet kontra prestanda. Är detta något som ni måste ta hänsyn till i era lösningar?

Lindberg och Ohlsson:

Alla vill ju ha högsta säkerhet med maximal prestanda, men det är inte alltid dessa går hand i hand, tyvärr så är det så att ofta får prestandan ge vika för säkerheten till viss grad. Allt beror ju på vad det handlar om för system. Att det tar tre sekunder att skicka ett mail har ju inte lika stor betydelse som om det tar tre sekunder att skicka en börstransaktion då varje sekund kan betyda stora pengar än om jag får iväg mitt mail tre sekunder efter det att jag tryckt på sändknappen.

Fråga:

Vad är det som gör att prestandan måste ge vika för säkerheten?

Lindberg och Ohlsson:

I vårt fall rör det sig oftast om styrkan på krypteringen. Desto kraftigare kryptering desto mer processor kraft behövs för att hålla samma prestanda som systemet hade utan kryptering. Detta kan innebära dyra investeringar för ett företag. Ta t ex ett företag med 2000 anställda som beslutar sig för att införa ett säkert faktureringssytem. Om man då vill ha stark kryptering kan detta innebära att företaget måste uppgradera 80% av sin maskinpark. Det blir dyrt inte bara i hårdvara utan nya maskiner innebär att nya intallationer måste införas gammal information ska sparas undan eller lagras om på nytt. Frågor som berör vad gör vi med all gammaldata vad kan vi slänga vad kan vi behållla. Det kan leda till otroligt mycket mera jobb att bara byta maskiner att den kostnaden blir lika hög som det nya systemet.

Om den gör det måste man välja en mindre säker lösning som går fortare, dvs svagare krypteringsalgoritmer som inte tar så mycket processortid att genomföra. Man vill givetvis ha en så säker lösning som möjligt. Detta innebär att det som efterfrågas är en så säker lösning som möjligt vilken ändå klarar av de krav man har på prestanda och pris – speciellt är produktivitet, i detta fall utnyttjande av systemresurser såsom minne, processtid och bandbredd, vitalt då en mindre produktiv lösning ger ett högre pris för samma prestanda.

Bilaga 3. Intervju med Thylander. SJ Data

Thylander arbetar som funktionsansvarig Win/Nt drift. SJ Data Stockholm. Tillverkar även Web-sidor.

Thylander kontaktade jag via E-post. Jag ställde endast en fråga men fick ett utförligt svar.

Fråga:

Du arbetar även med säkerhetsfrågor. Det talas om en avgränsning mellan säkerhet och effektivitet/produktivitet.Vilka problem har du stött på på din arbetsplats?

Thylander:

Det en användare uppfattar som effektivitet har ju hittils ständigt stått i motsats till allt vad säkerhetstänkande. Att t.ex. kunna surfa fritt, köra ICQ, IRC, RealAudio, kolla sin externa mailbox på passagen eller dylikt o.s.v. är ju alla exempel på aktiviteter många användare önskar kunna på utföra när han/hon befinner sig på sin arbetsplats. Att en användare ser dessa som effektiva tjänster gör dem icke desto mindre till stora möjliga säkerhetsluckor in till arbetsplatsen ifråga. Även andra tjänster som mer direkt kan ses som arbetsrelaterade har oftast tidigare helt ratats av it-personal på grund av de ofta extrema säkerhetsluckor de öppnar. Exempel på sådana tjänster kan vara att kunna läsa sin mail via "nätet" om man är ute och reser, att kunna komma åt sina hemliga arbetsdokument hemifrån eller från "nätet", i stort sett all form av uppringt hemarbete (då s.k. RAS-servrar sällan uppfyller de högtflygande säkerhetsmål många säkerhetschefer sätter upp). Liknande tjänster uppskattas av många användare och de anser ofta att de höjer effektivitetsgraden på arbetsplatsen.

Det är väl denna klassiska motsättning man på allt mer innovativa sätt försöker komma runt men ändå verkar användarens mål ständigt stå i konflikt med säkerheten på företaget. Ta t.ex. alla internet-banker som för tillfället existerar. Den klart "effektivaste" metoden att komma åt sin bank ur en användares synvinkel är förmodligen att han helt enkelt tvingas logga in med ett användarnamn och lösenord (som helst inte skall vara alltför svår-ihågkomligt) och vips så skall han kunna utföra sina tjänster. Problemet är dock som vanligt att detta också är ett väldigt osäkert sätt att hantera internettrafik, sniffningsattacker skulle då lätt kunna fånga upp anändares användarnamn och lösenord och sedemera tömma deras konton.

En variant som kan anses hyggligt effektiv ur en användares synvinkel är hanteringen av personliga certifikat. Användaren får ett personligt certifikat som han kan installera på den/de maskiner han önskar och sedan direkt komma åt sina bankaffärer. Problemet säkerhetsmässigt ligger där bland annat i hanteringen av certifikatet. Tar användaren verkligen bort det efter sig och ser till att ingen annan kan använda det på t.ex. ett kontor? Kommer den intet ont anande användaren att installera certifikatet på en maskin i skolan som kanske används av många andra för att sedan glömma bort att ta bort det efter sig? Återigen den klassiska konflikten effektivitet (ur en användares synvinkel) kontra säkerhet. Andra banker har börjat använda mer fysiska tillbehör som kortläsare varpå säkerheten möjligen höjs men samtidigt kan ju användaren "bara" sitta hemma med internetbanken och inte på jobbet (om han inte beställer hem 2 kortläsare, inte särskilt effektivt tycker den sure användaren)

Vissa företag använder en typ av "räknedosa" med vars hjälp man får fram koder för varje transaktion änd lite av en kompromiss mellan effektivitet och säkerhet då

användaren visserligen måste knappa fram lite koder då och då samtidigt som dosan alltid skall med.

Hur man än ser på det ställs man som säkerhetsansvarig ofta inför problemet vilken servicegrad man vill erbjuda kontra vilken säkerhetsnivå man vill lägga sig på och oftast tvingas man till en kompromiss. Man offrar en del säkerhet för att erbjuda en effektivare tjänst för kunder.

I en distribuerad miljö, med e-handel inte har samma möjlighet till att utbilda eller få användaren att följa säkerhetsrutiner och protokoll på samma sätt som en vanlig arbetsplats. Dessutom är det många websidor som crackas. Många små företag har bristande nyckellängder på sina krypteringar och detta innebär att olika typer av information lämnats ut eller att det varit möjligt att ändra på informationen så att man bara betalar tio kronor för en produkt som kostar tiotusen kronor.

En nitisk säkerhetschefs dröm är allt som oftast ett företag helt utan kontakt med omvärlden, men vad ska då företaget pyssla med idag?

Bilaga 4. Intervju med Bengtsson. ISG-Systems AB

Bengtsson arbetar som Projektplanerare/Programmerare. ISG-Systems AB, Höganäs.

Fråga:

Jag skriver ett arbete som behandlar effektivitet och datasäkerhet. Du arbetar ju med säkerhetoch har god datakännedom. Vad innebär begreppen effektivitet och datasäkerhet för dig?

Bengtsson:

Effektivitet avser oftast att utföra en uppgift med så få resurser som möjligt. Resurser i det här fallet avser oftast processtid, primärminne och, för nätverksorienterade uppgifter, bandbredd. I vissa sammanhang fokuseras uteslutande på tidsåtgången, men detta ser jag mer som en prestandafråga. Vill man se lite närmare till olika faktorer som inverkar på effektiviteten, kan man granska de olika komponenter som inverkar. Det som inverkar kan t ex vara vilken typ av CPU som används, vad för kringhårdvara som finns tillgänglig, vad för typ av buss eller crossbar som används, vad för typ av minnen som finns tillgängliga och hur de sitter - detta kallas interleaving, vilka inställningar som gjorts i OS:et, vad för processer - program - som körs samtidigt osv. Som du kanske misstänkte kan prestanda och effektivitets mätning bli ett tämligen klurigt problem om man vill ha precisa resultat.

Datasäkerhet - ja, då tänker jag nog först och främst på begrepp som feltolerans, återställning och transaktioner. Men jag gör antagandet att du syftar på kryptering, verifiering, signering och inbrottsskydd. Datasäkerhet ur det perspektivet kan ses antingen som att bygga säkra system som det är svårt att bryta sig in i, eller som att skydda data så att det inte spelar någon roll vem som ser den eftersom att endast de den är avsedd för kan förstå den. Detta görs väl nästan uteslutande med kryptering. Fast man kan ju förstås aldrig garantera att de som får läsa datamängden förstår den, så ordet "förstå" kanske inte var helt väl valt.

Fråga:

När du nämner mätning - på vilket sätt kan man mäta effektivitet?

Bengtsson:

Som jag nyss nämnde så kan det bli lätt lurigt om man vill ha riktigt precisa resultat. Dock går det att göra tämligen enkelt om man kan nöja sig med en fingervisning. De tre resurser som enklast låter sig mätas och uppskattas är utan tvekan bandbredd, minne och processtid. Allihop är kvantitativt mätbara, har hyfsad repeterbarhet och är enkla att förstå. Det är inte alltid detta räcker till, som t ex om man vill genomföra en worst case execution time analys av en komponent i ett hårt realtidssystem - men om man bara vill jämföra olika program, algoritmer eller maskiner så räcker det oftast gott och väl med en enklare lastmätning eller benchmarking.

Fråga:

Du säger att resultaten räcker gott och väl. Det innebär att resultaten inte till fullo är korrekta?

Bengtsson:

Det stämmer och detta beror på att vid olika typer av mätningar blir man tvungen att ta hänsyn till olika faktorer. Just i detta fall handlar det om att det antingen inte går eller kräver orimligt mycker tid och kunnande för att isolera de olika faktorer som

inverkar. Exakta effektivitetsmätningar är helt enkelt ett mycket svårt problem i verkligheten. Man blir nästan alltid tvungen att begränsa sig till enklare modeller om man inte redan utgår ifrån någonting väldigt enkelt som t ex en turingmaskin eller kanske någonting med lambda-kalkyl som bas. Exempelvis är det så att processtidsprofilen för ett program kan variera ganska kraftigt under dess körning - och då körs det en då på en och samma maskin hela tiden. Val av operativsystem är också avgörande, speciellt kan det försvåra mätningarna om man inte vet vad OS:et gör. Det är faktiskt bara bandbredden som inte påverkas av operativsystemet, men detta beror på att protokollet, kabeln och nätverkstrafiken som styr detta inte är delar av datorn som OS:et hanterar.