NTNU
Det skapende universitet

Metoder for sikring av kommunikasjon, data og autentisering.

Gorm Andersen

Master i informatikk
Oppgaven levert: Mai 2006
Hovedveileder: Arvid Staupe, IDI

Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

Metoder for sikring av kommunikasjon, data og autentisering

Gorm Andersen

17. mai 2006

FORORD

I løpet av de årene jeg har studert og jobbet innenfor fagfeltet informatikk har jeg hele tiden blitt konfrontert med sikkerhetsspørsmål, både når det gjelder data lagret på disk, og datakommunikasjon. Det har imidlertid vært vanskelig å finne en kortfattet oversikt over disse emnene, ihvertfall skrevet på norsk.

Da jeg gikk i gang med denne hovedoppgaven var målet mitt å gi en full oversikt over datasikkerhet, kommunikasjonssikkerhet og autentisering, men det viste seg veldig tidlig at dette ble altfor omfattende for en hovedoppgave. Jeg vinklet derfor oppgaven på en slik måte at den skulle bli nyttig og lesbar for vanlige mennesker som bruker datamaskiner. De aller fleste bøkene jeg har lest innen de emnene jeg tar opp i denne oppgaven legger seg på et altfor høyt nivå rent teknisk og matematisk. For vanlige brukere er det ikke så interessant å se hvordan de forskjellige krypteringsalgoritmene fungerer og hvordan dette utledes matematisk. De er som regel mer interessert i hvordan dette fungerer i praksis og hvordan de selv kan bruke dem.

Utfordringen min ble da ekstrahere den teknisk kompliserte informasjonen fra alle de fagbøkene og kilder på internett inn i oppgaven slik at det blir lettlest og forståelig. I tillegg til dette ville jeg ha med en egen praktisk del i oppgaven der oppsettet av forskjellige sikkerhetssystemer forklares steg for steg med figurer og tekst slik at brukere kan sette dette opp selv ved en senere anledning.

Det jeg håper å oppnå med denne oppgaven er å gi en introduksjon til sikkerhet innen data og datakommunikasjon slik bevisstheten rundt sikkerhetsspørsmål for de som leser den øker.

Trondheim, 17 Mai 2006
Gorm Andersen

Sammendrag

Hovedfokus i denne hovedfagsoppgaven er å gi en oversiktlig og lettfattelig innføring i data og kommunikasjonsikkerhet. Det er lagt vekt på å holde oppgaven på et såpass lavt teknisk og matematisk nivå at den vil være mulig å forstå for personer uten store datakunnskaper.

Oppgaven er organisert på den måten at den starter med en grunnleggende innføring i kryptografi. Dette er nødvendig for å forstå resten av oppgaven. Deretter forklares og vises de mest utbredte sikkerhetsmekanismene innen data og kommunikasjonsikkerhet. Etter det tar jeg for meg en del spesifikke applikasjoner som brukes mye, og hva som kan gjøres for å sikre kommunikasjonen ved å ta i bruk disse.

Til slutt i den teoretiske delen går jeg gjennom andre kommunikasjonsmetoder enn ethernet, som datamaskiner som oftest bruker, og hvilke sikkerhetsmekanismer som ligger innebygd i disse samt hva vi som brukere kan gjøre for å øke sikkerheten.

For å gjøre sikkerhet mer tilgjengelig for vanlige brukere har jeg i den praktiske delen av oppgaven valgt å gi detaljerte beskrivelser på oppsett av tre vanlige sikkerhetsmekanismer. De tre jeg har valgt å forklare er IPSec, SSL og digitale signaturer. Grunnen til at valget falt på disse er at de på en måte representerer hvert sitt aktuelle felt innen sikkerhet. IPSec benyttes når en vil ha total sikkerhet i kommunikasjonen mellom maskiner, SSL benyttes når en bare vil ha sikkerhet på informasjonsutveksling mellom utvalgte applikasjoner og digitale signaturer tar for seg integriteten til informasjon som sendes over usikre nettverk.

Figurene og beskrivelsene vil gjøre de som leser denne oppgaven i stand til å sette dette opp selv på sine egne datamaskiner.

Innhold

Figurer

Tabeller

Ord og Begreper

ASCII	American Standard Code for Information Interchange. Dette er en kode som representerer alle bokstaver, nummer og symboler som en 8 bits kode. (7 for data og 1 for paritet)
Asymmetrisk kryptering	Dette begrepet innebærer at krypteringen foregår med en offentlig tilgjengelig nøkkel.
DES	Data Encryption Standard. En metode som brukes for å kryptere data.
DOS	Denial of Service. Dataangrep som overstrømmer datamaskinen med unyttig trafikk slik at den ikke får utført sine primæroppgaver.
HTTP	HyperText Transfer Protocol. Kommunikasjonsprotokollen som benyttes mellom nettleser og webtjener.
HTTPS	HyperText Transfer Protocol Secure. Kommunikasjonsprotokollen som brukes mellom nettleser og webtjener for å overføre data sikkert.
ICMP	Internet Control Message Protocol. TCP/IP protokoll som benyttes til å sende feilmeldinger og informasjon.
Intranett	Internt privat nettverk som er avgrenset fra internett med en brannmur e.l.
IPv4	Internett Protocol version 4 med 32 bits adresserom.
IPv6	Internett Protocol version 6 med 128 bits adresserom.
Kryptogram	Kryptert melding.

MIME	Multipurpose Internat Mail Extensions. Standarden som brukes for å sende ikke-tekst via e-post.
PGP	Pretty Good Privacy. Et offentlig-nøkkel krypteringssytem som brukes til å kryptere og signere filer og e-post.
SSL	Secure Socket Layer. En protokoll som benyttes for å sende informasjon sikkert over internett.
Symmetrisk kryptering	Privat nøkkel kryptering. Nøkkelen er ikke offentlig tilgjengelig og gir derfor en sterkere kryptering.
TCP/IP-stakk	Transmission Control Protocol / Internet Protocol - En samling protokoller som definerer grunnlaget for kommunikasjon via internett.
TLS	Transport Layer Security. En videreutvikling av SSL som gir større flekibilitet og sikkerhet.
UDP	User Datagram Protocol. Kommunikasjonsprotokoll som sender data fra applikasjon til applikasjon uten feilsjekking.
VPN	Virtual Private Network. Et privat nettverk som settes opp på vanlig telekommunikasjonsutstyr og gir sikkerhet gjennom tunnelering.

Del I

Data og Kommunikasjonsikkerhet

1 Problemstillinger

De siste årene har vi sett en stor økning i at folk bruker internett til å utveksle informasjon av en mer privat og sensitiv karakter enn den vanlige surfingen på internettsider. Vi benytter oss mer og mer av internett når det gjelder ting av både personlig og økonomisk karakter. Et eksempel på økonomiske ting kan være bruken av nettbank, der det sendes og mottas informasjon som man slett ikke vil la andre få tilgang til. Av en mer privat karakter kan en for eksempel nevne at man ved de fleste universitet får tilgang til alle sine karakterer via en nettleser. Det er kanskje ikke så farlig at akkurat denne informasjonen kommer på avveie, men det kan kanskje i enkelte tilfeller være pinlig.

For at utveksling av informasjon over internett skal kunne fortsette å øke må folk ha tillit til at informasjonen som mottas er korrekt og at informasjonen som sendes ikke kan leses av andre enn den er tiltenkt.

De aller fleste brukere er ikke klar over hvilken risiko de utsetter seg for når de utveksler informasjon over internett, eller hvilke teknologiske løsninger som allerede eksisterer for å imøtekomme eventuelle trusler. Dette fører til at mye av ansvaret for sikkerhet tilfaller de som tilbyr tjenester der det må utveksles sensitiv informasjon. Et tankeeksperiment er: hvor mange "vanlig" brukere ville brukt et SSL[1] sertifikat på nettbanken sin dersom det var frivillig ? Opprettelsen av sertifikatet er kanskje litt vanskelig hvis du er usikker på databehandling generelt, samt at en må holde styr på hvilken bruker/maskin man laster ned sertifikatet på osv. Derfor tror jeg at de fleste ville valgt bort sertifikatet dersom man fikk valget. Ut i fra dette er det kanskje greit at de seriøse aktørene styrer sikkerheten og innfører de policy'ene de synes er nødvendig ut i fra hvilken type informasjon som utveksles. Men hva er egentlig informasjonssikkerhet ?

Informasjonssikkerhet er et meget omfattende og komplekst felt som inngår i mange områder innen databehandling. For å få en bedre oversikt kan man bryte det ned i følgende 2 hovedpunkter:

1. Datasikkerhet (computer security)
   Som omhandler sikringen av informasjon som er lagret på en datamaskin.

2. Kommunikasjonsikkerhet (network/internet security)
   Som omhandler sikkerheten til informasjon som sendes mellom 2 eller flere maskiner.

[1] Secure Socket Layer - En metode for å kryptere sensitiv informasjon som sendes over nettet

1.1 Datasikkerhet

Den absolutt eneste måten å være helt sikker på at informasjon lagret på en datamaskin er sikker, er å ikke ha den tilkoblet et nettverk. Det finnes mange typer informasjon som det ikke er ment å være tilgjengelige, ihvertfall ikke via et offentlig nettverk, og er derfor lagret på maskiner som ikke har nettverkstilkobling. Dersom det skal være mulig for maskinene å kommunisere med hverandre, vil det være mulig å oppnå dette ved å koble de til lukkede private nettverk som det ikke er tilgang til fra internett. Eksempler på dette kan være maskiner som inneholder sensitive militær informasjon og statshemmeligheter.

På de aller fleste tjenere derimot er det ikke så praktisk siden det da ikke vil være like enkelt å få tilgang til denne informasjonen uten en eller annen form for fysisk overlevering. Derfor er de aller fleste datatjenere tilkoblet nettverk og må da iverksette de nødvendige tiltakene for å beskytte den lagrede informasjonen.

Datasikkerhet omfatter også det at den informasjonen man får tilgang til er korrekt og ikke forandret på noen som helst måte siden den ble lagret. Og den eneste forandringen som er tillatt er den som er utført av autorisert personell. De største truslene mot datasikkerhet er:

1. Dårlig oppsett av datamaskinen.
   En stor trussel mot dataintegritet er dårlig oppsett og vedlikehold av datamaskinen. Dette inkluderer for eksempel at man lar være å installere sikkerhetsoppdateringer og oppgraderinger anbefalt av leverandøren av operativsystemet. For å opprettholde et godt sikkerhetsnivå på en datamaskin må en hele tiden følge med i informasjon fra både leverandør av operativsystemet og forskjellige sikkerhets fora. For enhver tjenermaskin som er tilkoblet et usikret nettverk må det etableres en policy for hvordan maskinen skal konfigureres og holdes oppdatert med hensyn på sikkerhet.

2. Egne brukere.
   Trusler fra egne brukere kan enten klassifiseres som uærlige/farlige eller som utilsiktet/utprøving. Den første kategorien er bevisst og det viser seg at de aller fleste angrep nettopp kommer fra slike brukere. Grunnen til dette er at det er mye enklere å bryte seg inn i et system å få tak i informasjon når man allerede har brukeradgang til systemet. En kan si at det største problemet, det å få tilgang til systemet, allerede er løst.
   Den andre kategorien er mindre farlig og skyldes først og fremst dårlig informasjon om systemets policy, samt eventuelle straffereaksjoner.

3. Eksterne brukere.
   Dette er et mindre problem, men farligere med tanke på at eksterne brukere kan få tak i informasjon. De aller fleste angrep i denne kategorien

kommer via virus, ormer, tjenestenekt (DOS[2]) og hacker angrep.

4. Tyveri.
   En ting som sjelden tas med når sikkerhetspolitikken defineres er tyveri av datautstyr med sensitiv informasjon. Det hjelper lite å ha sikret en datamaskin mot angrep via nettverk, hvis det er mulig å stjele hele maskinen å få tilgang til all informasjonen lagret på denne. Dette er særlig et problem når det gjelder bærbare datamaskiner.

5. Sletting av sensitiv data ved kassering.
   En ting det syndes ofte mot er sletting av sensitiv data som er lagret på datamaskinen når den kasseres. For å være sikker på at all informasjonen blir slettet for godt og er umulig å gjenopprette bør hele lagringsenheten overskrives 3-7 ganger. Det finnes mange gratis programmer som utfører dette for deg, som for eksempel DBAN. (`http://dban.sourceforge.net/`)

1.2 Kommunikasjonsikkerhet

Kommunikasjonsikkerhet tar for seg selve transporten av informasjon mellom 2 eller flere maskiner og at denne informasjonen ikke kan fanges opp av en tredjepart. All kommunikasjon, unntatt verbal kommunikasjon utenfor hørevidde av en tredjepart, kan teoretisk sett bli avlyttet av personer informasjonen ikke var tiltenkt for. De største truslene for Kommunikasjonsikkerhet er:

1. Avlytting av kommunikasjon
   I dette tilfellet vil en tredjepart avlytte kommunikasjonen mellom 2 parter. Dette kan være en potensiell sikkerhetsrisiko dersom det er snakk om sensitiv informasjon, som for eksempel kommunikasjonen mellom en bruker og nettbanken.

2. Avlytting samt endring av informasjon.
   Her snakker vi om en tredjepart som ikke bare avlytter, men også endrer informasjonen før den blir sendt videre. Dette forutsetter i de fleste tilfellene at angriperen må ha tilgang til nettverksutstyr mellom de 2 partene som kommuniserer. Dette er den typen angrep alle brukere frykter, og da spesielt når det gjelder økonomiske transaksjoner.

3. Identitetstyveri
   Her vil en tredjepart gå inn i kommunikasjonen mellom partene og utgi seg for å være en annen. Dette er også potensielt farlig dersom det utveksles sensitiv informasjon. Et annen eksempel på dette er falsk deltaker

[2]DOS - Denial of Service. Angrep som påvirker tjenestetilgangen på maskinen.

(phising), der brukere blir narret til å tro de kommuniserer med en legitim nettadresse, men i virkeligheten kommuniserer med en annen.

Men det finnes verktøy som sørger for at dette blir så vanskelig at det sjelden eller aldri er verd bryet å forsøke. De forskjellige metodene for å øke kommunikasjonsikkerheten er:

1. Gjøre datakommunikasjonen vanskelig å tyde.
   Dette kan gjøres ved å benytte seg av en av følgende metoder:

   (a) Kryptering
   Ved å kryptere (kryssreferanse til krypteringskapittelet) data kan man gjøre det meget vanskelig for en tredjepart å få lest informasjonen

   (b) Steganografi (gjemt skrift)
   Dette går ut på at man gjemmer data inne i annen informasjon, for eksempel et bilde. (eksempler fra gresk historie kanskje ?)

2. Anonymisering av de som kommuniserer

   (a) Ved å bruke anonyme kommunikasjonsutstyr, som for eksempel uregistrerte mobiltelefoner eller ved å benytte en internettkafé.

   (b) Anonyme proxy-servere.

3. Vanskeliggjøre observasjonen av kommunikasjon

   (a) Nåla i høystakken.
   Dette betyr at man gjemmer seg blant store mengder av kommunikasjon får å gjøre det vanskelig å avlytte kommunikasjonen.

   (b) Kontinuerlig trafikk.
   Hvis man sender kontinuerlig trafikk mellom partene som kommuniserer der bare små deler av trafikken er sensitiv, vil det være veldig vanskelig for en tredjepart å kunne tyde alt og samtidig plukke ut den informasjonen som er viktig

2 Kryptografi

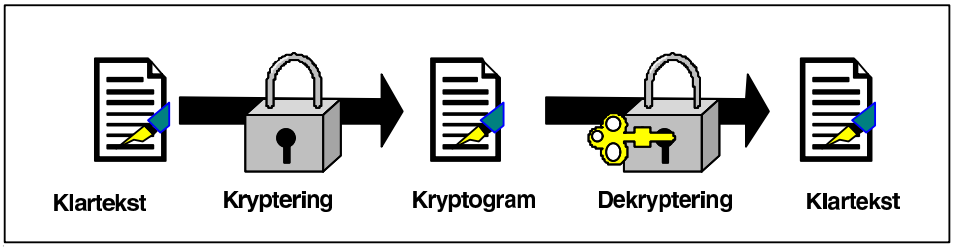
Hensikten med dette kapittelet er å gi en generell innføring i kryptografi slik at leseren har mulighet for å forstå begrepene som blir brukt senere i oppgaven.

2.1 Generell historie

Da Julius Cæsar sendte beskjeder til sine generaler, stolte han ikke på budbringerne sine. Det han da gjorde var å erstatte hver bokstav med bokstaven tre plasser lengre ut i alfabetet. For eksempel ble bokstaven 'a' erstattet med 'd' og 'b' erstattet med 'e'. Dette gjorde meldingen umulig å lese dersom en ikke kjente til kodingsalgoritmen som ble brukt for å kryptere meldingen. I dag vil denne formen for kryptering se banal ut, men på en tid der veldig få kunne lese og skrive gav det en god sikkerhet. Dagens krypteringer er langt mer avanserte enn dette, men prinsippene er de samme.

2.2 Kryptering og dekryptering

Data som kan leses og forstått uten å må behandles på noen måte, kalles ofte for ren-tekst eller klartekst (plaintext eller cleartext). Metoden som brukes for å gjøre slik data uforståelig for andre kalles kryptering. Når en krypterer data resulterer dette i uforståelig informasjon som vi kaller kryptogram *(ciphertext)*. Kryptering brukes for å forhindre at noen kan lese informasjonen, noe som også inkluderer de som kan lese de krypterte dataene. Prosessen som konverterer kryptogrammet til forståelig data igjen, kalles for dekryptering.



Figur 1: Kryptering og Dekryptering

2.3 Nøkler

En nøkkel er en verdi som brukes som innverdi i en krypteringsalgoritme for å produsere et unikt kryptogram. Nøkler er i all enkelhet egentlig bare veldig store tall. Nøkkellengden måles i bits, og i teorien så vil en lang nøkkel være sikrere enn en kort nøkkel. Det er ingen sammenheng mellom nøkler i kryptering med

offentlig-nøkkel (Kapittel 2.6 på side 9) og nøkler i konvensjonell kryptering. (Kapittel 2.5 på neste side) En 80-bits nøkkel i konvensjonell kryptering har en styrke som er lik en 1024-bits i kryptering med offentlig-nøkkel.[Lenstra and Verheul, 2000, side 446-465] En lang nøkkel vil være sikrere enn en kort i begge systemene, men algoritmene som brukes i de to systemene er så forskjellige at de ikke kan sammenlignes.

I kryptering med offentlig-nøkkel henger det private og offentlige nøkkelparet sammen ved hjelp av matematiske algoritmer. Det er derfor veldig vanskelig å få ut den private nøkkelen fra den offentlige nøkkelen, men det er mulig dersom man har nok tid og prosessorkraft. Dette gjør det viktig å velge nøkler med rett størrelse; store nok slik at de er relativt sikre, men samtidig små nok til at de er praktisk brukbare ved at det ikke tar for lang tid å kryptere med dem. Lange nøkler vil teoretisk sett være sikrere i lengre tid enn korte. Derfor er det fornuftig å bruke lange nøkler for data som skal krypteres og lagres i lengre tid, mens det for data med kort holdbarhet kanskje holder med korte nøkler.

2.4 Hva er kryptografi ?

Kryptografi er en metode som bruker matematikk til å kryptere og dekryptere data. Kryptografi gjør det også mulig å lagre eller sende sensitiv informasjon i usikre nettverk, som for eksempel internett, slik at innholdet ikke kan leses av andre enn det er ment for.

2.4.1 Sterk eller svak kryptografi

Kryptografi kan være sterk eller svak. Styrken måles ut ifra tiden og ressursene det vil ta å hente ut den krypterte informasjonen til klartekst slik at den kan leses. Resultatet av *Sterk Kryptografi* er et kryptogram som er veldig vanskelig å reversere uten å ha den nødvendige nøklene. Akkurat hvor vanskelig det er å hente ut informasjon som er kryptert på denne måten er litt vanskelig å svare på, men i dagens krypteringssystemer brukes det 128 eller 256 bits nøkler, noe som gir temmelig store tall. Hvis man regner med 128-bits nøkler og hver bit kan være 0 eller 1, så er det 2^{128}, eller tilnærmet $3x10^{38}$ kombinasjoner. Hvis man tenker seg at 10 milliarder datamaskiner tas i bruk for å knekke nøkkelen og at hver av disse kan teste 10 milliarder nøkler i sekundet. Setter man sammen dette vil maskinene kunne teste 10^{20}nøkler i sekundet. Dette vil gi:

$3x10^{38}$nøkler / 10^{20}pr. sekund = $3x10^{18}$ sekunder

Noe som tilsvarer 100 milliarder år.

Statistisk sannsynlighet tilsier at du antageligvis vil holde å sjekke rundt halvparten av nøklene for å finne den rette, men dette vil fortsatt ta rundt 50 milliarder år.

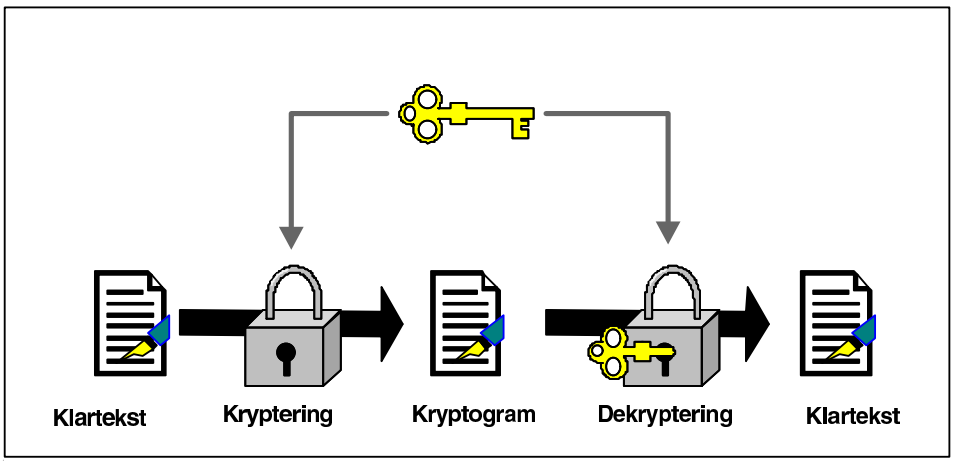
Det det derimot ikke tas hensyn til her er hvor mye raskere datamaskinene blir i fremtiden. Dette er det ikke mulig å spå noe om, men om man følger "Moores Lov" som sier at datakraften dobles hver 9 måned, vil det ikke utgjøre noen praktisk forskjell.

2.4.2 Hvordan virker kryptografi

En krypteringsalgoritme eller kodenøkkel er en matematisk funksjon som brukes i en krypterings eller dekrypteringsprosess. Algoritmen brukes i kombinasjon med en nøkkel - et ord, et tall, en setning - for å kryptere klartekst. Den samme klar-teksten krypteres til forskjellig kryptogram ved bruk av forskjellige nøkler. Sikkerheten til de krypterte dataene avhenger av to ting: styrken til krypteringsalgoritmen og hemmeligholdelse av nøkkelen.

2.5 Konvensjonell kryptografi

I konvensjonell kryptografi, som også kalles hemmelig-nøkkel eller symmetrisk-nøkkel kryptering (secret-key eller symmetric-key), brukes samme nøkkel til både kryptering og dekryptering. Data Encryption Standard (DES[3]) er et eksempel på et slikt krypteringssystem.



Figur 2: Symmetrisk kryptering

2.5.1 Cæsars kodenøkkel

Et veldig enkelt eksempel på konvensjonell kryptografi er Cæsars kodenøkkel. I denne formen for krypterings bytter man ut en informasjonsbit med en annen.

[3]Data Encryption Standard - Den mest brukte krypteringsmetoden

Som oftest gjøres dette med å bytte bokstavene i alfabetet ett eller flere hakk, slik at bokstaven 'a' gir 'd' og bokstaven 'b' gir 'e'. Dette blir ofte kalt "Shift by x", der x står for hvor mange bokstaver en går opp. Det er også mulig å benytte andre metoder for å bytte informasjonsbiter. En kan for eksempel representere bokstavene med tall. 'a'=1 'b'=2 og så videre. Den som skal dekryptere en slik melding må vite hvilken nøkkel som er brukt for å kunne dekryptere meldingen.

Cæsar's kodenøkkel er et godt eksempel på hvordan konvensjonell kryptering virker, men akkurat denne krypteringsalgoritmen er på langt nær kraftig nok for kryptering i dag.

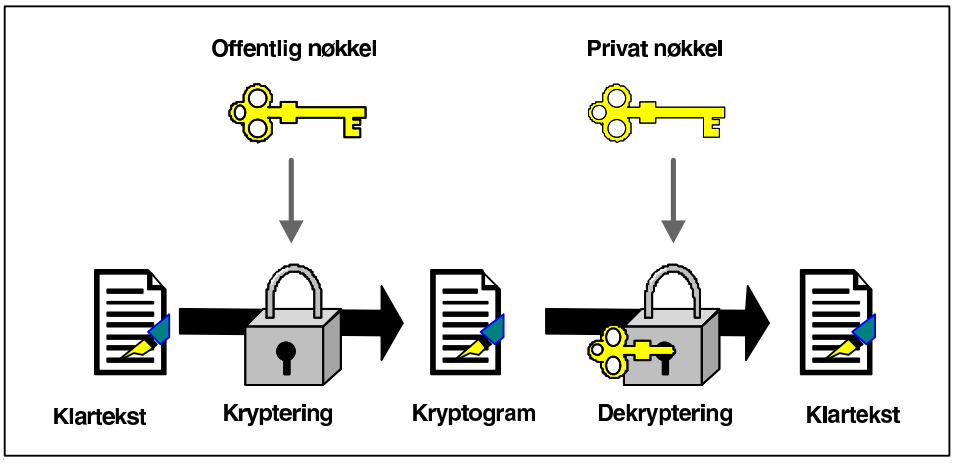
2.5.2 Nøkkelhåndtering

Konvensjonell kryptering har en del fordeler: Det er veldig raskt og er nyttig for kryptering av data som ikke skal sendes til noen andre. Grunnen til dette er problemet konvensjonell kryptering har med å utveksle nøklene som skal til for å dekryptere dataene mellom sender og mottaker. Der dokument skal sendes kryptert og sikkert over et usikret nett må også nødvendigvis nøklene som skal til for å dekryptere også sendes over det usikre nettet. Det har alltid vært akilleshælen til alle konvensjonelle krypteringssystemer, fra DES til Cæsar's kodenøkkel. En måte å omgå problemet på, er å sende nøkkelen med en kurer som begge parter stoler på. Det er som regel metoden nettbanker og kredittkortselskaper opererer på, der de sender passord rekommandert med posten til den som skal ha nøkkelen. Fortsatt er ikke metoden 100% sikker og nøkler kan fortsatt bli kompromittert ved for eksempel utro tjenere som arbeider i postverket.

2.6 Offentlig-nøkkel kryptografi

Problemet med nøkkelhåndtering i konvensjonell kryptografi løses ved hjelp av offentlig-nøkkel kryptografi som ble lansert av Whitfield Diffie og Martin Hellman i 1975. [Diffie and Hellman, 1976]

Offentlig-nøkkel kryptografi er en asymmetrisk metode som bruker et nøkkelpar for kryptering og dekryptering av data: en *offentlig-nøkkel* for å kryptere, og en korresponderende *privat* nøkkel for å dekryptere. Den offentlige nøkkelen er fritt tilgjengelig for alle, mens den private nøkkelen holdes hemmelig. Alle som har tilgang til den offentlig-nøkkelen kan kryptere data som kun den med den private nøkkelen kan dekryptere. Måten det fungerer på vises her:



Figur 3: offentlig-nøkkel kryptering

Fordelen med offentlig-nøkkel kryptering er at det ikke behøves noen eksisterende rammeverk for å sende data sikkert. Nødvendigheten for å finne sikre måter slik at sender og mottaker kan utveksle nøkler er eliminert. All kommunikasjon baserer seg på kryptering med en offentlig-nøkkel som legges offentlig tilgjengelig. Den private nøkkelen som trengs for å dekryptere dataene blir aldri sendt over et usikret nett.

Eksempler på slike offentlig-nøkkel kryptosystemer er Elgamal (navngitt etter oppfinneren, Taher Elgamal), RSA (navngitt etter oppfinnerne, Ron Rivest, Adi Shamir og Leonard Adleman), Diffie-Hellman (etter oppfinnerne, Whitfield Diffie og Martin E. Hellman) og DSA (Digital Signature Algorithm)

2.7 Digitale signaturer

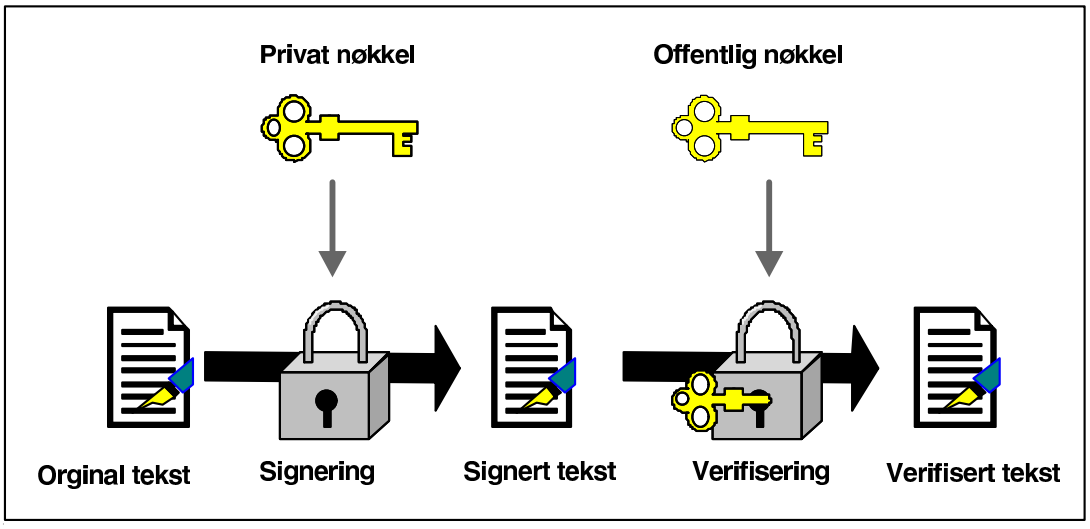
(se kapittel 6 på side 46)

En stor fordel kryptering med offentlig-nøkkel gir er metode for å generere *digitale signaturer*. Digitale signaturer gjør det mulig for mottakeren å verifisere ektheten til informasjonen, det vil si at mottakeren kan være sikker på at informasjonen ikke er forandret siden den ble skrevet, og at den som skriver dokumentet ikke kan fraskrive seg opphavet til dokumentet, siden det er umulig å generere en gyldig digital signatur uten den private nøkkelen.

En digital signatur gjør samme nytten som en håndskrevet underskrift. Men i motsetning til en håndskrevet underskrift som er forholdsvis enkel å forfalske, er digitale signaturer nesten umulig å forfalske samt at den gir mulighet til å verifisere at innholdet ikke er forandret.

Figur 4 illustrerer hvordan en digital signatur fungerer. Senderen har på forhånd skaffet seg en privat og en offentlig nøkkel og den offentlige nøkkelen er

lastet opp til en nøkkeltjener. (se kapittel 13.2.1 på side 97) Deretter lages det en hash (se kapittel 2.8) ut fra dokumentet som så krypteres med den private nøkkelen. Deretter sendes dokumentet og den krypterte hash'en til mottakeren. Mottakeren må da først dekryptere den mottatte hash'en med den offentlige nøkkelen. Dersom dette går bra, er mottakeren sikker på hvem det kommer fra. Deretter genereres det en hash fra det samme dokumentet på mottakerens side. Dersom de to verdiene er like, er vi sikre på at dokumentet ikke er forandret underveis. På denne måten har vi fått bekreftet hvem det er fra og at informasjonen ikke er forandret siden det ble signert.

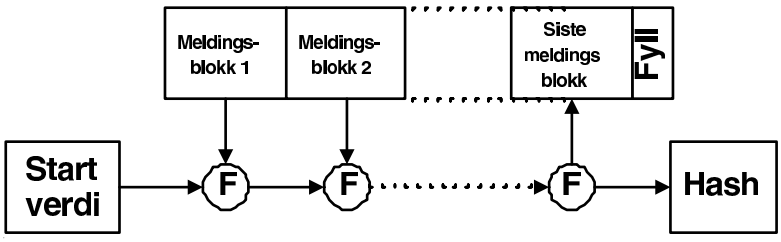


Figur 4: Digitale signaturer

2.8 Hash funksjoner

Problemet med å kryptere hele dokumenter med offentlig-nøkkel kryptering er at dette genererer store datamengder, opp til dobbelt så mye som utgangspunktet. En metode for å løse dette problemet på er å introdusere en en-veis hash funksjon i prosessen. En en-veis hash funksjon tar en melding med variabel lengde og produserer ut av meldingen et ekstrakt med en bestemt lengde. Hash funksjonen vil da gå igjennom hver bit i meldingen og kalkulere et ekstrakt for hele dokumentet samlet. Denne verdien vil være av en bestemt predefinert lengde og dersom meldingen blir forandret på noe vis, vil dette ekstraktet forandres fullstendig.

Metoden ekstraktet blir generert på er illustrert i figur 5.

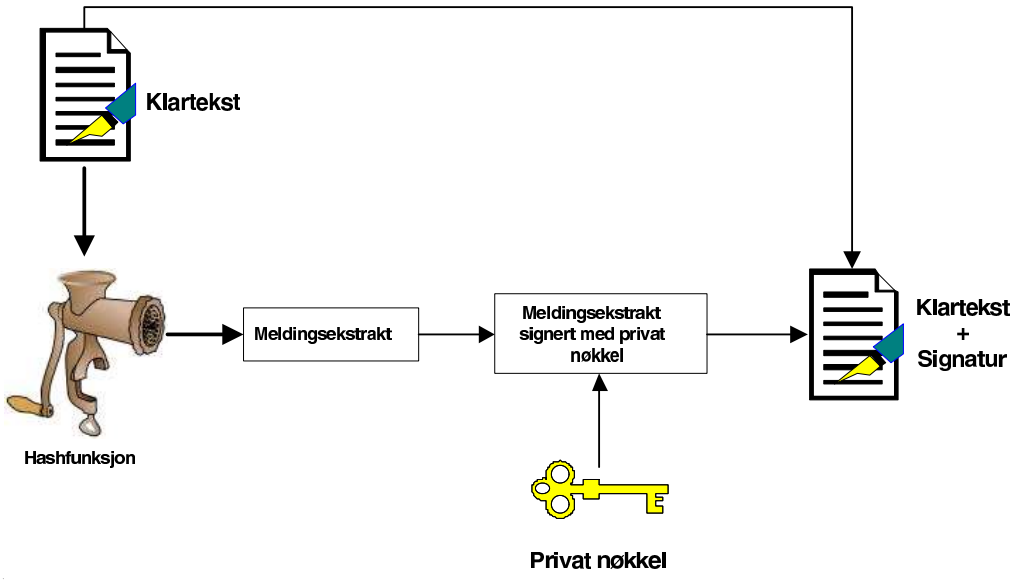


Figur 5: Hash algoritme.

Meldingen som sendes deles opp i blokker og det legges til fyll på slutten dersom det trengs. Dette gjøres for at den skal kunne deles i like store blokker og at den skal gi det samme ekstraktet hver gang. Deretter lages det et ekstrakt ut fra startverdien og alle disse ekstraktene.

Resultatet av en hash funksjon kalles ofte for *meldingsekstrakt* (Message Digest). Dersom en eneste bit i meldingen blir forandret, vil det gi et helt annet meldingsekstrakt.

Hvis vi så kombinerer en hash funksjon med digitale signaturer får vi følgende:

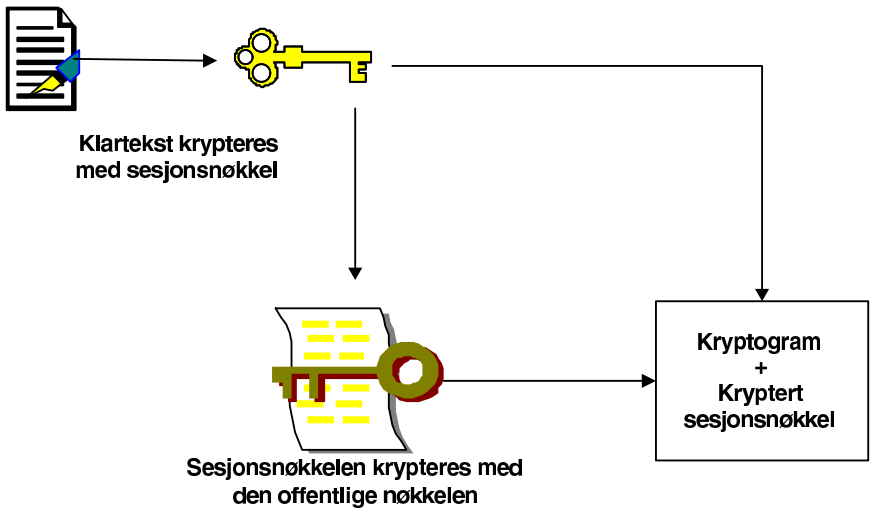


Figur 6: Digital signatur med hashfunksjon

Forskjellen blir da at en ikke signerer hele dokumentet, men kun ekstraktet av det. Siden det er tilnærmet umulig å lage to dokumenter som gir samme ekstrakt, vil dette gi en god indikasjon på ektheten til dokumentet og senderen.

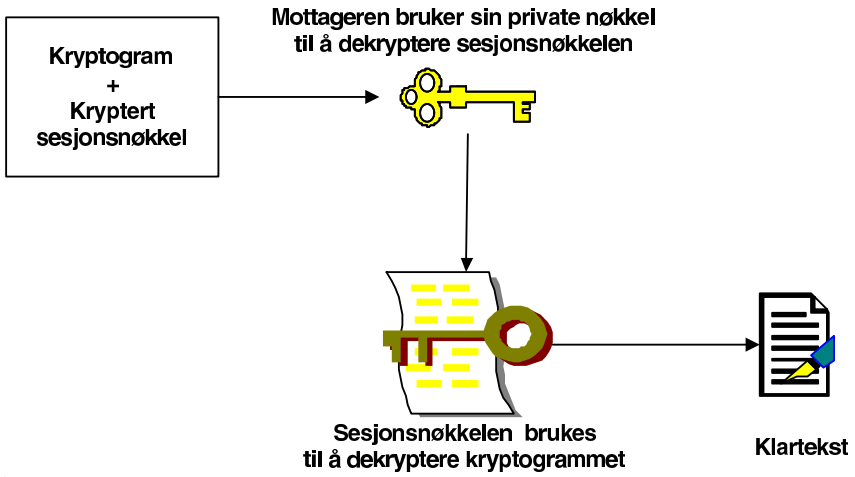
2.9 Pretty Good Privacy (PGP)

PGP er et kryptosystem som benytter seg av både konvensjonell og offentlig-nøkkel kryptering. PGP er dermed et hybrid-kryptosystem. Når en krypterer klartekst med PGP vil først klarteksten komprimeres. Datakomprimering sparer tid ved overføringer, men det er ikke hovedårsaken til at dette gjøres. De fleste kodeknekkerne utnytter sammenhenger i mønstre mellom kryptogrammene og klarteksten for å finne algoritmene. Ved å komprimere meldingene først reduseres faren for at dette kan skje. PGP genererer deretter en nøkkel som kalles "sesjons-nøkkel" (*session key*) som er en en-gangs hemmelig nøkkel. Denne blir som regel generert ut ifra noe tilfeldig, som for eksempel bevegelse av musepekeren. Sesjons-nøkkelen brukes sammen en rask og sikker konvensjonell krypteringsalgoritme til å kryptere klarteksten til et kryptogram. Deretter blir sesjons-nøkkelen kryptert med mottakerens offentlige nøkkel og blir så sendt med kryptogrammet til mottakeren. Her blir det altså først brukt en konvensjonell kryptering på klarteksten og deretter en offentlig-nøkkel kryptering på nøkkelen.



Figur 7: PGP kryptering

Dekryptering av kryptogrammene foregår på motsatt vis. Mottakeren bruker sin private nøkkel til å dekryptere sesjons-nøkkelen som deretter blir brukt til å dekryptere den konvensjonelt krypterte kryptogrammet.



Figur 8: PGP dekryptering

PGP kombinerer de beste egenskapene til de to krypteringsmetodene: Hastigheten til konvensjonell kryptering, som er rundt 1000 ganger raskere, og tilgjengeligheten til offentlig-nøkkel kryptering, som tar seg av problemet med utveksling av nøkler.

2.10 Digitale sertifikater

En utfordring med offentlig-nøkkel kryptering er at brukerne må alltid passe på å kryptere med riktig mottakers offentlig nøkkel. Ved å bruke usikre nett til å distribuere de offentlig nøklene kan man komme opp i såkalte *"man-in-the-middle"* angrep. Med dette menes det at en utenforstående legger ut en offentlig nøkkel med falskt navn. Dermed vil informasjon kryptert med denne offentlige nøkkelen være tilgjengelig til en annen person enn den det var ment for.

Løsningen på dette problemet er å introdusere digitale sertifikater som er en form for akkreditering der en tredjepart vailderer identiteten til den offentlige nøkkelen. En slik tredjepart kalles ofte Sertifikats Autoritet (Certification Authority) og som ofte er godkjente firma eller offentlige institusjoner.

Et digitalt sertifikat inneholder minst minst tre deler:

- En offentlig nøkkel
- Sertifikatsinformasjon
- En eller flere digitale signaturer

Meningen med den digitale signaturen på sertifikatet er å vise at informasjonen sertifikatet gir er attestert av en tredjepart. Den digitale signaturen validerer

ikke ektheten på hele sertifikatet, men at den signerte identiteten er bundet til den offentlige nøkkelen. En kan si at et digitalt sertifikat er en offentlig nøkkel med en eller flere identifikasjonsmerker samt et godkjent-merke fra en betrodd tredjepart.

3 Datakommunikasjon

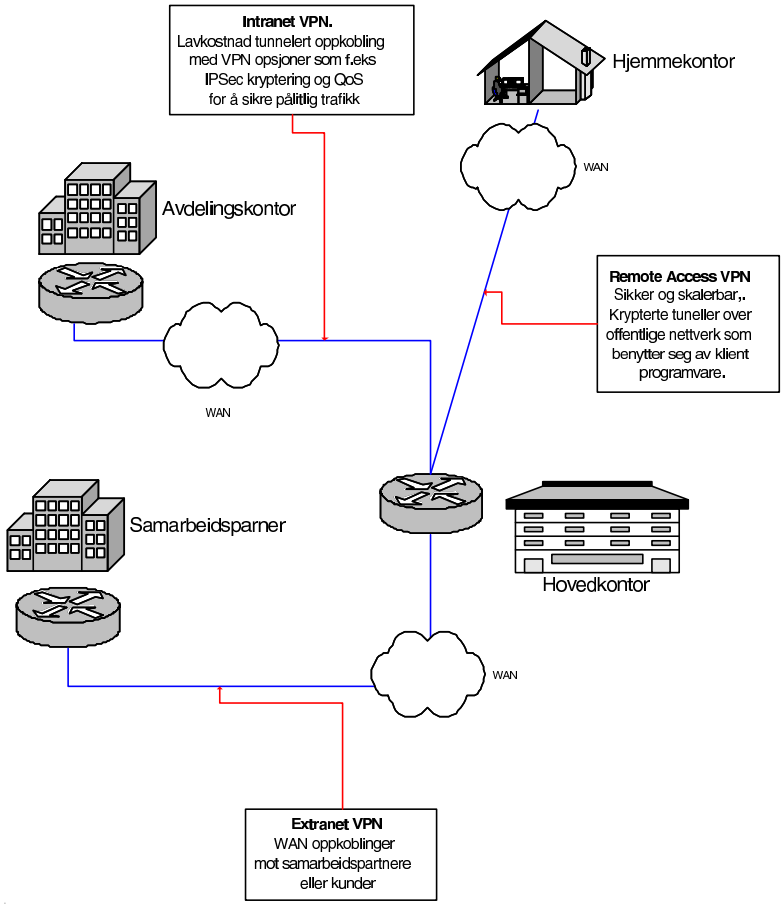
Når vi her snakker om kommunikasjon så omfatter det digitalt kodede elektroniske overføringer som brukes for lagring og prosessering av datamaskiner. Kommunikasjonen mellom datamaskiner kan sikres med en rekke forskjellig metoder og jeg vil ta for meg de viktigste her.

3.1 VPN

Et Virtuelt Privat Nettverk (Virtual Private Network) er en metode som benytter seg av den allerede eksisterende offentlige infrastrukturen for telekommunikasjon, som for eksempel internett, til å tilby en sikker aksess til organisasjonens nettverk. Motstykket til VPN vil være at organisasjonen må eie/leie datalinjene som knytter de forskjellige sammen selv og dermed kun kjøre egen trafikk gjennom disse. Målet med VPN er å tilby den samme funksjonalitet og sikkerhet, men til en mye lavere kostnad.

Det finnes 3 standard VPN typer:

- Remote-Access: også kalt Virtual Private Dial-up Network (VPDN). Det er en Bruker-til-LAN tilkobling der brukere kobler seg opp mot organisasjonens nettverk og dens resursser via VPN.
- Site-to-Site: Ved å bruke fast utstyr kan en organisasjon koble seg opp mot flere faste punkter. Site-to-Site VPN kan være:
  - Intranet-based: Hvis en organisasjon har en eller flere avdelinger som de ønsker å binde sammen til et privat nettverk så kan de bruke Intranett VPN til å koble sammen de forskjellige LAN'ene.
  - Extranet-based: Hvis organisasjonen har nære forbindelser med en annen organisasjon (for eksempel en kunde eller en partner) så kan det settes opp et ekstranet VPN som gjør det mulig for flere organisasjoner å jobbe i et delt miljø.



Figur 9: VPN oversikt

De fleste VPN løsningene benytter seg av tunnelering for å skape et privat nettverk over internett. I et nøtteskall kan man si at tunnelering er å plassere en IP-pakke inn i en annen pakke, kapsle inn, for deretter å sende den over internett. Protokollen i den ytre pakken forstås av nettverket og begge endepunktene, kalt "tunnell interfacene", der pakken går inn og ut av det private nettverket. Tunnelering krever 3 forskjellige protokoller:

- Transport protokoll: Protokollen som brukes av nettverket informasjonen sendes over.
- Tunnellerings protokoll: Protokollen som innkapsler den orginale informasjonen.
- Passasjer protokoll: Den orginale informasjonen (IPX, NetBeui, IP) som blir sendt.

Det har blitt utviklet mange tunnelleringsprotokoller. De tre mest populære er:

- Point-to-Point Tunneling Protocol (PPTP)
  Denne protokollen utvider Point-to-Point Protocol (PPP) standarden for tradisjonell oppringt-nettverk. Den opererer på data link laget i TCP/IP-stakken.

- Layer Two Tunneling Protocol (L2TP)
  Denne protokollen kombinerer en av de tidligere tunnelleringsprotokollene, L2F som for det meste ble brukt av Cisco, og PPTP. Denne opererer, som PPTP, på data link laget i TCP/IP-stakken.

- Internet Protocol Security (IPSEC)
  IPSec er egentlig en samling av relaterte protokoller. Den kan brukes som en komplett VPN protokoll, eller brukes som krypterings skjema innen L2TP eller PPTP. IPSec opererer på nettverkslaget i TCP/IP-stakken.

3.1.1 VPN Sikkerhet.

Et god designet VPN nettverk bruker flere metoder for å holde oppkoblingen og data sikre:

- Brannvegger: En brannvegg gir en sterk beskyttelse mellom private nettverk og internett. Brannveggen kan settes opp til kun å slippe gjennom trafikk på bestemte porter, hvilken type pakker som kan slippe gjennom og hvilke protokoller som kan benyttes. I de fleste nye VPN gatewayer er det som regel innebygget brannvegger.

- Kryptering: Dette er en prosess der en tar all informasjon som en maskin sender til en annen og krypterer denne slik at det bare er mottagermaskinen som kan dekryptere og lese informasjonen. Det finnes 2 typer kryptering i VPN:

  – privat nøkkel (se kapittel 2.5 på side 8)
  – offentlig-nøkkel (se kapittel 2.6 på side 9)

  "Private key" kryptering betyr at hver maskin har en hemmelig nøkkel som brukes for å kryptere informasjon før det sendes til den andre maskinen. "Private key" metoden fordrer at man vet hvilken maskiner som skal kommunisere og at nøkkelen må installeres på hver av dem.

  "Public key" metoden bruker en kombinasjon av en privat nøkkel og en offentlig-nøkkel. Den private nøkkelen er kjent bare av maskinen selv, mens den offentlige blir gitt til enhver maskin som vil kommunisere sikkert med

den. For å dekryptere en melding må maskinen benytte "public key" fra den andre maskinen og sin egen "private key". Måten dette fungerer på i praksis er at den første oppkoblingen benytter "public key". Når så denne oppkoblingen er foretatt sender den ene maskinen en "private key" og deretter går oppkoblingen over på "private key" kryptering.

- IPSec - (Internet Protocol Security Protocol) gir økte sikkerhetsaspekter som for eksempel bedre krypterings algoritmer og en mer utfyllende autentisering. (se eget kapittel)
- AAA Server: (Authentication, Authorization and Accounting) brukes for å trygge sikkerheten fra "Remote Access"-VPN. Når en klient spør om aksess sendes denne gjennom en AAA server som sjekker følgende:
  - Hvem du er (Authentication)
  - Hva du har lov til å gjøre (Authorization)
  - Hva du gjør (Accounting)

3.1.2 Fordeler og ulemper med VPN

VPN gir to hovedgevinster: kostnadsbesparelse og skalerbarhet. En VPN forbindelse eliminerer behovet for å leie private linjer for å sikre kommunikasjonen. Brukeren trenger kun å ha et aksesspunkt til det globale nettet, noe som er forholdsvis rimelig. Det er også lett å utvide sitt eksisterende VPN-nettverk når en skal knytte til seg et nytt nettverk. Det eneste som trengs er at det nye nettverket har tilgang til det globale nettverket også kan resten settes opp i VPN programvaren.

Det er også enkelte ulemper tilknyttet et VPN-nettverk. Dette er i hovedsak fire punkter:

1. VPN krever en dyp forståelse av nettverksikkerhet og nøye planlegging når det settes opp.
2. Tilgjengeligheten og ytelsen gjennom VPN forbindelsen er avhengig av faktorer som er utenfor brukerens kontroll siden dette kjøres over det offentlige nettverket.
3. VPN teknologier fra forskjellige leverandører trenger ikke fungere sammen siden standardene ikke er fastsatt.
4. VPN må forholde seg til andre og mindre stabile protokoller en IP.

3.2 SSL/TLS

SSL (Secure Socket Layer) ble utviklet an Netscape Communications i 1994 for å sikre kommunikasjon over WWW. Like etter begynte Internet Engineering Task Force (IETF) og utvikle en standardprotokoll for å tilby den samme funksjonaliteten. SSL 3.0 ble da brukt som basis for dette arbeidet som er kjent som TLS (Transport Layer Security). Selv om det er forskjeller mellom disse implementasjonene vil de her bli behandlet som en protokoll.

Ved klient/server kommunikasjon er det mange fordeler ved å bruke SSL/TLS. Disse er:

- Kraftig autentisering, meldingssikkerhet og integritet: Primæroppgaven til SSL/TLS er å sikre datakommunikasjon ved hjelp av kryptering, men den tilbyr også klient/server autentisering for å sikre identiteten på de som foretar den sikre kommunikasjonen samt at den sikrer data integritet.
- Interoperabilitet: SSL/TLS virker med de fleste nettlesere og operativsystemer.
- Algoritme fleksibilitet: SSL/TLS gir valgmuligheter for de autentiseringsmekanismer, krypteringsalgoritmer og hashingalgoritmer som blir brukt under den sikre kommunikasjonen.
- Enkel å ta i bruk: Siden SSL/TLS er implementert under applikasjonlaget vil de fleste operasjoner være usynlig for klienten. Dette gjør at klienten ikke trenger å vite noe om den sikre kommunikasjonen og fortsatt være beskyttet.

Men det er også noen bakdeler ved å bruke SSL/TLS også:

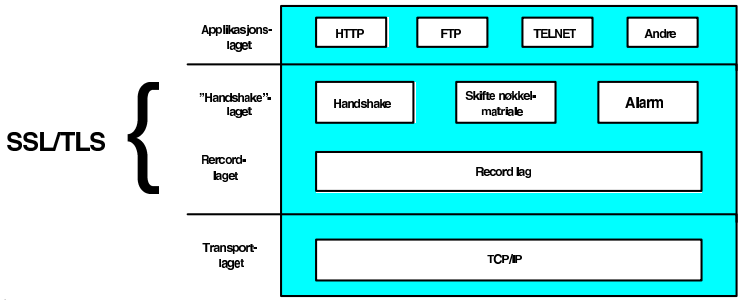
- Økt prosessorbruk: Kryptering, og da spesielt "public key"-operasjoner, er CPU-intensiv.
- Administrative kostnader: Et SSL/TLS oppsett er ofte komplekst og krever vedlikehold.

SSL/TLS brukes primært til å kryptere data som blir sendt over et åpent nettverk som for eksempel internett. I HTTPS protokollen vil URL, HTTP hodet, cookies [4] og data sendt via forms bli kryptert

SSL/TLS sikkerhetsprotokollen ligger mellom applikasjonslaget og TCP/IP laget der det kan sikre og sende applikasjonsdata til transportlaget. Siden det

[4]Cookies er informasjonskapsler som nettleseren får av webtjeneren. Disse inneholder ofte informasjon om hvem du er og hvor du kommer fra. Disse cookiene sendes så til webtjeneren ved påfølgende forespørsler slik at tjeneren kan kjenne deg igjen.

ligger mellom applikasjonslaget og TCP/IP laget støtter SSL/TLS mange forskjellige protokoller på applikasjonslaget. Protokollen kan videre deles inn i to lag:



Figur 10: SSL/TLS protokoll

3.2.1 "Handshake Protokol layer"

Dette laget består av tre under-protokoller som utfører en rekke viktige sikkerhetsfunksjoner:

- Handshake Protocol
  Denne underprotokollen brukes for å forhandle sesjonsinformasjon mellom klient og server. Sesjonsinformasjonen består av en sesjons ID, sertifikat, krypteringsform som skal brukes, komprimeringsalgoritmen som skal brukes og en hemmelig nøkkel som brukes for å generere nøkler.

- Change Cipher Spec Protocol
  Denne underprotokollen brukes for å skifte nøkkel materialet,"keying material", som brukes mellom klient og server. "Keying material" er rå data som brukes for å lage nøkler for kryptering. protokollen består av en enkel melding som forteller den andre part i SSL/TLS sesjonen at sender vil skifte til et nytt sett med krypteringsnøkler. Nøklene genereres fra informasjon som er utvekslet i "Handshake" underprotokollen.

- Alert Protocol
  Denne underprotokollen brukes for å sende statusmeldinger mellom klient og server med for eksempel statusendringer og feilmeldinger.

3.2.2 Autentisering

Til autentisering bruker "Handshake"-protokollen X.509[5] sertifikat som gir sterkt bevis til den andre part om at identiteten til den parten som har sertifikatet og den korresponderende private nøkkelen er riktig. Et sertifikat er en digital identifikasjon som gis ut av et godkjent firma, eller Sertifikats Autoritet (Certification Authority), og inneholder identifikasjonsinformasjon, en gyldighetsperiode, en offentlig-nøkkel, et serienummer og den digitale signaturen til sertifikats utgiveren. En Sertifikats Autoritet er en godkjent leverandør av sertifikater som bekrefter identiteten til den part som har sertifikatet.

3.2.3 Kryptering

SSL/TLS bruker "public key"-kryptering for å autentisere serveren for klienten. Det vil si at det brukes et nøkkelpar som genereres gjennom en komplisert matematisk prosess. En av nøklene blir offentliggjort gjennom Sertifikats Autoritetens sertifikat for sertifikatsholderen. Den private nøkkelen blir holdt hemmelig for alle parter. Disse nøklene jobber sammen på den måten at de utfører den omvendte operasjonen av den andre. Det vil si at om en melding er kryptert med den offentlige nøkkelen kan kun den private nøkkelen dekryptere meldingen og omvendt.

3.2.4 Hashing Algoritme

I "Handshake" prosessen blir partene enige om hvilken hash algoritme som skal brukes. En hash funksjon er en funksjon som komprimerer et input av vilkårlig lengde til et resultat av fast lengde. De hash algoritmene som brukes her er enveis noe som vil si at det ikke er mulig å gjenskape den orginale data ut fra hash nøkkelen. I dette tilfellet kan en sammenligne en hash med et fingeravtrykk: et fingeravtrykk er unikt for et individ og mye mindre en den orginale personen, og man kan ikke gjenskape hele personen ut fra fingeravtrykket.[Stallings, 2003, Kapittel 12 side 347-377]

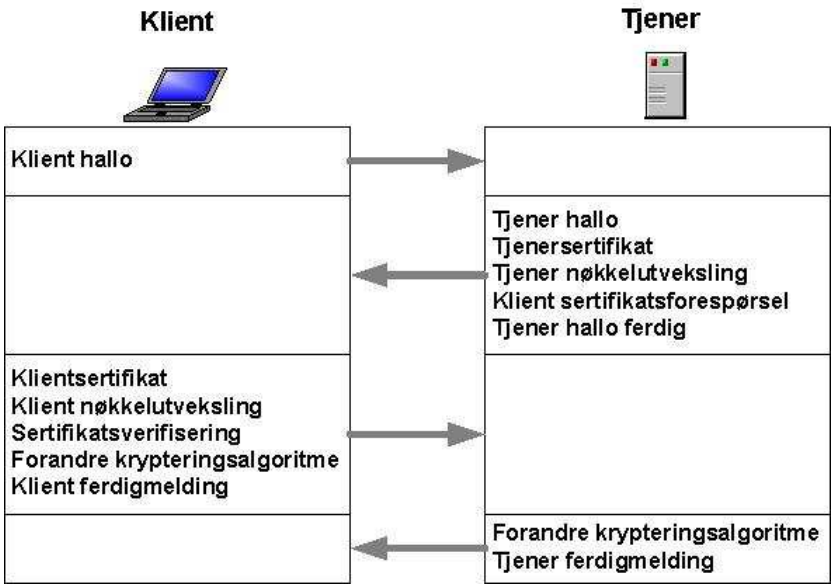
Hashing blir brukt for å sikre dataintegritet under transport. De to mest vanlige hash algoritmene i dag er "Message Digest 5" (MD5) og Standard Hash Algoritme 1 (SHA-1). MD5 gir 128 bits hash verdier og SHA-1 gir 160 bits verdier. Hash algoritmen inkluderer en verdi som brukes til å sjekke integriteten på data som blir overført. Denne verdien blir dannet enten ved å bruke MAC (Message Authentication Code) eller HMAC (Keyed-Hashing for Message Autentication). MAC bruker en mapping-funksjon for å representere vilkårlig data med en fast lengde og deretter hasher meldingen. MAC sikrer dermed at

[5]ITU-T X.509 er en standard som definerer strukturen i et sertifikat. Se http://www.itu.int/home/index.html

data ikke har blitt endret på overføringen. HMAC er ganske lik MAC, men den bruker i tillegg en delt nøkkel til hash algoritmen. Den delte nøkkelen legges til data som skal hashes. Dette gjør HMAC sikrere fordi begge parter må ha den samme delte nøkkelen for at overført data skal være autentisert. TLS bruker HMAC mens SSL bruker MAC.

3.2.5 "Record Layer"

Protokollen på "record layer" mottar og krypterer data fra applikasjonslaget og leverer det til transportlaget. Record protokollen fragmenterer data til passende størrelse slik at det passer i krypteringsalgoritmen, komprimerer eller dekomprimerer, legger til en MAC eller HMAC og deretter krypterer eller dekrypterer dataene ved å bruke informasjonen som ble forhandlet frem i "Handshake"-protokollen



Figur 11: SSL data flow

3.3 IPSec

Internet Protocol Security (IPSec) tilbyr krypteringsbeskyttelse for IP datagram i IPv4 og IPv6 pakker. IPSec utføres inne i IP modulen ved å gi operativsystemet beskjed om å velge de sikkerhetsprotokoller som trengs, velge de algoritmene som trengs for å utføre tjenesten og framskaffe krypteringsnøkler som trengs av tjenestene. De forskjellige tjenestene er:

- Tilgangskontroll
- Kryptering på pakkenivå (Connectionless integrity)
- Autentisering av opprinnelse
- Delvis sekvensintegritet (replay beskyttelse)
- Datakryptering
- Dataflytbeskyttelse

Det er 2 protokoller som benyttes for å tilby sikkerhet i IPSec. Den ene er Autentication Header (AH) som er en ren autentiseringsprotokoll og den andre er Encapsulating Security Payload (ESP) som er en kombinert krypterings og autentiseringsprotokoll.

	AH	ESP (bare kryptering)	ESP (kryptering + autentisering
Aksess kontroll	x	x	x
Kryptering på pakkenivå	x		x
Opprinnelses Autentisering	x		x
Delvis sekvens integritet	x	x	x
Datakryptering		x	x
Dataflytbeskyttelse		x	x

Tabell 2: Oversikt over hvilke tjenester IPSec protokollene kan tilby.

3.3.1 Security Associations

IPSec har en abstrakt idé som kalles Security Associations (SA) som er en opptegnelse over hvordan en forbindelse er kodet for en en-veis IPSec kommunikasjon. Derfor trengs det 2 SA for å oppnå både inn og utgående trafikk. En SA identifiseres ved hjelp av tre parametre:

- Security Parameter Index (SPI)
  Dette er en bit-string tilordnet denne SA. SPI transporteres i AH/ESP hodet til mottakeren som ut fra denne velger hvilken SA pakken skal prosesseres under.
- Mottakers IP-adresse
- Security Protocol Identifier
  Denne viser hvilken av AH og ESP som brukes

3.3.2 SA Parametre

Hver IPSec implementering har en Security Association Database som definerer parametrene i hver SA oppkobling. En SA defineres med følgende parametre:

- Sequence Number Counter
  En 32 bits verdi som genererer "sekvens nummer" feltet i AH eller ESP-hodet
- Sequence Counter Overflow
  Et flagg som indikerer om overflow i Sequence Number Counter skal generere en feilmelding og stoppe kommunikasjonen til denne SA.
- Anti-Replay Window
  Brukes til å finne ut om en innkommende AH eller ESP pakke er en gjentagelse.
- AH informasjon
  Autentiserings algoritme, nøkler, nøklenes tidsbegrensning og lignende parametre som brukes av AH.
- ESP informasjon
  Krypterings og autentiserings algoritmer, nøkler, initierings verdier, nøklenes tidsbegrensning og lignende parametre som brukes i ESP.
- Livstid for denne Security Association
  Et tidsintervall eller teller som avgjør hvor lenge denne SA varer før den må byttes.
- IPSec Protocol Mode
  Tunnell, transport eller tilfeldig.
- Path MTU
  Den maksimale størrelsen på en pakke som kan sendes uten fragmentering og aldringsvariabler for pakkene.

3.3.3 Security Policy Database

Hvis en maskin kommuniserer med flere parter samtidig der ikke alle kjører IPSec kan man komme opp i situasjonen der bare noe av kommunikasjonen skal gjennom IPSec. Måten IP-trafikk er knyttet til en spesifikk SA på (eller ingen SA i de tilfeller der trafikken ikke skal gjennom IPSec) er et innslag i Security Policy Databasen (SPD). I sin enkleste form inneholder SPD innslag som definerer et subsett av IP-trafikk og en del "øvre-lag"-protokoll verdier som kalles *selectors* som vil gi en peker til en SA for denne trafikken. Utgående trafikk følger denne sekvensen for hver IP pakke:

1. Sammenlign verdiene på selector-feltene i pakken opp mot SPD for å finne et innslag som peker til en SA.
2. Finn SA for denne pakken og SPI'en som er assosiert med denne.
3. Utfør IPSec prosesseringen (AH eller ESP prosessering)

Følgende selector'er inngår i et SPD innslag: [Stallings, 2003, kapittel 16.2 side 489]

- IP-adressen til mottakeren
- IP-adressen til senderen
- Bruker ID
- Sensitivitetsnivået
- Transportlagsprotokollen (IPv4. IPv6)
- IPSec Protokoll (AH, ESP)
- Sender og mottakers port (TCP, UDP)
- IPv6 klasse
- IPv6 Flow Label
- IPv4 TOS

3.3.4 Transport og tunnel modus

Både AH og ESP kan brukes på to forskjellig måter: transport og tunnel. Forskjellen på disse er at i transportmodus vil IPSec operere på selve IP-pakken slik den kommer til IP-laget mens den i tunnelmodus vil pakke inn IP-pakken i en ny "ytre" pakke.

	Transport Modus	Tunnell Modus
AH	Autentiserer IP-nyttelast og utvalgte deler av IP-hodet.	Autentiserer hele "indre" IP-pakke samt utvalgte deler av den "ytre"
ESP	Krypterer IP-nyttelast	Krypterer "indre" IP-pakke
ESP med autentisering	Krypterer IP-nyttelast og autentisering av IP-nyttelast, men ikke IP-hodet.	Krypterer "indre IP-pakke og autentiserer den.

Tabell 3: Oversikt over IPSec i transport/tunnel modus

3.3.5 Nøkkelhåndtering

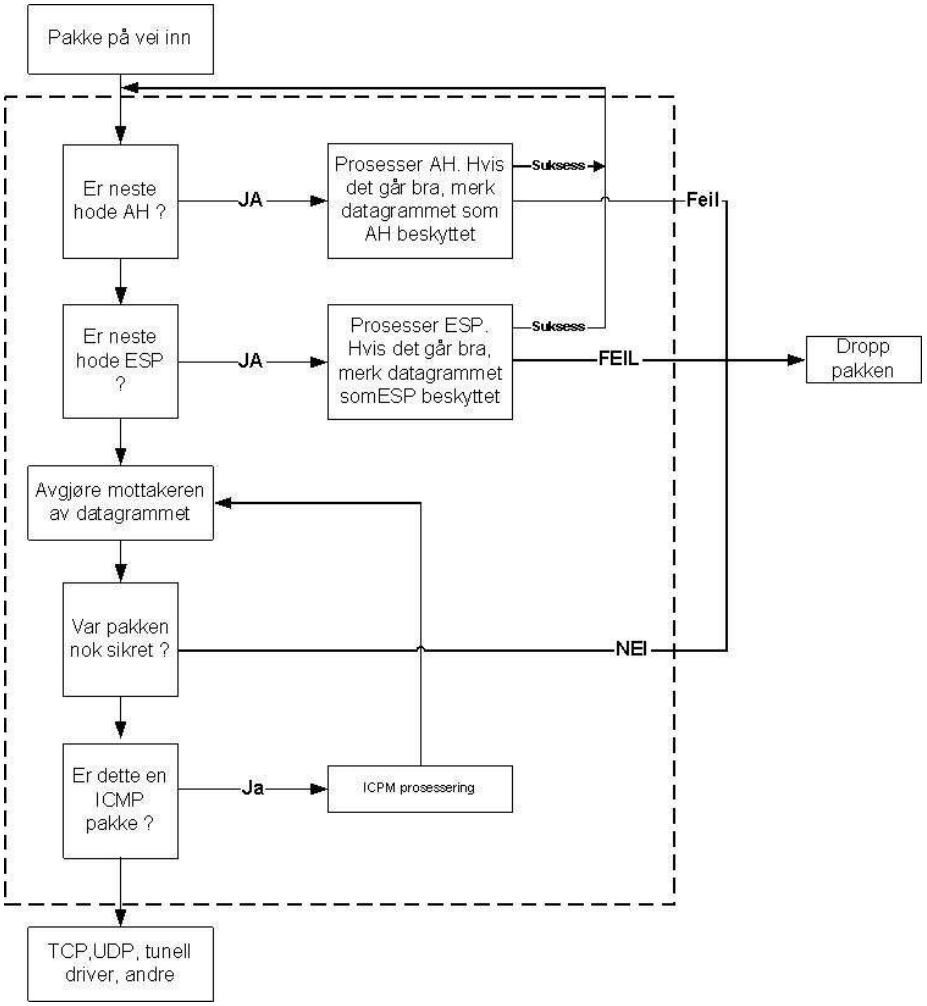
Nøkkelhåndteringen i IPSec involverer utvelgelse og distribusjon av hemmelige nøkler. IPsec arkitekturen gir mulighet for to typer nøkkelhåndtering:

- Manuell
  Dette betyr at en systemadministrator manuelt konfigurerer hvert system med sine egne nøkler samt nøklene fra de andre systemene en skal kommunisere med. Dette er mest praktisk for små, statiske systemer.
- Automatisk
  En automatisk nøkkelhåndtering vil muliggjøre å generere nøkler på forespørsel fra SA'en og gjøre det forenkle bruken av nøkler i et stort dynamisk system.

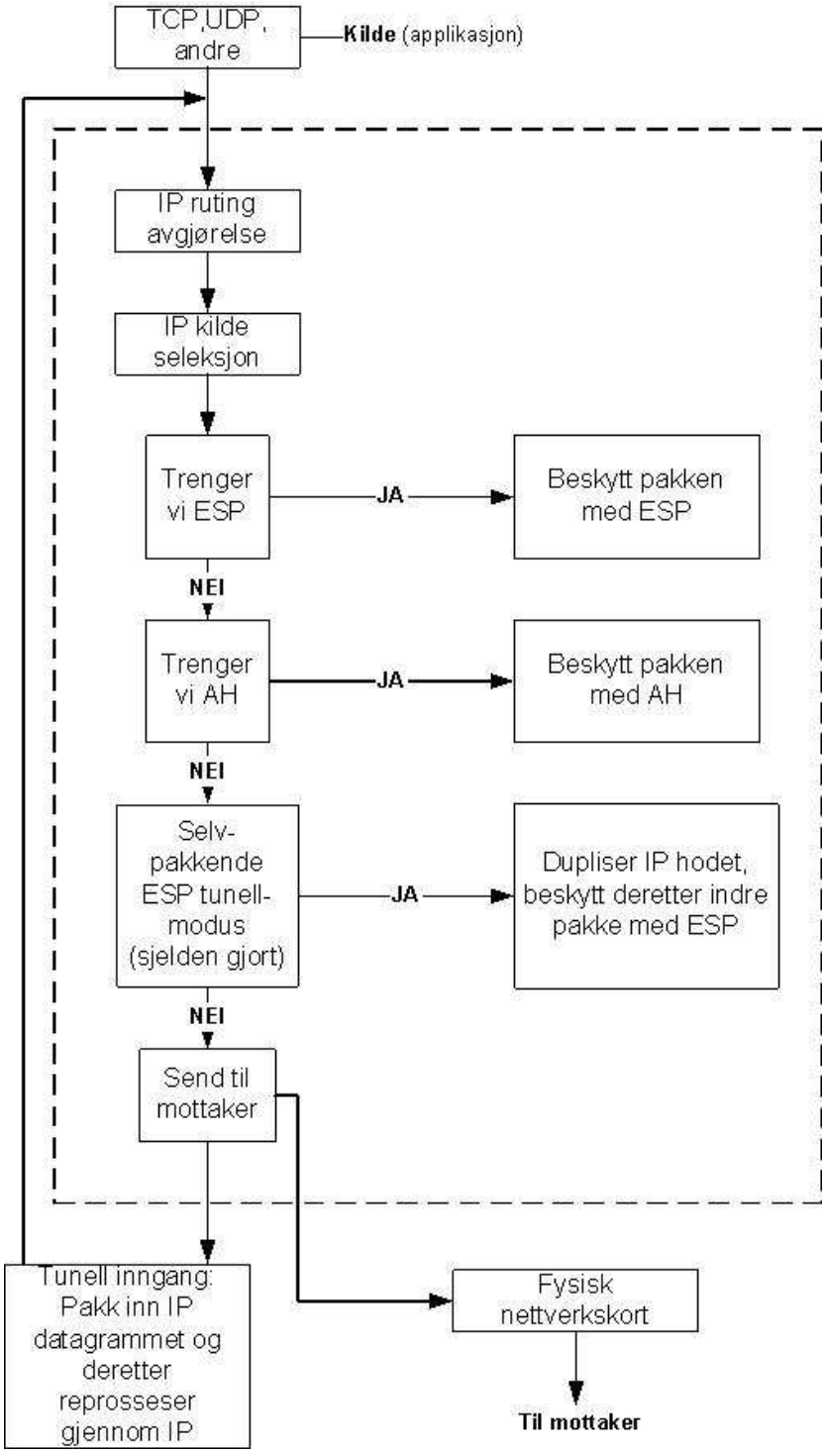
Den automatisk nøkkelhåndteringen i IPSec kalles ISAKMP/Oakley og består av følgende:

- Oakley Key Determination Protocol
  Oakley er en nøkkel utvekslingsprotokoll basert på Diffie-Hellmann[Stallings, 2003, side 293-296] algoritmen, men med økt sikkerhet.
- Internet Security Association and Key Management Protocol (ISAKMP)
  Denne gir et rammeverk for nøkkelhåndtering via internett.

3.3.6 Dataflyt med IPSec



Figur 12: IPSec dataflyt for innkommende pakker



Figur 13: IPSec dataflyt for utgående pakker

4 Datasikkerhet

Når vi her snakker om data er dette programmer, filer eller annen informasjon som lagres, kommuniseres med eller prosesseres av en datamaskin. For å sikre dataene til en organisasjon er det i hovedsak 5 hovedpunkter en bør konsentrere seg om. Disse er:

1. Data Backup
2. Virusforsvar
3. Brannvegg
4. ID og Passordbeskyttelse
5. Overvåkningsfil (Audit trail)

4.1 Data Backup

Å ta backup betyr at en kopierer data over på et annet medium, som for eksempel magnetbånd, andre disker eller DVD-ROM, slik at man har en sikkerhetskopi i tilfelle det primære mediet feiler slik at data mistes eller blir korrupt. Det er mange konsekvenser med ikke å ta backup, men de viktigste er:

- En kan miste data som ikke kan erstattes eller vil ta lang tid å gjenskape.
- Det kan forårsake problemer for andre mennesker. For eksempel brukere av systemet eller kunder.
- En kan miste troverdighet

Enhver organisasjon som tilbyr en tjeneste bør ha satt opp en egen Backup Plan. Dette er en oversikt som blant annet forteller hvilke data det må tas backup av og hvor ofte det skal gjøres. For å sette opp en slik plan må man vurdere følgende:

1. Hvor viktige data er det på systemet og hvilke konsekvenser vil det ha om de går tapt ?
   Dette vil gi en god pekepinn på om man trenger gode backuprutiner og hvor ofte det skal gjøres. For kritiske deler av systemet trengs det kanskje en grundig backup med lang lagringstid slik at man kan hente inn data fra langt tilbake i tid, mens det for mindre kritiske deler kun er nødvendig med daglig/ukentlig backup med mye kortere lagring.
2. Hvor ofte forandres informasjonen på systemet ?
   Dess høyere frekvens på forandringen, dess oftere må det tas backup. For eksempel så må data som forandres daglig også tas backup av daglig.

3. Hvor raskt er det nødvendig å få lagt tilbake backup ?
   Dette er et ganske viktig punkt for planleggingen av en backup plan. For kritiske system kan det være nødvendig å få tilbake dataene veldig raskt, noe som vil kreve at backup er lett tilgjengelig og ikke lagret langt fra systemet.

4. Har du utstyr til å utføre backup ?
   En må ha godt utstyr for å utføre god backup. Det trengs ofte mer enn en backup maskinvare for å utføre backup på systemer med mye data, samt flere sett med lagringsmedia. Det mest vanlige lagringsmedia er magnetbånd som er billigere, men også tregere når det gjelder overføringshastighet, enn de andre alternativene.

5. Trengs det fjernlagring av backup ?
   Fjernlagring av backup brukes som oftest for å sikre seg mot større katastrofer som vil ødelegge systemet fullstendig, som for eksempel oversvømmelse o.l.

6. Trengs det spesielle tilpasninger for å få tatt backup ?
   Kritiske applikasjoner kjører gjerne 24x7 og en må ofte bruke spesielle metoder for å få tatt en nøyaktig backup.

Ved å vurdere disse punktene kan man sette opp en backup plan for systemet. Hvis så denne planen blir satt i drift kan man delvis sikre seg mot at uerstattelig informasjon går tapt samt at man kan redusere nedetiden på systemet.

4.2 Virusforsvar

Et datavirus er et lite program som er utviklet for å utfører uønskede operasjoner på datamaskinen. Det kan angripe og infisere programfiler på maskinen, ødelegge datafiler eller "stjele" ressurser fra maskiner, som for eksempel båndbredde. Et datavirus er ingen trussel før det aktiveres. Det vil si at programmet ikke vil kunne utføre noen operasjoner før det blir prosessert. Virusene kan ligge skjult i datafiler eller programmer som, når disse blir eksekvert, vil gi viruset muligheten til å utføre det den er skrevet for å gjøre. En annen måte virus kan eksekveres på er at de legger seg "Master Boot Record"[6] og kjøres når maskinen startes eller "System Boot Sector"[7] og kjøres når maskinen starter operativsystemet.

For å være et datavirus må et dataprogram oppfylle disse kriteriene:

- Viruset overføres ved at det formerer/kopierer seg selv.

[6] "Master Boot Record" (MBR) er et lit program som kjøres når maskinen starter opp.
[7] "Sytem Boot Sector" er det stedet på disken der operativsystemet først lastes inn fra.

- Virus krever en vert for å spre seg. Dette kan være en fil på en datamaskin, et dokument, Master Boot Record eller System Boot Sector
- Virus fører til en eller annen utilsiktet handling fra brukerens side. Dette kan for eksempel være sletting av filer, sending av en melding eller kun replikering av viruset.

Et datavirus består gjerne av fire deler: Kamuflasje, lunte, formeringsmotor og sprengladning.

- Kamuflasje
  Vanlige virus gjemmer seg gjerne i andre dataprogrammer. Makrovirus gjemmer seg ofte i epost, tekstdokumenter og regneark.
- Lunte/utløser
  Dersom et virus slår til med det samme det har infisert maskinen vil det være forholdsvis lett og begrense spredningen av viruset. Men om det derimot tar lang tid, har lang og uregelmessig inkubasjonstid, vil det gjøre mye mer skade.
- Formering
  Når et virus har blitt aktivisert formerer det seg selv. Deretter kan det enten spre seg selv ved å søke kontakt med andre maskiner selv, eller det kan legge seg inn i programmer/filer for så å bli spredd utilsiktet av brukeren via epost eller lignende.
- Sprengladning
  Enkelte virus kan være forholdsvis harmløse og mangle sprengladning, mens andre kan gjøre ubotelig skade.

De vanligste typene av datavirus er:

1. Makrovirus
   Makrovirus bruker kommandoer (makroer) innebygd i programmer til å infisere og spre seg til andre filer som blir åpnet av de samme programmene. Eksempler på dette kan være Word og Eudora som begge har makroer som virus kan utnytte.
2. Ormer
   Ormer dupliserer seg selv og bruker kommunikasjonsverktøy som for eksempel en epost-klient til å spre seg videre. Disse kan for eksempel lese gjennom adresseboken til epost-klienten og sende seg selv til alle som står oppført.

3. Fil virus
   Disse virusene fester seg til andre programmer slik at når de blir kjørt laster viruset seg inn i minnet på datamaskinen slik at det kan infisere andre filer eller begynne å ødelegge datamaskinen.

4. Trojanere
   Disse virusene er programmer som utgir seg for å være noe annet. Når de så blir eksekvert kan de for eksempel infisere datamaskinen med et virus eller gjøre annen skade på maskinen.

5. Bakdørs trojanere
   Dette er virus som ligner på vanlige trojanere, men når de har blitt eksekvert så åpner de bakdører på datamaskinen slik at andre programmer kan komme seg inn. Det blir litt som å åpne vinduet på gløtt så andre innbruddstyver kan komme inn senere.

6. Boot sektor virus
   Dette er en gammel type virus og datamaskiner er lite utsatt for disse i dag. De legger inn programmer i datamaskinens boot sektor slik at disse aktiveres og blir aktive når maskinen startes.

4.3 Sikring mot virus

For å sikre seg mot virus er det viktig at alle maskiner kjøre AntiVirus programvare. Disse finnes både i gratis og kommersielle utgaver og er uansett en billig forsikring mot de skadene virus kan gjøre på datamaskinen.

Andre ting som kan gjøres er at epost-tjeneren som brukes til å sende og motta epost fra, kjører antivirus-programvare beregnet for epost-tjenere. På denne måten så skrelles virusene av de virusinfiserte epostene allerede i tjeneren. Dette gjør at de virusene som sprer seg via epost blir kraftig hemmet og dø ut.

I tillegg til dette så bør alle maskiner på en eller annen måte stå bak en brannvegg, enten via en stor maskinvare-brannvegg eller aller helst (eller i tillegg) en programvare-brannvegg på datamaskinen. På denne måten hindrer man virus å trenge inn på maskinen på måter antivirus-programvaren ikke har kontroll på.

4.4 Brannvegg

Bruken av internett har forandret seg en del i løpet av de siste årene. Når bredbånd og trådløse nett kom inn for fullt, ble det mer og mer vanlig å ha 24 timers tilknytning til internett. Dette gjør at mange maskiner er mer utsatt for uønsket inntrengning nå enn tidligere. For å holde uønskede besøkende og

"identitets pirater" borte, samt å sikre forretningens informasjon er det viktigere enn noensinne at maskinene står bake en robust brannvegg.

En brannvegg er et beskyttende system som ligger mellom datamaskinen og internett. Den har i oppgave og beskytte mot uautorisert bruk og tilgang til datamaskinen. Den analyserer trafikken som går inn og ut fra datamaskinen basert på en konfigurasjon. En brannvegg kan enten være maskinvare eller programvare, eller en kombinasjon av begge.

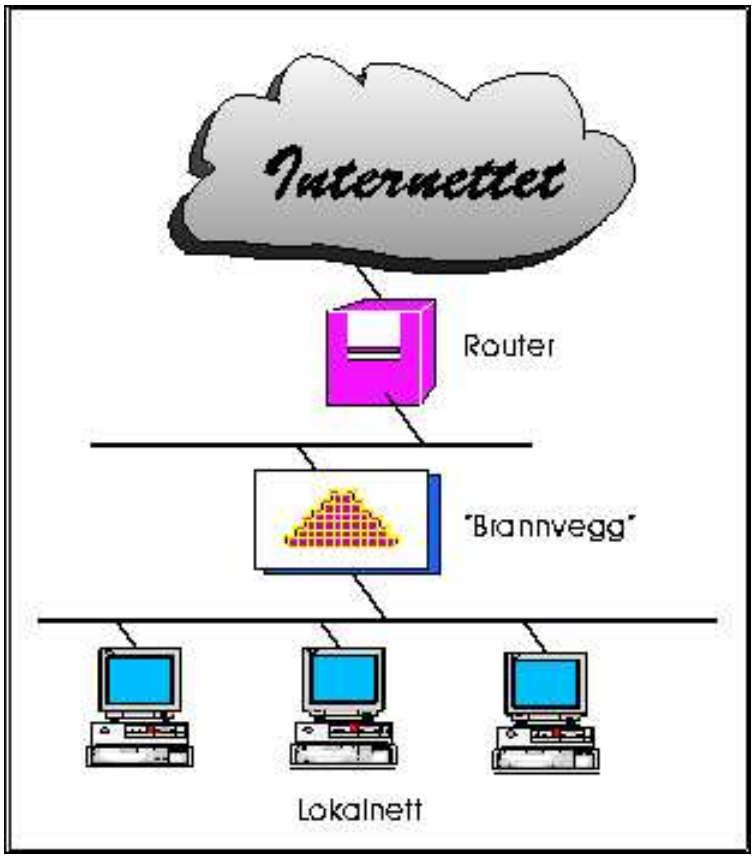
En maskinvare-brannvegg brukes for det meste til å beskytte et intranett med flere datamaskiner innenfor. Det vil si at brannveggen deles av alle maskinene som står på dette intranettet. Det finnes flere typer teknikker som brukes av maskinvare-brannvegger:

- Pakke filter.
  Hodene på IP-pakkene blir eksaminert for å finne mottaker og sender. Denne informasjonen blir så sammenlignet med ett sett definerte regler for å finne ut om pakken skal sendes eller forkastes. Pakke filter er effektivt og transparent for brukere, men det er vanskelig å konfigurere.

- Applikasjon gateway
  Legger til sikkerhetsmekanismer til spesielle applikasjoner som for eksempel FTP og Telnet servere. Dette er veldig effektivt, men kan gi en ytelsesdegradering.

- Circuit-level gateway
  Legger til sikkerhetsmekanismer når en UDP eller TCP forbindelse er opprettet. Når forbindelsen først er opprettet kan trafikken fortsette uten noe mer sjekking.

- Proxy server
  Fanger opp alle pakker til eller fra nettverket. Proxy tjeneren gjemmer den egentlige IP-adressen.



Figur 14: Brannvegg

Programvare-brannvegger er mer brukt av individuelle brukere som ikke nødvendigvis sitter på et intranett. Disse brannveggene er installert på datamaskinen og kan konfigureres for å passe for de oppgavene datamaskinen skal gjøre. Problemet med programvare-brannvegger er at brannveggen må ta imot pakkene på maskinen for å kunne behandle dem. Det vil si at de allerede er inne på maskinen, om enn i karantene, og kan potensielt gjøre skade.

Det aller beste er selvsagt å sitte bak en maskinvare-brannvegg og samtidig kjøre en programvare-brannvegg. På denne måten har man dobbel beskyttelse og det skal mye til for at noen kan få tilgang til maskinen.

4.5 ID og passordbeskyttelse

Passord er en veldig vanlig form for autentisering og er ofte den eneste barrièren mellom en bruker og personlig informasjon. Uvedkommende kan benytte seg av programvare for å gjette eller knekke passord, men ved å ha en streng passord

politikk kan dette gjøres tryggere[8]:

- Ikke bruk passord som er basert på personlig informasjon som andre kan ha tilgang til. For eksempel fødselsdag eller postnummer
- Ikke bruk ord som finnes i ordbøker
- Lag huskeregler for komplekse passord
- Bruk både store og små bokstaver
- Bruk en kombinasjon av bokstaver, tall og spesialtegn
- Bruk forskjellige passord på forskjellige systemer
- Ikke skriv ned passordet
- Lær å gjenkjenne "social engineering" der noen prøver å få deg til å oppgi passord eller annen informasjon
- Bytt passord ofte. Minst hver tredje måned.

De fleste passordsystemer har innstillinger for å sjekke vanskeligheten av passord samt hvor lenge siden det er byttet, og kan ut fra dette gi brukeren beskjed om at passordet er i orden eller må byttes.

Et system kan autentisere deg ut fra tre ting:

- Hva du vet
- Hva du har
- Hva du er

Ikke alle systemer bruker alle disse tre tingene. Standard passordbeskyttelse for eksempel, benytter kun en av disse (hva du vet). Andre løsninger begynner å komme for å sikre kritiske datasystemer som nettbanker og lignende der det tas i bruk flere av disse. Et eksempel på dette er det som kalles et "Token". Tokenet består av en elektronisk apparat (hva du har) som må parres med et passord (hva du vet) eller en biometrisk informasjon (hva du er) for å gi en sterkere autentisering.

[8]Hentet fra National Cyber Alert System "Cyber Security Tip ST04-002" http://www.us-cert.gov/cas/tips/ST04-002.html

4.6 Overvåkningsfil (Audit trail)

En overvåkningsfil er et monitoreringssytem som kan logge hendelser fra datamaskinen, operativsystemet, programmer eller brukere. Audit trail kan brukes til flere ting på et datasystem:

- Individuell ansvarliggjørelse
  Brukerens handlinger blir logget slik at de holdes ansvarlige for sine handlinger. Dette avskrekker brukerne fra å prøve å omgå sikkerheten på systemet.
- Rekonstruere hendelser
  Overvåkningsfiler kan også brukes til å rekonstruere hendelser etter at et problem har oppstått. Den skaden problemet har forårsaket kan bedømmes ved å finne ut nøyaktig hvordan, når og hvorfor det oppsto.
- Problem monitorering
  Overvåkningsfiler kan også brukes til sanntids verktøy for å monitorere problemer når de oppstår, som for eksempel diskfeil eller nettverksfeil.
- Oppdage innbrudd
  Overvåkningsfiler kan også brukes til å monitorere innbruddsforsøk mot datamaskinen.

Ved å ha et komplett Audit Trail oppsett på datamaskinen kan man sikre seg mot at uønskede hendelser skjer uten at man oppdager det. Det kan også brukes til å rekonstruere hendelser slik at en kan skaffe en oversikt over hvilke filer og konfigurasjoner en må hente inn igjen fra backup for å få maskinen tilbake til full operativ stand.

5 WEB sikkerhet

Det vi i dag kjenner som Verdensveven (World Wide Web) er essensielt en standard klient/server applikasjoner som kjører over internett via TCP/IP. Internett er ikke et sikkert kommuniksjonsmedium og dette medfører en del utfordringer med tanke på sikkerhet.

- Internett er to-veis. Dette betyr at i motsetning til tradisjonelle metoder for å publisere informasjon, som for eksempel bøker, kan denne metoden utsatt for for angrep som vil forandre eller slette informasjonen.
- Internett er blitt en populær metode for bedrifter å markedsføre seg og drive forretning. Ved innbrudd på web-serverene kan troverdigheten til selskapet ødelegges og penger gå tapt.
- Den underliggende programvaren på web-servere er så kompleks at den kan skjule sikkerhetsfeil som er nesten umulig å oppdage for de som installerer/vedlikeholder den. Dette kan føre til innbrudd på serveren.
- Om noen bryter seg inn på web-server kan det hende at de dermed får tilgang til resten av bedriftens nettverk. Med andre ord så brukes serveren som et springbrett for å få tilgang videre inn i systemet
- Uvitenhet blant brukere og web-server administratorer kan føre til store sikkerhetsrisikoer ved at de benytter seg av usikker programvare og ikke er nøye nok med sikkerheten rundt autentisering.

5.1 Sikkerhets trusler

	Trussel	Konsekvens	Mottiltak
Integritet	* Modifisering av brukerdata * Trojanere får tilgang til data * Forandring av minne * Forandring av kommunikasjonsmeldinger	* Tap av data * Kompromittering av maskinen * Åpen fo alle andre trussler	* Kryptografiske sjekksummer
Konfidensialitet	* Lytting på nettet * Tyveri av informasjon på serveren * Tyveri av data fra klient * Informasjon om nettverkstopologi * Informasjon om hvilken klient som kommuniserer mot serveren	* Tap av informasjon * Tap av "privacy"	* Kryptering * Web-proxy
Denial of Service	* Kommunikasjons - avbrudd * Overbelaster maskinen med falsk informasjon * Fyller opp diskplass eller minne * Isolerer maskinen ved å angripe DNS	* Forstyrrende * Irriterende * Hindrer arbeid	* Vanskelig å hindre
Autentisering	* Utgi seg for legale brukere * Data forfalskning	* "Misrepresentation" av brukeren * Falsk informasjon tas som ekte	* Kryptering

Tabell 4: Web sikkerhet trusler,konsekvenser og mottiltak

Tabell 3 gir et sammendrag av de forskjellige sikkerhetstruslene som vi møter via Web.

En kan gruppere truslene mot en web-server på følgende måter:

1. Webserveren
2. Nettleseren

3. Trafikken mellom nettleseren og webserveren

5.1.1 Sikring av webserveren

En webserver som står koblet til internett er tilgjengelig for alle andre som er koblet til internett. Dette gjelder også de som er ute etter å skaffe seg adgang til andres informasjon og rettigheter, såkalte angripere (hackere). Det som kan bli angrepet på en webserver er:

- Webserverens operativsystem
- Selve Webserver programvaren
- Programmer, script eller plug-ins som webserveren bruker
- Brannvegger og rutere som skiller webserveren fra organisasjonens interne nettverk

Angripernes mål kan være:

- Skaffe direkte adgang til webserveren
- Forandre informasjonen på webserveren
- Hindre andre å få tilgang til webserveren

For å forhindre slike angrep må man utarbeide en plan for å vedlikeholde webserveren slik at denne kan motstå angrep. Dette gjøres ved å:

- Holde webserveren oppdatert.
  Dette innebærer å jevnlig sjekke informasjon om sårbarheter og hendelser vedrørende webserveren en kjører. Det finnes mange organisasjoner som forsker på websikkerhet og som publiserer informasjon med jevne mellomrom. Dette inkluderer både webserveren og eksterne programmer denne bruker, som for eksempel script-språk.
- Oppdatering av sikkerhetskontroller
  I tillegg til å holde webserver programvaren oppdatert må en også.
  - Oppdatere oppdagelsesmekanismer slik at de kan oppdage de nye sårbarhetene.
  - Hindre angrep ved å oppdatere brannveggen.
  - Midlertidig slå av tjenester som er utsatt for angrep.
- Samarbeid mellom administrator og nettleverandør.
  For å forhindre angrep er det viktig at administratoren av webserveren samarbeider med nettverksleverandøren for å forhindre angrep. Dette gjelder spesielt såkalte Deniel of Service angrep (DOS).

5.1.2 Sikring av nettleseren

Sikkerhetsrisikoene for en nettleser kan grupperes inn i følgende kategorier:

- Webserveren er ikke nødvendigvis sikret.
  All data en bruker sender fra seg via en nettleser blir til slutt prosessert av en webserver og som oftest lagret i en database. De fleste brukere antar at en profesjonell organisasjon som tilbyr en tjeneste har sikret webserveren, men dette er ikke nødvendigvis tilfelle. Det beste forsvaret er å sende fra seg så lite sensitiv informasjon som mulig.

- Nettleseren kjører farlige script eller programmer.
  En nettleser kan automatisk kjøre script eller eksterne programmer for å vise frem informasjon. Men den er samtidig sårbar for de samme mulighetene. Dersom et script blir forandret av ondsinnede personer, kan dette føre til at nettleseren kjører script som kan ødelegge eller kompromittert sensitive data.

- En angriper kan lytte på trafikken mellom nettleser og webserver.
  Brukere må være oppmerksomme på at sikkerheten til data som blir sendt fra nettleser til webserver ikke er sikrere enn nettverket den sendes over. Sikkerheten kan økes ved at partene bruker SSL for å kryptere/dekryptere trafikken.

- En hacker kan kjøre et man-in-the-middle-attack.
  Sesjonsløse web-baserte applikasjoner. som en webserver, er potensielle ofre for kapring og gjentakelse (hijacking og replay). Sesjonskapring består i at en angriper lytter på nettverket og overta en forbindelse mellom en webserver og nettleser. Dette er mulig siden denne type applikasjoner ikke er bundet sammen i en sesjon. Angriperen lurer deretter webserveren til og sende/motta data til angriperen istedet for brukeren.
  Et gjentakelsesangrep består i at en angriper registrerer autentisert trafikk mellom en webserver og nettleser. Informasjonen blir deretter modifisert og sendt til webserveren. For eksempel: Du er innlogget i en nettbank og flytter over penger fra en konto til en annen bank. Angriperen kopierer informasjonen du sender til nettbanken og setter inn sitt eget kontonummer der pengene skal til og sender dette så til webserveren.

Proxy

En metode for å sikre nettleseren på er å ta i bruk en sikker proxy server. Dette er en spesiell programvare som kjører på en maskin og fungerer som en innfallsport (gateway) til internett for en eller flere protokoller, deriblant web. All trafikk

nettleseren sender ut på internett må gå gjennom denne web proxyserveren. Ved å bruke en sikret proxyserver kan en oppnå en rekke fordeler:

- En del av sikkerhetsinnstillingene kan flyttes fra nettleseren til proxyserveren.
  Det er mye enklere for en nettverksadministrator å vedlikeholde ett sett med innstillinger på proxyserveren enn kanskje hundrevis av individuelle nettlesere.
- Sikkerhetsinnstillingene på proxyserveren vil virke for alle typer nettlesere.
  Alle moderne nettlesere støtter bruken av web proxy og dermed kan nettverksadministratoren for eksempel nekte kjøring av VisualBasic-script, som er kjent for å være en sikkerhetsrisiko, i proxyserveren istedet for å måtte finne ut hvordan dette må gjøres på hver enkelt type nettleser.
- Bruk av proxyserver kan øke hastigheten for brukerne ved at den lagrer et buffer (cacher) med de oftest benyttede websidene.
- Proxyservere kan være veldig nyttige for barn ved at de gjør det mulig å legge restriksjoner på oppslag mot enkelte sider.

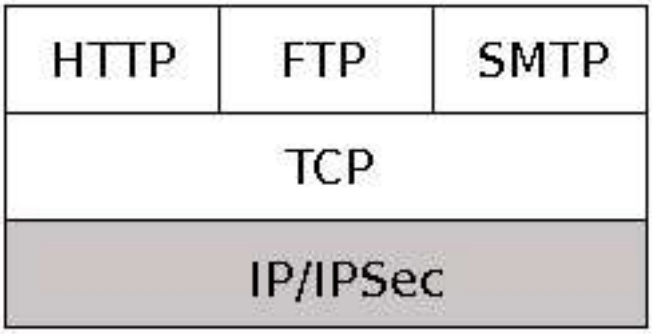
Å sikre en nettleser fullt og helt er en meget vanskelig oppgave. Hvis en holder nettleseren oppdatert til enhver tid og stiller inn sikkerheten på maksimum, kommer en en fort opp i tilgjengelighet versus sikkerhets-dilemmaet. Med sikkerhetsinnstillingene på maksimum vil brukeren hele tiden måtte sitte å vurdere om han vil akseptere cookies, javaskript og andre ting som brukes av diverse websider. Men det er en rekke ting brukere kan gjøre for å sikre seg bedre:

- Holde nettleser og operativsystem oppdatert med hensyn på sikkerhetspatcher
- Unngå virus
- Bruke sikre websider for finansielle og andre sensitive transaksjoner.
- Bruke en sikker proxyserver.
- Sikre det lokale nettverket.
- Unngå å sende sensitiv informasjon over internett.
- Vær oppmerksom når en gjør forandringer på innstillingene til nettleser.

5.1.3 Sikring av trafikk mellom nettleser og webserver

Det finnes mange metoder for å sikre trafikken mellom nettleseren og webserveren. Metodene er forholdsvis like med hensyn på hva de tilbyr av sikkerhet og til en viss grad måten de oppnår dette på, men de skiller seg fra hverandre med hvor de opererer i TCP/IP og hvor lett de kan benyttes.

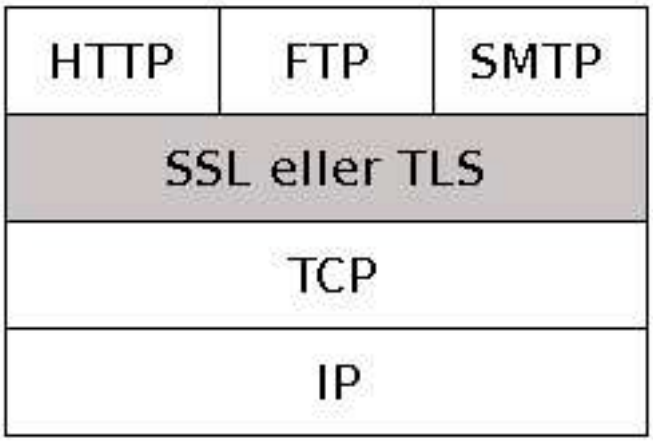
En metode for å gi websikkerhet er å benytte IPSec på trafikken mellom klient og server:



Figur 15: Sikkerhet på nettverksnivå.

Fordelen med denne metoden er at den er transparent for både brukeren og applikasjonene. I tillegg til dette er det en generell metode som gir relativt god sikkerhet på alle typer trafikk mellom klient og server. IPSec har også filtreringsmekanismer som gjør det mulig å skille trafikk der det er ønskelig med sikring og kun prosessere dette gjennom IPSec.

En annen metode er å implementere sikkerheten rett over TCP i TCP/IP-stakken. Dette gir også en relativt generell sikkerhet på all trafikken mellom klient og server. De mest brukte mekanismene for å implementere sikkerhet på dette nivået er SSL eller TLS.

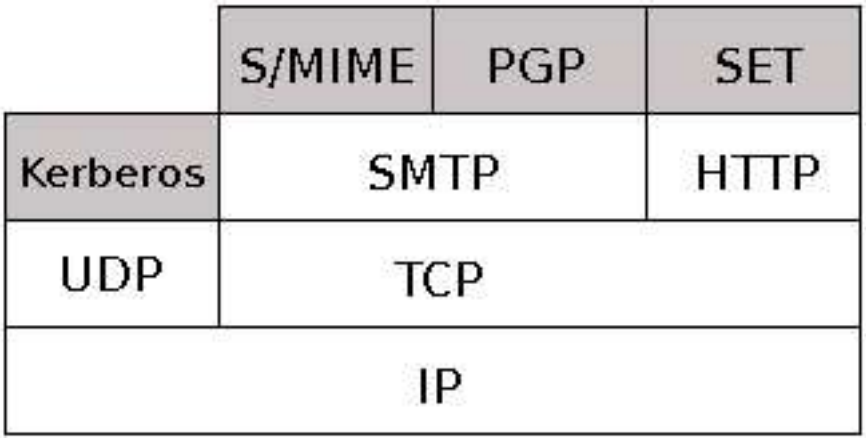


Figur 16: Sikkerhet på transportnivå

Det er igjen to forskjellige implementasjonsmuligheter for å sikre trafikken på dette nivået.

- For å få en generell sikring av trafikken kan SSL eller TLS implementeres i de underliggende protokollene på dette nivået og dermed bli transparent for bruker og applikasjoner.
- Alternativet er at SSL eller TLS støtte bygges inn i applikasjoner som kan arbeide på dette nivået. Eksempel på dette kan være at de fleste nettlesere og webservere kommer med innebygd støtte for SSL og/eller TLS.

Den tredje metoden lar applikasjonene som kjører på toppen av TCP/IP-stakken styre sikkerheten selv:



Figur 17: Sikkerhet på applikasjonsnivå

Her er det ingen underliggende sikkerhet og dette gir ingen generell sikkerhet på trafikken mellom klient og server. Hver applikasjon må styre sikkerheten selv. Fordelen med dette er at sikringen av trafikken kan spesialtilpasses hver enkelt applikasjons spesielle behov. Et eksempel på dette er Secure Electronic Transaction (SET) som er et komplekst sikkerhetssystem som ofte benyttes i betalingstjenester.

6 Digitale signaturer

Grunnlaget for begrepet digital signatur er det samme som vi tenker på når det gjelder å signere et papirdokument med penn. En signatur på papir vil gjøre det mulig å spore opp hvem som har signert og og dokumentet vil være bindende for denne personen. Overføring av dette til elektroniske dokumenter synes å være vanskelig. Med en pennesignatur på papir vil det være forholdsvis vanskelig å fjerne eller forfalske signaturen uten å ødelegge papiret den er skrevet på. Hvis en tenker på en elektronisk signatur på samme måte vil det være enkelt og forandre denne ved å editere signaturen på dokumentet og/eller innholdet i dokumentet. Løsningen på dette er å signere et dokument på en slik måte at det ikke er mulig å skille signaturen fra dokumentet og omvendt. Det vil si at signaturen genereres ut fra dokumentet og hvis dokumentet forandres vil også signaturen forandres. En digital signatur må ha følgende egenskaper:

- Den må verifisere forfatteren og tidspunktet signaturen er generert
- Den må autentisere innholdet da signaturen ble generert
- Det må være mulig å verifisere for en tredje-person i tilfelle det skulle oppstå konflikter

6.1 Kryptering versus digital signering

Det er lett å blande sammen begrepene signering og kryptering av elektroniske dokumenter. Kryptering brukes for å gjøre klartekst uforståelig for uvedkommende. Dette er kun nødvendig dersom dokumentets innhold skal holdes hemmelig. Signering er en rutine som sørger for å sikre at dokumentet er autentisk slik at innhold og identiteten til opphavet er kjent. Kryptering og digitale signaturer kan brukes sammen eller separat:

- Dokumentet er kryptert, men ikke digitalt signert:
  Bare de som kan dekryptere dokumentet kan lese det, men leseren kan ikke være sikker på hvem som skrev det.
- Dokumentet er digitalt signert, men ikke kryptert:
  Alle kan verifisere hvem som skrev det og alle kan lese det.
- Dokumentet er først kryptert, deretter digitalt signert
  Bare de som kan dekryptere dokumentet kan lese det, men alle kan verifisere hvem som skrev det.
- Dokumentet kan først signeres digitalt, deretter krypteres:
  Bare de som kan dekryptere dokumentet kan lese det, og leseren kan samtidig være sikker på hvem som skrev det.

6.2 Offentlig-nøkkel kryptering

Den første publiserte offentlig-nøkkel algoritmen ble lansert av to forskere, Diffie og Hellman, der offentlig-nøkkel kryptografi ble definert. Denne omtales som regel som "Diffie-Hellman nøkkel-utveksling". Denne formen for kryptering går ut på at en kan la krypteringsalgoritmen og en offentlig-nøkkel være kjent, mens man benytter en privat nøkkel for å kryptere/dekryptere meldinger. Denne formen for kryptering kalles asymmetrisk kryptering. Symmetrisk kryptering betyr at en bruker samme nøkkel til å kryptere og dekryptere et dokument. For å gi en kort oversikt hvordan offentlig-nøkkel kryptering fungerer i praksis så vil jeg her ta et eksempel. La oss tenke oss 2 brukere, Per og Pål:

1. Per har 2 elektroniske nøkler. En offentlig som alle har tilgang til og en privat som kun per kjenner.
2. Pål krypterer et dokument ved å bruke per's *offentlige* nøkkel og sender det så til per.
3. Per mottar det krypterte dokumentet og dekrypterer det med sin *private* nøkkel.

Offentlig-nøkkel kryptering løser mange av problemene som oppstår ved bruk av klassisk kryptering. En slipper problemet med nøkkelutveksling og en kan opprette sentrale nøkkelkataloger der de offentlige nøklene kan hentes.

6.3 Digital signatur modell

En typisk implementasjon av digitale signaturer skjer på følgende måte. La oss bruke Per og Pål igjen:

1. Per reduserer dokumentet han skal sende til en hashverdi ved hjelp av en hashalgoritme. Han vil da sitte igjen med noe som kalles en "message digest" . (Denne prosessen er en-veis noe som vil si at det er ikke mulig å gjenskape det orginale dokumentet ut fra denne hashverdien)
2. Per krypterer så denne hashverdien med sin *private* nøkkel. resultatet av denne operasjonen er en digital signatur. Hashverdien er *unik* for akkurat dette dokumentet og den er kryptert med per's *private* nøkkel.
3. Per sender så dokumentet til pål.
4. Det første pål gjør er å dekryptere den digitale signaturen med per's *offentlige* nøkkel. Hvis det går bra beviser dette at per sendte meldingen fordi han er den eneste som har den *private* nøkkelen.

5. Deretter genererer pål en hashverdi ut fra det samme dokumentet og sammenligner denne verdien med den han dekrypterte fra den digitale signaturen. Hvis disse to er like vet pål at dokumentet er nøyaktig likt det som den digitale signaturen er generert ut ifra og dermed autentisk. Hvis de ikke er like forkastes dokumentet.

7 E-post sikkerhet

E-post er sammen med websurfing og peer-2-peer applikasjoner, noe av det som har gjort internett veldig populært de seneste årene. Til tross for sin kritiske rolle er e-post relativt usikkert i forhold til andre applikasjoner som brukes på internett. E-post har en veldefinert og universalt implementert protokoll. Dette gjør at det er et yndet mål for angrep fra ondsinnede personer. Angrep på e-post fokuseres gjerne på to områder:

- Leveranser og kjøring av farlige script og programmer (malcode)
- Frigjøring av sensitiv informasjon

7.1 Angrep med malcode

E-Post, slik det er definert av Network Working Groups RFC, er implementert i standard ASCII tekst. ASCII tekst kan ikke eksekveres direkte, noe som er et problem for malcode, som trenger å bli kjørt eller kopiert for å gjøre skade. Hvis en for eksempel mottar en e-post med ordet "shutdown" inne i teksten vil ikke dette føre til at mailleseren eller operativsystemet stopper. Malcode tilbringer det meste av sitt liv i ren ASCII form, der den enten opptrer som et rent tekstskript eller som en kodet blokk.

Det følgende viser en e-post med en kodet binærfil som et vedlegg:

```
Subject: test
From: gorm.andersen@idi.ntnu.no
To: gorm.andersen@idi.ntnu.no
Content-Type: multipart/mixed; boundary''Innholdsgrense''
Mime-Version: 1.0
--Innholdsgrense
Content-Type: text/plain
Content-Transfer-Encoding: 8bit
Dette er en test med filen 'byebye.vbs'
Content-Type: application/octet-stream; name="byebye.vbs"
Content-Transfer-Encoding: base64
MjIyMjIyMjIyMjIyMjIyMjIyMjIyMjL/wAARCAEgAa4DASIA
--Innholdsgrense
```

Dette er en e-post slik den ser ut i rått format. Det som står på linjene mellom "*Subject*" og "*Mime-Version*" er selve hodet på e-posten. Dette definerer hva tillelen er (*Subject*), hvem e-posten er fra (*From*), hvem den skal til (*To*) og hva den inneholder (*Content-Type*). I innholdsforklaringen ser vi at denne e-posten er flerdelt med forskjellig innhold (*multipart/mixed*). Deretter defineres

en grense (*Innholdsgrense*) slik at e-postleseren skal vite hvordan den skal vise frem e-posten. Her er det to deler, en som er ren tekst som er kodet i 8 bits form (*Dette er en test med filen 'byebye.vbs'*) , og et binært vedlegg som er kodet med base64 koding (*MjIyMjIyMjIyMjIyMjIyMjIyMjIyMjL/wAARCAEgAa4DASIA).*

I dette eksemplet er binærfilen kodet med base64 koding[Stallings, 2003, Appendix 15B]. Hvis filen forblir urørt i sin kodede form er den helt harmløs. Den må dekodes og kjøres for å gjøre skade. Den kodede filen er vanligvis forkledd med et harmløst navn slik at brukeren blir narret til å åpne vedlegget med et program. Dermed får malcoden muligheten til å utføre ugjerninger.

Et eksempel på malcode av denne typen er W32/Novarg.A malcoden som herjet i 2004. (`http://www.cert.org/incident_notes/IN-2004-01.html`). Hvis denne malcoden ble eksekvert på en Windows maskin gjorde den følgende:

1. Modifiserte verdier i registeret på maskinen slik at viruset ble kjørt igjen ved reboot.
2. Åpnet en inngående TCP port på maskinen. (antageligvis for å gjøre det mulig å komme seg inn på maskinen utenfra)
3. Installerte en kopi av seg selv under: C:\Program Files\KaZaA\My Shared Folder\ slik at den var nedlastbar for andre KaZaA brukere.[9]

7.2 E-post angrep

All tekst som sendes via E-post er lett å fange opp og leses på IP-pakkenivå siden den sendes uten noen form for kryptering. Det er også en forholdsvis enkel jobb å fange opp IP-pakkene, forandre innholdet, og sende disse videre til E-postserveren uten at dette merkes av senderen eller mottakeren. Dette kan gjøres på to forskjellige måter: mann-i-midten angrep (man-in-the-middle attack) eller gjentagelse (replay)

7.2.1 Mann-i-midten (man-in-the-middle)

Ut ifra navnet, mann-i-midten, går det frem at angriperen må da ha tilgang til nettverk eller nettverksutstyr mellom senderen og mottakeren. Dette kan være brannvegger, svitsjer, E-posttjeneren eller gateway'en. Hvis angriperen sitter på samme nettverk er det mulig å bruke en Adress Resolution Protocol (ARP)[10] nettverkssniffer, som for eksempel etherreal[11], til å fange opp trafikk mellom en annen maskin og E-posttjeneren. Angriperen kan da også forandre innholdet

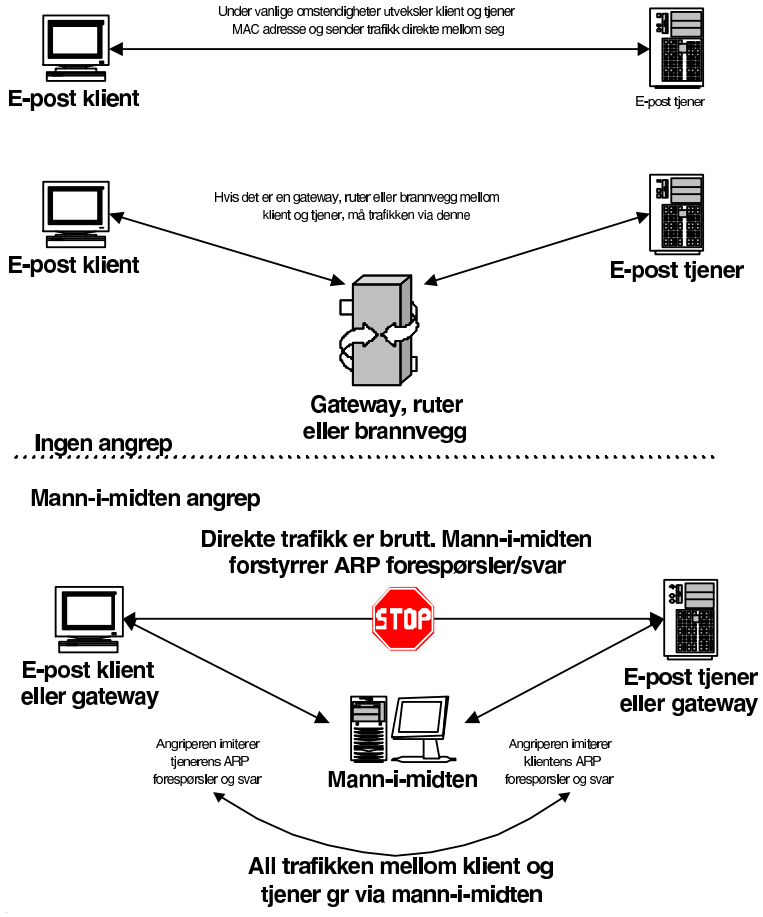
[9] KaZaA er et populært peer-to-peer fildelingsprogram. For mer info: http://www.kazaa.com/

[10] ARP er en TCP/IP protokoll som brukes til å konvertere IP-addresser til fysiske adresser (MAC = Media Access Control, som er en unik identifikasjon til hver enhet på nettverket)

[11] http://www.ethereal.com/

i E-posten før han sender den videre. ARP angrep kan skje på 4 forskjellige plasser:

1. Mellom E-post klienten og E-post tjeneren - Dette forutsetter at klienten og tjeneren er på samme lokalnettverk.
2. Mellom E-post klienten og gatewayen - Gatewayen må da være veien E-posten tar mellom klient og tjener.
3. Mellom de to gateway'ene E-posten går
4. Mellom gateway'en og E-post tjeneren.



Figur 18: Mann-i-midten angrep

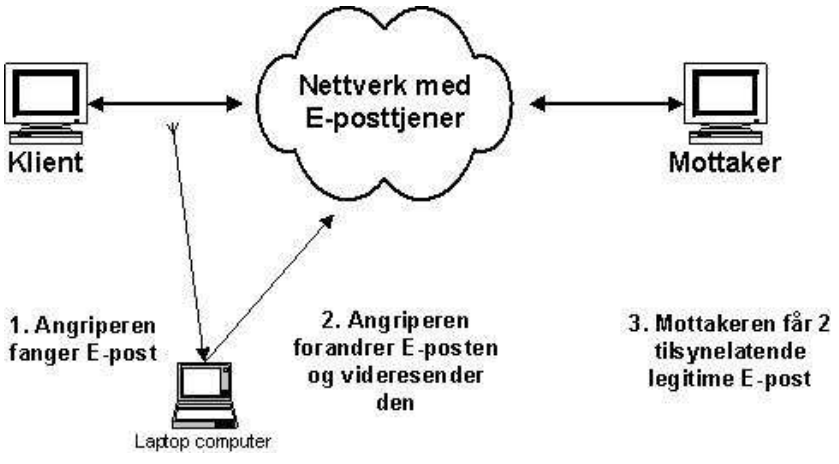
I et ARP mann-i-midten angrep vil E-post som går fra klienten bli fanget opp på IP-nivå på vei til tjeneren og lest/forandre innholdet. Den beste måten for å unngå slike angrep er å bruke kryptering og digitale signaturer. (se kapittel 6 på side 46 Hvis krypteringen er sterk nok vil ikke angriperen være i stand

til å dekryptere og lese/forandre innholdet i E-posten. Digitale signaturer sikrer integriteten i E-posten. Med andre ord: Den inneholder det den skal inneholde. E-posten signeres med senderens private nøkkel, noe som gir en ekstrakt som sendes sammen med E-posten. Denne kan så dekrypteres av mottakkeren med den offentlige nøkkelen og mottakeren kan da verifisere at innholdet er uforandret siden det ble signert. Et mann-i-midten angrep vil ikke kunne forandre innholdet i E-posten uten å bli oppdaget siden dekrypteringen da vil feile. (se kapittel 13.2.1 på side 112)

7.2.2 Gjentagelse (replay)

Et E-post gjentagelses angrep skjer når en E-post pakke blir fanget opp, innholdet forandret, og deretter sendt ut på nettverket igjen ved en senere anledning (gjentagelse) . Dette fører til at to tilsynelatende legitime E-post når mottakeren, men den siste kan ha forfalsket innhold. Faren med denne type angrep er at E-posten med forfalsket innhold blir tatt som legitim, noe som kan føre til alvorlige problemer. Tenk på følgende: En firma sender en E-post til en person og ber han betaler inn 100 kroner til kontonummer NNN for å få utført en tjeneste. En angriper fanger opp E-posten og forandre innholdet slik at det kommer en ny og tilsynelatende legitim mail til brukeren der det bes om å overføre 1000 kroner til kontonummer AAA (som da kanskje angriperen har tilgang til). Dette ser man kan skape problemer.

I gjentagelsesangrep trenger ikke angriperen å ha tilgang til lokalnettverket til klient eller tjener. Det holder at han har tilgang til et nettverk der E-posten blir sendt gjennom på vei til eller fra klienten.



Figur 19: Replayangrep i E-post

8 Sikkerhet i trådløse nettverk

Trådløse nettverk gir brukeren frihet og fleksibilitet, noe som har gjort dette veldig populært både blant private og profesjonelle brukere. Man regner med at det i 2006 finnes over 50 millioner trådløse enheter som kobler seg opp mot nettverk og dette tallet øker fortsatt.

Når man skal vurdere bruken av trådløs nettverksteknologi, blir det som regel en avveining mellom fleksibilitet og mobilitet på den ene siden og sikkerheten på den andre. Helt siden starten har trådløse nettverk vært plaget av sikkerhetshull og en del av disse problemene finnes fortsatt. Det finnes derimot løsninger som kan forbedre sikkerheten slik at den tilfredsstiller de fleste krav og jeg vil her gå igjennom en del av dem.

Hvor høy skal sikkerheten være før det er sikkert nok ? Dette er et spørsmål som i mange tilfeller er vanskelig å besvare. Det kommer i mange tilfeller an på hvilken type trafikk som sendes over nettverket, men som en generell regel kan man si at private hjemmenett har mindre behov for sikkerhet enn trådløse nett hos bedrifter. Derfor vi jeg skille mellom disse to når jeg her tar for meg sikkerhet i trådløse nett.

8.1 Trådløse hjemmenett

På private trådløse nett er det ofte begrenset hvor mye arbeid en legger ned i å bedre sikkerheten fordi dette vil gå ut over fleksibilitet og brukervennlighet. De aller fleste private nettverk kommer med en ferdig konfigurasjon og med det samme nettverket er satt opp og fungerer, vil de fleste slå seg til ro med dette og ikke gjøre mye mer for å bedre sikkerheten. Men det finnes en del mekanismer for enkelt å bedre sikkerheten og de fleste leverandører av trådløse enheter har begynt å konfigurere enhetene med litt høyere sikkerhet i den senere tid.

Trådløse nettverk benytter et nettverksnavn, SSID (Service Set Identifier), for å skille de ulike nettene fra hverandre. Disse navnene er ofte satt til standardnavn på enhetene fra produsentens side. Disse annonseres ut til de trådløse klientene slik at disse kan se og koble seg opp mot aksesspunktet. Ved å skru av annonseringen av dette nett-navnet og samtidig forandre det til noe annen enn fabrikkinstillingen, blir det med ett litt vanskeligere for en angriper å komme seg inn på nettverket.

En annen tjeneste de trådløse aksesspunktene ofte tilbyr er en dynamisk tildeling av IP-adresse for klientmaskiner som ønsker dette [12]. Dette fungerer slik at en klient som kontakter aksesspunktet sender MAC-adressen sin (Media

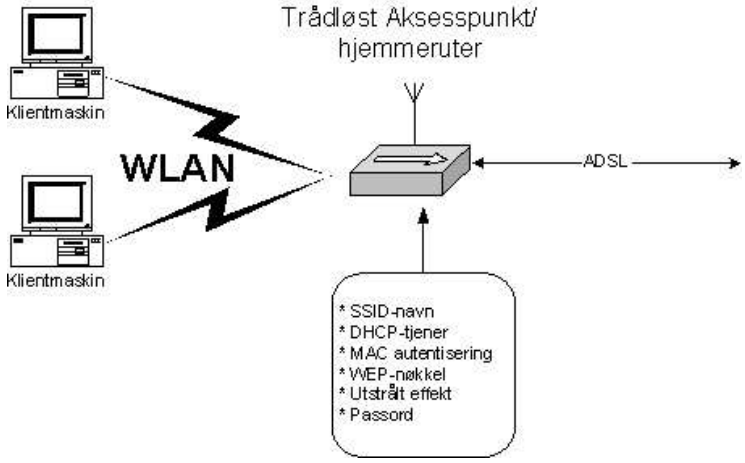
[12]Denne tjenesten kalles som regel DHCP (Dynamic Host Configuration Protocol) og gir klientmaskinen nettverksinformasjon den trenger for å fungere. Informasjonen som sendes ut inneholder blant annet: IP-addresse, nettmaske, router og navnetjenere.

Access Control), som er en unik identifikator for nettverkskortet i maskinen, og får en IP-adresse i retur fra aksesspunktet. De aller fleste aksesspunkt har muligheten til å sette opp en liste med MAC-adresser som får tilgang til nettverket. Hvis en slår av den automatiske IP tildeling og samtidig slår på MAC-autentisering samt legge inn alle de godkjente MAC-adressene i listen, vil man også gjøre det litt vanskeligere for en angriper.

Disse metodene er egentlig det vi kaller sikkerhet ved tåkelegging (security by obscurity) siden det er mulig for en angriper å sniffe på nettverket og fange opp både SSID og MAC-adresse på det trådløse nettverket. Angriperen kan deretter omprogrammere nettverkskortet sitt til å benytte den samme MAC-adressen som de godkjente adressene, og dermed få tilgang til nettverket. Det aller viktigste punktet i sikkerheten på hjemmenettverket er derfor å slå på den krypteringsmulighetene det trådløse aksesspunktet støtter. Dette vil ofte være WEP (Wired Equivalent Privacy) [13], som imidlertid er kjent for sin svært begrensede sikkerhet [Palmer, 2004, side 381]. Men sett i forhold til hvilke data det her er snakk om, så er denne sikkerheten mye bedre enn åpen kommunikasjon. Om noen av enhetene skulle støtte nye og bedre krypteringsteknikker så er problemet ofte at det ikke er alle enhetene hjemme som støtter disse, og for å da gjøre det enkelt så velges ofte minste felles multiplum som er WEP.

En annen ting det ofte er smart å gjøre er å sjekke senderadiusen på det trådløse aksesspunktet ditt. I de aller fleste tilfellene leveres disse med maksimal senderstyrke fra produsenten og det vil i mange tilfeller si at det er mulig å fange opp signalene langt utenfor husets fire vegger. Derfor kan det være smart å sette ned senderstyrke slik at det bare er mulig å aksessere det trådløse nettverket fra det huset der den står. Dette vil med en gang gjøre det mye vanskeligere for en angriper, siden han da må være i umiddelbar nærhet av huset der aksesspunktet står.

[13] Alle trådløse aksesspunkter leveres med WEP støtte. Dette er den mest utbredte sikkerhetsmekanismen innen trådløs teknologi.



Figur 20: Denne figuren oppsummerer de sikkerhetstiltakene som kan iverksettes i et trådløst hjemmenett. Tesktboksen lister opp punktene som med enkle midler og ingen spesielle datakunnskaper kan gjøre det trådløse hjemmenettverket sikrere.

8.2 Trådløse bedriftsnett

Den gang trådløse nettverk ble tatt i bruk av bedrifter var det mest vanlig at det trådløse aksesspunktet ble satt på utsiden av bedriftens brannvegg. Brukerne måtte derfor tunnelere seg inn med en VPN klient (se kapittel 3.1 på side 16) eller en annen sikkerhetsløsning for å komme seg inn på bedriftens intranett etter at de koblet seg opp på det trådløse nettverket. På denne måten unngikk man de fleste sikkerhetshullene i det trådløse nettverket. I dag er det imidlertid kommet WLAN løsninger som regnes for fullgode sikkerhetsmessige. Disse går under betegnelsen WPA (Wi-Fi Protected Access) og finnes i to versjoner og er videreutviklinger og forbedringer av WEP. Når vi går litt mer detaljert til verks angående sikkerhet deler vi det inn i tre funksjoner:

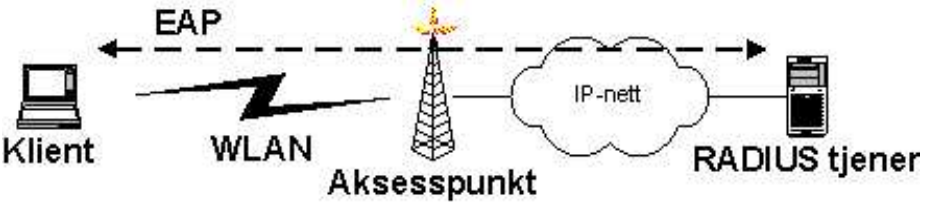
- Autentisering
- Konfidensialitet
- Integritet

WPA har to modi som opererer på litt forskjellige måter:

- Personmodus (personal mode)
- Bedriftsmodus (enterprise mode)

Ut fra navnene kan vi se at disse sikkerhetsløsningene også er tiltenkt det private markedet, men da i litt forenklede utgaver.

Autentisering består i at maskinene som kobler seg til det trådløse nettverket må bevise hvem de er og at de er autorisert til å få tilgang til nettverket. I personmodus foregår dette med en fast konfigurert nøkkel (PSK. Pre Shared Key) på 256-bits. I bedriftsmodus foregår denne autentiseringen på en litt annen måte. Der er det en sentral autentiseringstjener som tar seg av denne oppgaven. Denne tjeneren kan betjene et antall aksesspunkter. Selve autentiseringsprotokollen som går ende til ende mellom klientmaskinen og autentiseringstjeneren heter EAP (Extensible Authentication Protocol) og bæres over radiolinken av WLAN-protokollen og videre over IP-nettet av RADIUS-protokollen. (se figur)



Figur 21: EAP autentisering i WPA

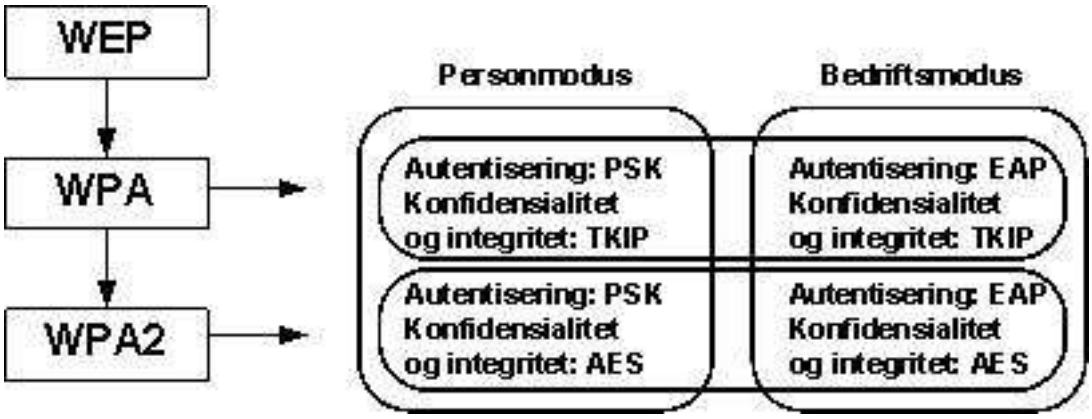
Dette fungerer på den måten at aksesspunktet i utgangspunktet kun åpner for EAP-protokollen inntil autentiseringen er fullført. Etterhvert som klienten blir autentisert får aksesspunktet beskjed om dette og åpner dermed for vanlig trafikk. EAP er en videreutvikling av CHAP (Challange Handshake Protocol) som er kjent fra oppringt PPP [14] forbindelser i telenettet. Fordelen med EAP er at denne spesifiserer en generell autentiseringsprotokoll som kan benytte ulike autentiseringsmetoder som for eksempel passord, digitale sertifikater eller SIM kort. Det er også kommet utvidede versjoner av EAP som for eksempel EAP-TTLS (EAP-Tunneled Transport Layer Security) som som setter opp en sikret kommunikasjon mellom klient og RADIUS-tjener slik at også denne trafikken ikke kan avlyttes.

Når så selve autentiseringen er unnagjort og det er tid for selve dataoverføringen, må en sikre denne kommunikasjonen for å opprettholde konfidensialitet (hemmelighold av data) og integritet (forsikre seg mot at data ikke er endret). Det blir til en viss grad også utført av WEP, men denne beskyttelsen er mangelfull. WEP baserer seg i utgangspunktet på en 40-bits nøkkel og anvender RC4-krypteringsalgoritmen [Stallings, 2003, side 192]. Integriteten sjekkes ved at sjekksummen overføres kryptert sammen med dataene. Det er dermed mulig å fange opp nøkkelen siden denne må sendes forholdsvis dårlig kryptert. Dette

[14] PPP = Punkt til punkt protokoll (Point to Point Protocol) som er ine internettprotokoll som benyttes for å koble sammen datamaskiner via serielle linjer, som for eksempel telefonlinjer.

gjør det da mulig å fange opp senere data og forandre disse. Problemet kommer av at det brukes statiske nøkler. WPA derimot, innfører en protokoll som heter TKIP (Temporal Key Integrity Protocol) som gjenbruker den grunnleggende krypteringsmekanismene til WEP. Men TKIP bruker derimot en 128-bits nøkkel som byttes jevnlig. Dette gjør det mye vanskeligere for en angriper å knekke nøkkelen og hvis det skjer, vil bare nøkkelen være gyldig en viss tid. WPA benytter seg også av en ny og forbedret integritetsbeskyttelse som kalles MIC (Message Integrity Check). WPA inneholder også såkalte mottiltak som følger opp MIC-feilene og overvåker at hyppigheten på disse ligger under et visst nivå.

Det skjer hele tiden nyutvikling når det gjelder trådløs sikkerhet, og den aller nyeste WEP2 benytter seg av AES-krypteringer (Advanced Encryption Standard) [Stallings, 2003, side 139] som er etterfølgeren til den mer kjente DES (Data Encryption Standard).



Figur 22: Forskjeller mellom WEP/WPA

9 Mobiltelefoner/PDA

Mobile enheter er kommet for fullt de siste årene. Dette innbefatter mobile enheter som for eksempel mobiltelefoner og andre håndholdte enheter samt vanlige datamaskiner med trådløse nettverkskort. Dette har gitt brukerene stor frihet når de skal kommunisere med venner, kjente eller i skole/jobbsammenheng. Med denne friheten kommer det også en risiko når det gjelder sikkerhet.

9.1 Mobilnettverket

Det nettverket som benyttes for å muliggjøre kommunikasjon mellom håndholdte enheter som benytter seg av for eksempel GSM,UMTS,GPRS og WAP består av en rekke enheter:

- Mobile station
  Dette er selve enheten som benyttes som kommunikasjonsmiddel. For eksempel en mobiltelefon.
- International Mobile Subscriber Identity (IMSI)
  En unik identifikator som er tilegnet brukeren+mobilenheten. Denne består av en land-kode (MCC), en mobil nettverkskode (MNC) og en mobil stasjon identifikasjonsnummer (MSIN)
- Subscriber Identity module
  Et smartkort som som plugges inn i den mobile enheten. Dette inneholder en hemmelig nøkkel som brukes til autentisering. Denne nøkkelen beskyttes med et PIN nummer som må tastes inn for å aktiveres.
- Electronic serial number
  Dette er en 32-bits identifikator som tilegnes mobilstasjonen av produsenten
- Cell tower
  Dette er selve mobilantennene som står rundt omkring i landskapet
- Base Transciever station
  Dette er enheten som står for selve kommunikasjonen mellom mobilantennen og den mobile enheten.
- Base station controller eller base statsjon
  Disse kontrollerer et antall mobilantenner. De kontrollerer også overgangen mellom de forskjellige antennene nå mobilenheten beveger seg og formidler dette videre til mobil svitsjesenteret.

- Mobile switching center (MSC)
  Denne mottar kommunikasjonen fra basestasjonene og ruter disse videre til de basestasjonene kommunikasjonen er tiltenkt eventuelt til fasttelefonlinjer. De utfører også andre oppgaver slik som registrering, autentisering, roaming og ruting for mobile enheter. For å få til dette har svitsjesenterene følgende komponenter:

  - Home location register (HLR)
    Denne holder orden på abonnentens informasjon om hvor den befinner seg i mobilnettverket. HLR holder orden på all nødvendig informasjon som trengs for å initiere, terminere eller motta et anrop
  - Visitor location register (VLR)
    Denne holder orden på et subset av det som befinner seg i HLR, men er tilknyttet et bestemt geografisk område.
  - Autentication center (AuC)
    Denne bruker en beskyttet database for å autentisere og validere all trafikk som foregår på mobilnettet. Autentiseringen skjer med nøkkelen som ligger lagret på mobilenhetens SIM kort.
  - Equipment identity register (EIR)
    Dette er en database som inneholder en liste over alle registrerte mobilenheter på nettverket basert på IMEI. [15] Hvis en mobilenhet blir stjålet kan dette legges inn i EIR og denne enheten blir da forhindret fra å benyttes.

[15] IMEI =International Mobile Equipment Identity. Dette er en unik identifikator fr hver mobile enhet

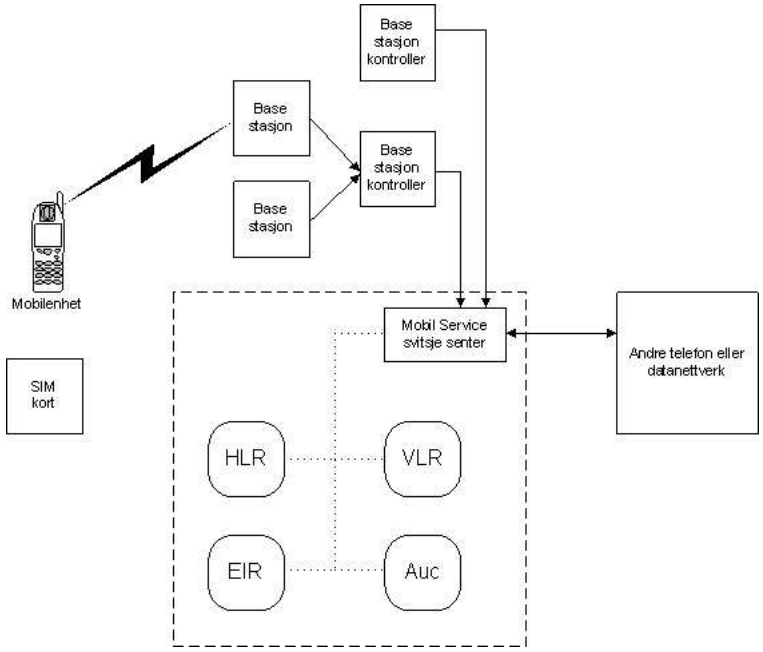


Figure 23: Hoveddelene i et mobilnettverk

9.2 Sikkerhet i GSM nettet

De sikkerhetmekanismene som finnes i GSM nettet er:

- PIN kode.
  Dette autentiserer SIM kortet som benyttes i kommunikasjonen og har følgende egenskaper:
  - Den er lagret i SIM-kortet. Autensisering skjer derfor lokalt og ikke over nettverk
  - Må oppgis når enheten aktiveres.
  - Hvis feil kode tastes inn 3 ganger må PUK koden oppgis. (Personal Unblocking Key)
  - Hvis PUK koden tastes inn feil 10 ganger, låses SIM kortet og det må skaffes et nytt kort fra operatør.

- Bruker autentisering.
  GSM nettverket autentiserer identiteten til moblienheten ved å bruke en utfordring/svarmekanisme (challange/respons). Et 128-bits tilfeldig tall (RAND) sendes til mobilenheten og denne lager så et 32-bits svar på denne utfordringen (SRES). Når så GSM nettverket får dette svaret tilbake

fra mobilenheten sjekkes så dette svaret, og dersom det er gyldig, er mobileneheten autentisert og får iverksette kommunikasjon.

- Kryptering av kommunikasjon.
  SIM kortet inneholder en krypteringsalgoritme som brukes til å produsere en 64-bits krypteringsnøkkel. Nøkkelen genereres ved å bruke det 128-bits tilfeldige tallet som ble sendt til mobilenheten fra GSM nettverket. GSM genererer også denne 64-bits nøkkelen ved å bruke den samme krypteringsalgoritmen og tilfeldige tall. Dermed har både mobilenhet og GSM nettverk en felles krypteringsnøkkel som brukes til å kryptere kommunikasjonen. Denne krypteringsnøkkelen blir forandret med jevne mellomrom og som oftest ved hvert oppkall.

9.3 Sikkerhetsproblemer i GSM nettet

Hvor sikkert GSM nettet er, er det nesten ingen som vet. Det vi derimot vet er at krypterings og autentiseringsalgoritmene som benyttes i GSM er unndratt offentligheten. Det vil si at vi ikke vet med sikkerhet hvor sterke disse er, og dermed hvor sikker trafikken er. Det er kommet frem på flere hold at algoritmene som brukes er for svake, og selv om det aldri er offentliggjort, er det kommet sterke antydninger om at krypteringsalgoritmen er knekket ved hjelp av såkalt "reverse engineering". Det vil si at man tar utganspunkt i den krypterte meldingen og kommer frem til den krypteringsalgoritmen som er benyttet for å kryptere den. Dette vil da føre til at det er mulig å dekryptere alle andre GSM signaler som går via radiobølger

10 Blåtann

Blåtann er en peer-to-peer protokoll med kort rekkevidde som brukes for å koble mobiltelefoner, bærbare datamaskiner, håndholdte databaskiner (PDA), digitalkamera, skrivere osv. sammen, eller til et nettverk. Blåtann har følgende egenskaper:

- Benytter FHSS i overføringen. [Cole and Conley, 2005, side 395]
  FHSS står for "Frequency Hopping Spread Spectrum" (Frekvenshopping i bredt spekter). Dette betyr at bæresignalet til blåtann hopper 1600 ganger i sekundet mellom 79 forskjellige frekvenser som ligger mellom 2.4 GHz og 2.5 GHz.
- Overføringshastigheten ligger normalt på rundt 1 Mbps.
- Sigalstyrken som normalt benyttes ligger rund 1 milliwatt. (mobiltelefoner kan sende på så mye som 3 watt)
- Maksimal overføringsavstand ligger i praksis på rundt 9 meter.
- Overføringsavstanden kan økes til rundt 100 meter desom man øker senderstyrken til maksimum.
- Antall enheter hver basestasjon kan håndtere er 8.

Fordi FHSS benyttes kan flere enheter benyttes på samme nettverk uten å forstyrre hverandre. Blåtann enheter opererer ved at de sette opp et personlig netverk (PAN - Personal Area Network) som kalles et piconett (pico betyr "lite" i denne sammenhengen). Dette nettverket fungerer slik:

- Som et *ad hoc* nettverk.
  Dette betyr at nettverket settes opp på sparket og planlegges ikke av en nettverksadministrator.
- Alle blåtann-enhetene på nettverket er klient enheter (peer)
- Forskjellige piconett har forskjellige frekvenshopping-sekvenser for å forhindre forstyrrelser.
- Alle enhetene innen et piconett er synkronisert i forhold til frekvenshoppingen.
- En blåtann-enhet opererer som sjef og de andre enhetene som slaver. Sjefen er basestasjonen.
- Maksimalt 7 aktive slaver kan operere samtidig på et piconett. Hver av disse har en 3-bits adresse.

- Opp til 256 inaktive slaver (parkerte) som er synkronisert til frekvenshoppingen kan være på piconettet. Disse kan aktiveres raskt, siden de allerede er synkronisert.

10.1 Sikkerhet i blåtann

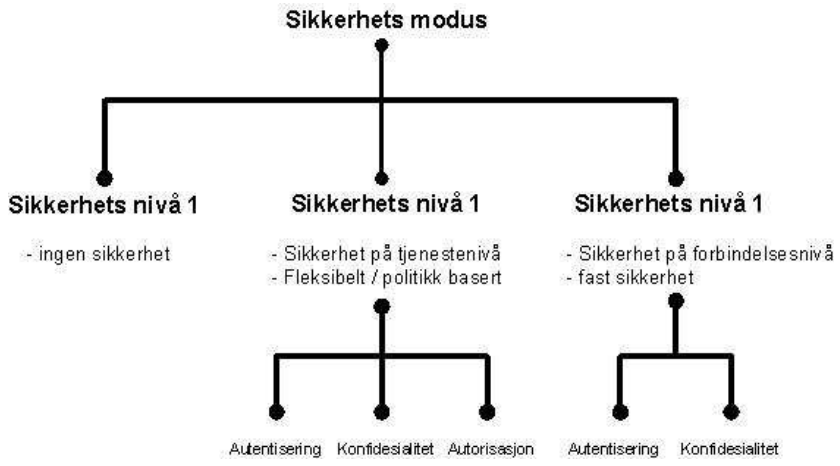
Blåtann ha i følge spesifikasjonene tre forskjellige operasjonsmodus nå det gjelder sikkerhet. Hver enhet kan bare operere i en modus på et gitt tidspunkt. De tre nivåene er:

- Sikkerhetsnivå 1 - Usikker modus
- Sikkerhetsnivå 2 - Sikkerheten må benyttes på tjenestenivå
- Sikkerhetsnivå 3 - Sikkerheten må benyttes på forbindelsesnivå

På sikkerhetsnivå 1 vil ikke enheten benytte seg av noen form for sikkerhet, hverken kryptering eller autentisering. Dette er potensielt farlig, og bør kun benyttes på enheter som ikke inneholder viktig informasjon.

På sikkerhetsnivå 2 opererer applikasjoner med forskjellige krav til autentisering, konfidensialitet og autorisasjon. Applikasjoner som tilbyr brukerene graderte tjenester bør kjøre på sikkerhetsnivå 2.

På sikkerhetsnivå 3 blir blåtann-enheten påtvunget sikkerhet på samme nivå for alle applikasjoner når oppkoblingen skjer.



Figur 24: Sikkerhetsnivåer i blåtann

For mer informasjon om sikkerheten i blåtann henvises det til [Karygiannis and Owens, 2002, Kapittel 4]

10.2 Enkle kjøreregler for å øke sikkerheten med blåtann

- Slå av blåtann når du ikke bruker den.
  En av de største feilene folk gjør er å lå blåtann-enheter være aktive hele tiden. Dette utgjør en stor sikkerhetrisiko og er i tilegg helt unødvendig.

- Bruk blåtann i "usynlig" modus (hidden).
  Når du bruker blåtann-enheten så sørg for å ha den i skjult (hidden) stilling og ikke i oppdagbar (discoverable). I skjult modus er det mye vanskeligere for andre blåtann-enheter å oppdage din enheten. Dette forhindrer deg ikke å bruke enheten og du kan fortsatt koble sammen forskjellige blåtann-enheter ved å "parre" (pair) disse.

- Vær forsiktig når du bruker blåtann.
  vær oppmerksom på omgivelsene når du bruker blåtann. Dette gjelder spesielt når du "parrer" enheter, eller har de i oppdagbar modus. Det er mye større sjanse for å bli utsatt for et angrep på et folksomt sted, som for eksempel på en kafe, enn det er hjemme.

- Se over sikkerhetsinstillingene på blåtann-enheten.
  De fleste blåtann-enheter kommer ferdigkonfigurert fra fabrikken. Problemet med dette er at innstillingene er altfor åpne og usikre. Se derfor over innstillingene på enheten din og benytt deg av de sikkerhetsmekanismene enheten tilbyr.

Del II

Praktisk anvendelse av sikkerhetsmekanismer

11 IP Security

(Se kapittel 3.3 på side 24)

11.1 Når skal du bruke IPSec

IPsec brukes når en vil kryptere all forbindelse mellom to eller flere maskiner og er en av de beste metodene for å sikre kommunikasjon over ut usikret nettverk. IPSec sikrer TCP/IP kommunikasjon ved å sikre IP på nettverks-laget (eller lag 3) i TCP/IP-stakken som er selve hjertet til IP.

11.2 Typiske eksempler på bruk av IPsec

IPSec gir muligheten til å sikre kommunikasjon over LAN,WAN og internett. Eksempler på anvendelsesområder for IPSec kan være:

- Sikre trafikk til/fra avdelingskontor og hovedkontor.
  En bedrift kan sette opp et virtuelt privat nettverk over internett for å koble sammen avdelingskontoret og hovedkontoret. Før måtte man leie dedikerte linjer for å sette opp slik kommunikasjon, men dette lar seg nå gjøre ved å ta i bruk IPSec.

- Sikker adkomst via internett
  En sluttbruker kan ved hjelp av IPSec få adgang til bedriftens nettverk ved å ta i bruk IPSec. Dette ligger ofte innebygd i såkalte VPN-løsninger (se kapittel 3.1 på side 16), og i bunnen av disse er det IPSec som står for sikkerheten.

- Åpne muligheter for extranet-/intranetdeling mellom bedrifter som jobber i partnerskap.
  IPsec kan brukes for å ha delte netverksressurser mellom partnere som samtidig sikrer konfidensialitet og autentisering.

- Øke sikkerheten ved økonomiske transaksjoner på internett.
  Selv om enkelte nettverktøy og nett-applikasjoner har innebygde sikkerhetsmekanismer vil det å ta i bruk IPSec i tillegg øke sikkerheten.

11.3 IPSec terminologi

IPsec term	Akro-nym	Definisjon
Security association	SA	En unik forbindelse mellom to noder i et nettverk. Forbindelsen blir definert av tre ting: en sikkerhetsprotokoll, et sikkerhetsparameter indeks og en IP destinasjon
Security association database	SADB	Database som inneholder alle aktive sikkerhetsassiasjoner
Security parameter index	SPI	Indeksverdien for en sikkerhetsassosiasjon. En SPI er en 32-bits verdi som skiller mellom SA'er som har samme IP destinasjon og sikkerhetsprotokoll
Security policy database	SPD	Database som bestemmer om utgående eller inngående pakker har det spesifiserte nivået sikkerhet.
Key exchange		Prosessen som genererer nøkler for asymetriske krypteringsalgoritmer. De to hovedmetodene er RSA og Diffie-Hellmann.
Diffie-Hellmann protokoll	DH	En nøkkelutvekslingsprotokoll som inbefatter nøkkelgenerering og nøkkelautentisering. Ofte kalt **autentisert nøkkelutveksling.**
RSA protokoll	RSA	En nøkkelutvekslingsprotokoll som involverer nøkkelgenerering og distribusjon
Internet Security Association and Key Management Protocol	ISAKMP	Det felles rammeverket for etablering av formatet på SA attributter, og for forhandling, modifisering og sletting av SA'er.
Authentication header	AH	Et "tilleggshode" på IP-pakken som gir autentisering og integritet , men ikke konfidensialitet, til IP datagram
Encapsulating security payload	ESP	Et "tilleggshode" på IP-pakken som gir integritet og konfidensialitet til IP-datagrammet
Internet Key Exchange	IKE	IKE automatiserer tilgangen til nøkkelmateriale som brukes til autentisering i IPSec.

Table 6: IPSec terminologi

11.4 Oppsett av IPSec på Windows XP

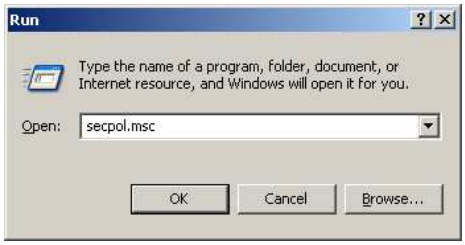
Dette eksemplet viser hvordan en setter opp IPSec mellom to maskiner på Windows XP.

Forutsetninger:

- All trafikk mellom de to maskinene skal krypteres med IPSec
- All trafikk foregår med IPv4
- Begge maskinene står på samme lokalnettverk (LAN)
- Maskinene heter :
  - helldesk med IP-adresse 129.159.112.77
  - lab-laptop med IP-adresse 129.159.112.31
- Brukeren er logget inn som administrator
- Her brukes Engelsk versjon av Windows XP. Ordlyden på Norsk versjon vil variere litt, men funksjonene og plasseringene er de samme.

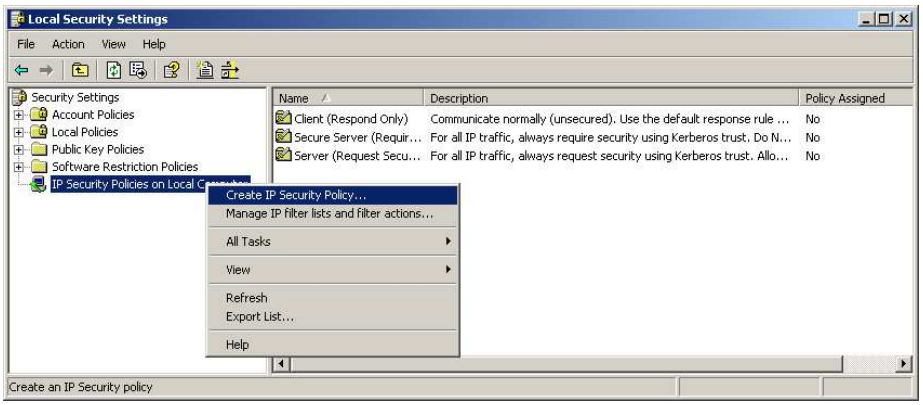
11.4.1 Oppsett med eksempler

Kjør: *START –> Run* og skriv inn "secpol.msc"



Figur 25: Oppstart av secpol.msc

Da kommer en til den lokale IPSec konfigurasjonen i Windows XP. Der høyreklikker man på: *"IP Security Policies on Local Computer* " og velger "*Create IP Security Policy*":

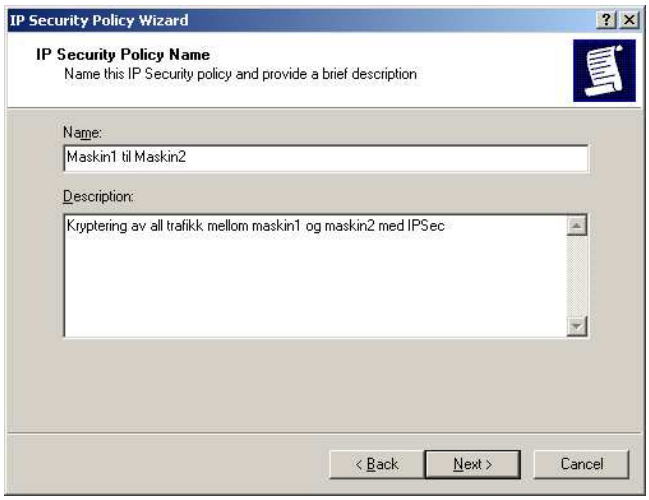


Figur 26: Windows lokale IPSec konfigurasjon

Vi kommer da inn i en hjelper som vil lede oss fremover. Velg deretter "*Next*":

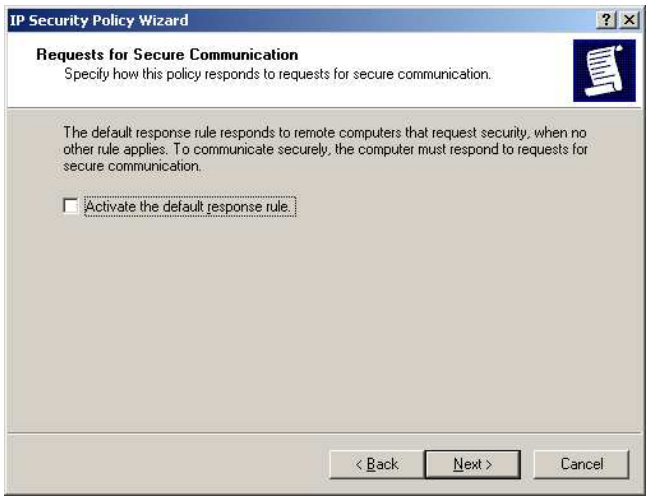
Figur 27: IPSec hjelper. Steg 1

I det neste skjermbildet setter man inn et navn på den nye policy'en og en liten forklaring på hva den gjør. En maskin kan ha mange forskjellige policy'er, men navnene på disse må være unike. Etter at man har satt inn et navn og en liten forklaring, trykk "*next*":



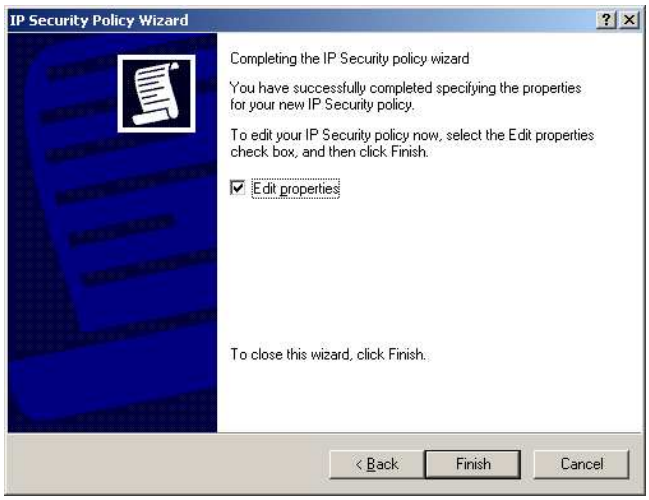
Figur 28: IPSec hjelper. Steg 2

I det neste skjermbildet tar man vekk krysset i "*Activate the default respons rule*" fordi vi skal lage vår egen policy her og ikke bruke de forhåndsdefinerte. Klikk deretter "*next*":



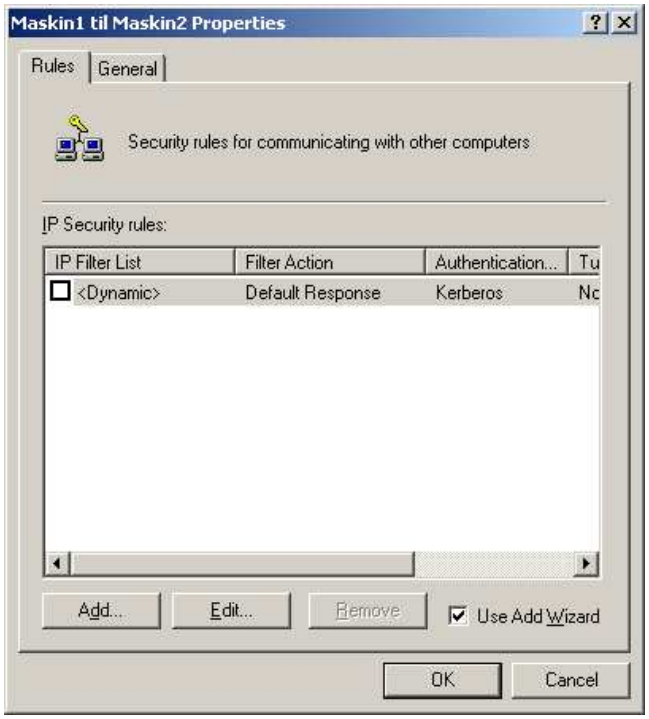
Figur 29: IPSec hjelper. Steg 3

I det neste skjermbildet lar man krysset i "*Edit* properties" være og velger "*next*":



Figur 30: IPSec hjelper. Steg 4

Da kommer man til egenskapene til den nye policy'en vi har laget. Som vi ser så er ingen regler forbundet med den nye policy'en ennå. Derfor velger vi "*Add*" på den nest nederste linjen:



Figur 31: IPSec hjelper. Steg 5

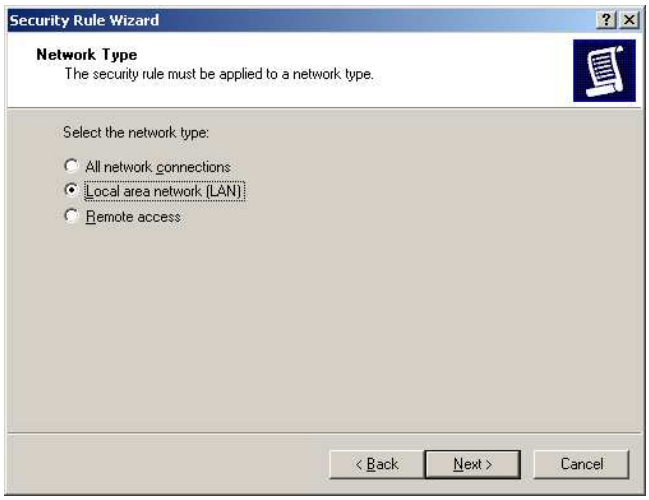
Da kommer man inn i en ny hjelper som leder oss gjennom opprettelsen av en ny regel. Velg deretter "*next*":

Figur 32: IPSec hjelper. Steg 6

Vi skal i dette tilfellet ikke kjøre IPSec i tunnelmodus, men i transportmodus. Pass derfor på at "*This rule does not specify a tunnel*" er valgt. Klikk deretter "*next*":

Figur 33: IPSec hjelper. Steg 7

I det neste vinduet spesifiseres det hvordan maskinene står i forhold til hverandre. I dette eksemplet velger vi "*Local area network (LAN)*". Klikk deretter "*next*":

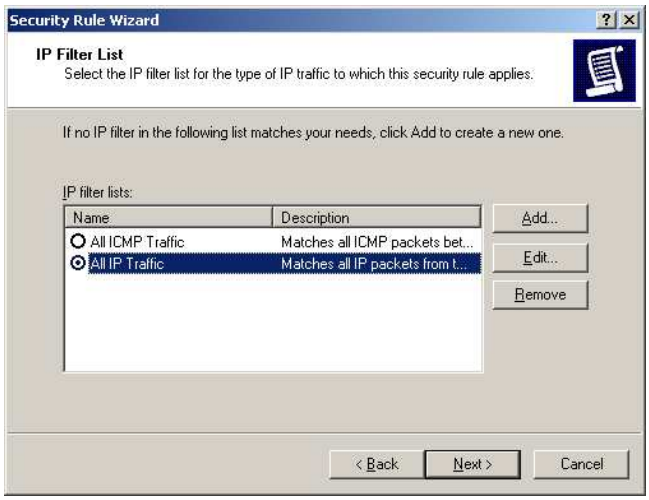


Figur 34: IPSec hjelper. Steg 8

I det neste skjermbildet velger vi hvilken autentiseringsmetode vi ønsker. Her kan vi velge "*Active Directory*", "*seritfikatsbasert*" eller *"Preshared nøkler*". I dette eksemplet bruker vi forhåndsdelte nøkler og skriver inn nøkkelen i feltet under. Denne nøkkelen bør være en randomisert streng, men det er også mulig å bruke en vanlig setning. Dette gjør det mye enklere å dele nøkkelen. Man kan for eksempel gi den via telefon eller lignende, noe som er vanskelig med randomiserte strenger. Her i eksemplet har vi brukt: "***Dette er den forhåndsdelte nøkkelen***" som nøkkel. Velge deretter "*next*"*:*

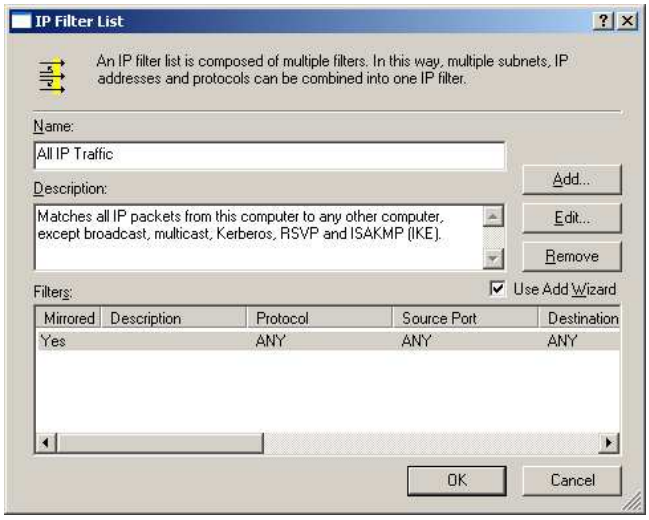
Figur 35: IPSec hjelper. Steg 9

Kryss av for "*all IP Traffic*" i det neste skjermbildet og velg "*Edit*" fra knappene til høyre.



Figur 36: IPSec hjelper. Steg 10

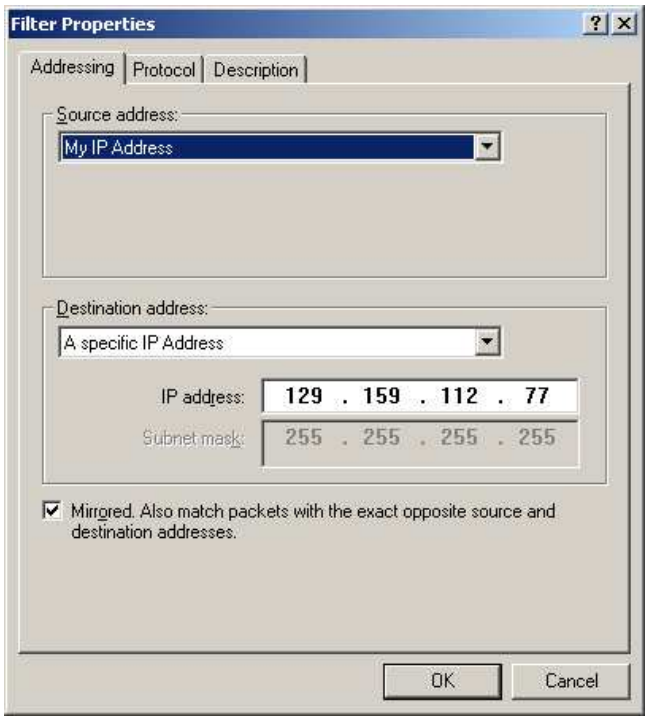
I dette skjermbildet vises filterlisten. Velg deretter "*Edit*" fra knappene til høyre.



Figur 37: IPSec hjelper. Steg 11

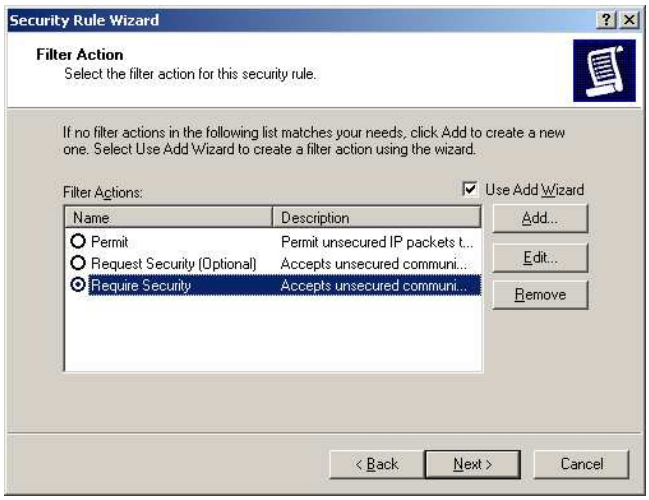
I dette skjermbildet velger vi hvilke adresser dette filteret gjelder for. Her lar vi "*Source address:*" øverst være på *"My IP Address"* og velger *"Destination address"* til *"A specified IP Address"* og fyller inn IP-adressen på den andre

maskinen vi skal kommunisere med i *"IP address"*-feltet. I dette tilfellet er IP-adressen: 129.159.112.77. På den andre maskinen, 129.159.112.31, vil dette selvsagt være omvendt. Pass også på at sjekkboksen til *"mirrored"* er avkrysset. Dette betyr at filteret gjelder begge veier. Velge deretter *"OK"* for å komme tilbake til det forrige skjermbildet, og deretter *"OK"* en gang til for å komme tilbake til *"IP filter list"*-menyen. Velg deretter "*Next*"*:*



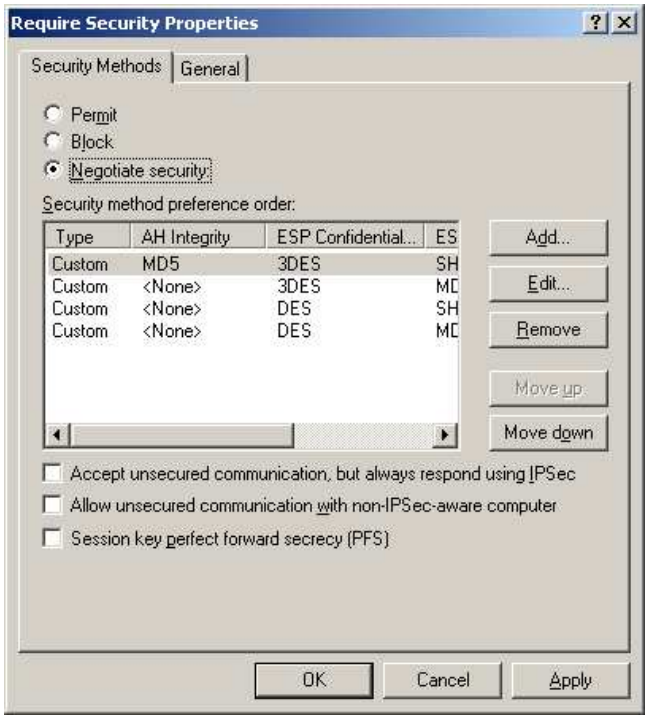
Figur 38: IPSec hjelper. Steg 12

Den neste menyen bestemmer hva filteret skal gjøre. Kryss av for "*Require Security*" og velg "*Edit*" fra knappene til høyre.



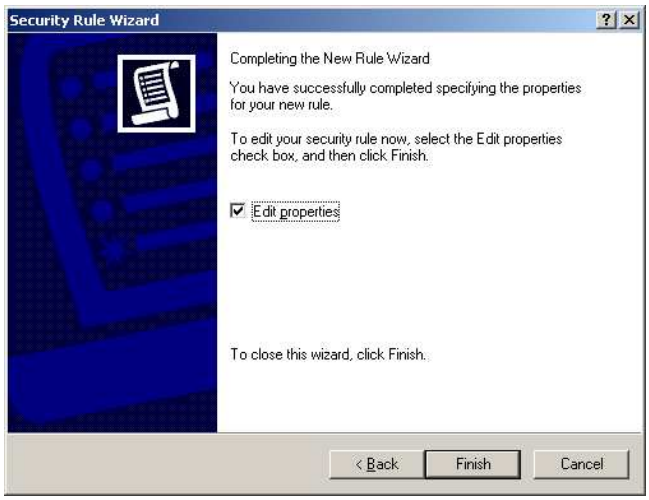
Figur 39: IPSec hjelper. Steg 13

I denne menyen velger vi "*Negotiate security*" og tar vekk eventuelle kryss på opsjonene nederst. Velg deretter "*OK*".



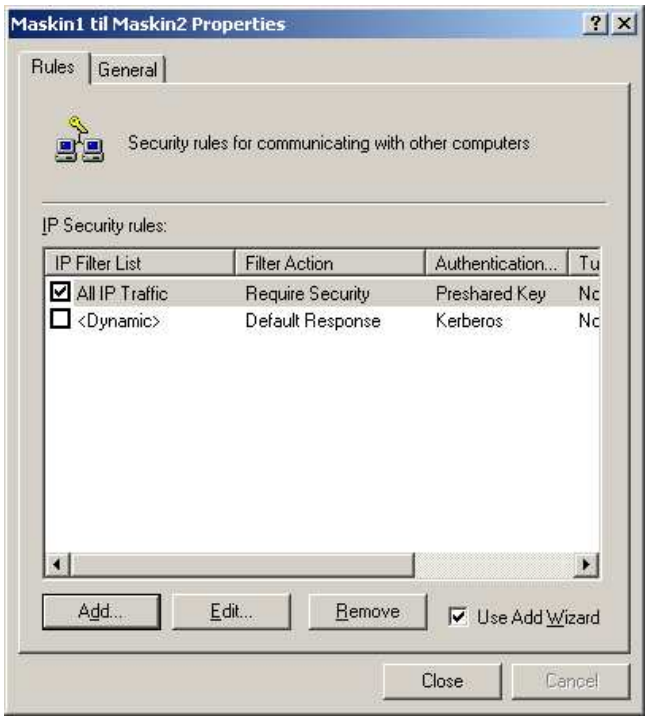
Figur 40: IPSec hjelper. Steg 14

I det siste bildet i hjelperen velger vi "*finish*":



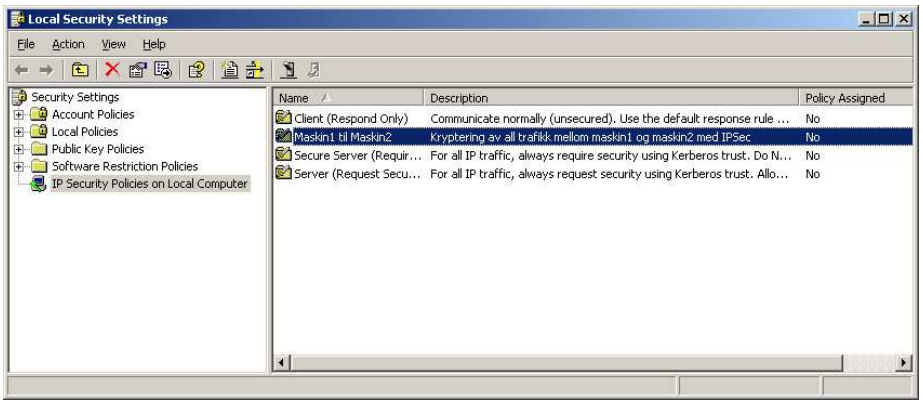
Figur 41: IPSec hjelper. Steg 15

Deretter får vi et oversiktsbilde over det vi har laget. Her er filteret vi nettopp har laget krysset av og reglene listet opp. Velg deretter "*close*"*:*



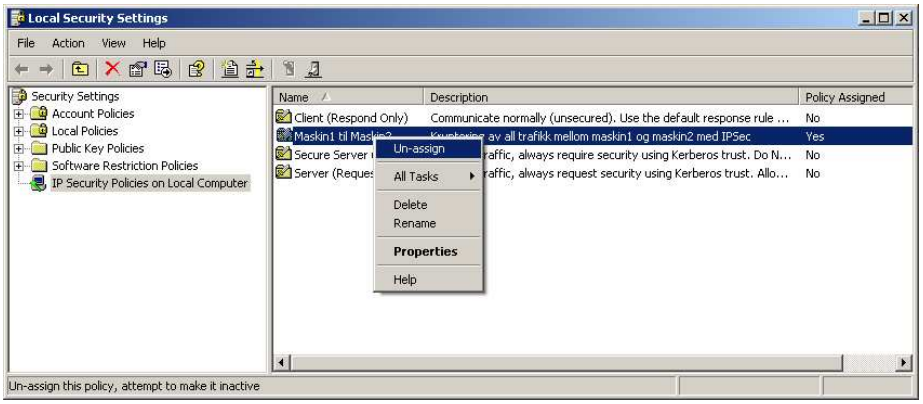
Figur 42: IPSec hjelper. Steg 16

Tilbake til sikkerhets-senteret til Windows XP ser vi nå at vi har fått ett nytt innslag under "*IP Security Policies on Local Computer*", "*maskin1 til maskin2*".



Figur 43: Windows sikkerhetssenter II

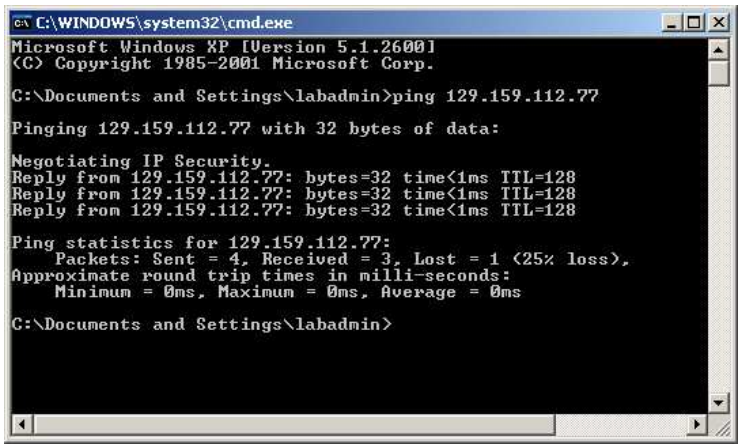
For å aktivisere dette nye filteret, høyreklikker vi på navnet, "*Maskin1 til Maskin2*", og velger "*Assign*". Dette vil vises visuelt ved at ikonet foran policy'en blir merket grønt.



Figur 44: Windows sikkerhetssenter III

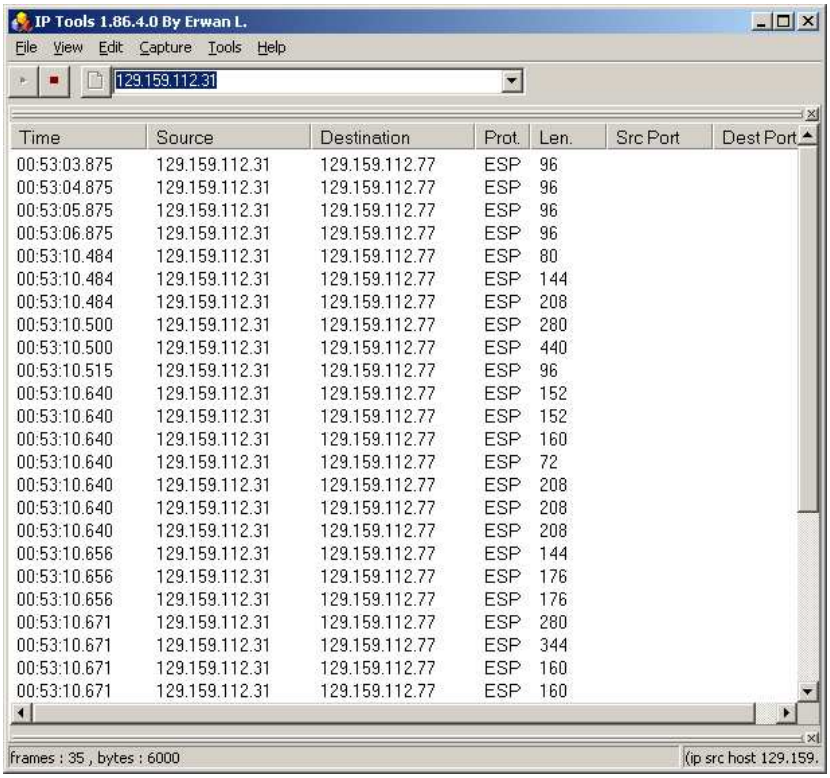
Når så disse punktene er utført på begge maskinene, kan vi så sjekke at kommunikasjonen blir kryptert med IPSec mellom disse to maskinene. Dessverre har ikke Windows XP innebygd noen monitor for dette, men vi kan se det ved å kjøre en standard ping [16] fra den ene maskinen til den andre:

[16]Ping er et verktøy som brukes for å fastslå om en IP-adresse er tilgjengelig. Programmet sender en spesiell pakke til en spesifisert IP-adresse og venter deretter på svar.



Figur 45: Ping test av IPSec

For virkelig å se at trafikken mellom disse maskinene blir kryptert med IPSec, kan man laste ned et tredjeparts program som for eksempel *IP Tools* og fange trafikken mellom de to maskinene slik:



Figur 46: IPtools verifikasjon av IPSec

Her ser vi trafikken fra 129.159.112.31 til 129.159.112.77 oppdelt etter tidspunkt, protokoll som benyttes og lengden på IP-pakken. Legg merke til at vi kun ser trafikken den ene veien, fra 129.159.112.31 til 129.159.112.77 og ikke motsatt vei.

12 SSL

(Se kapittel 3.2 på side 20)

12.1 Hvordan brukes SSL

For å sende og motta informasjon via en nettside benyttes som regel SSL (Secure Socket Layer). Dette er en protokoll som ligger mellom protokollen til applikasjonen og overføringsprotokollen. Etter at kontakt er opprettet mellom tjener og nettleser via en "handshake" prosess vil overføringen deretter bli kryptert. I "handshake" prosessen sender først tjeneren sitt sertifikat til nettleseren og dersom nettleseren også har sertifikat så sendes dette til tjeneren. Deretter lager nettleseren et sett med nøkler som krypteres med tjenerens offentlige nøkkel og sendes til tjeneren. Disse nye nøklene benyttes så til å kryptere all videre trafikk med en symmetrisk algoritme. En kryptert overføring krever ikke at nettleseren har sertifikat, men tjeneren må ha det.

12.2 Typiske eksempler på bruk av SSL

SSL benyttes ofte når web-tjeneren tilbyr en av følgende tjenester:

1. Det benyttes innloggingsmuligheter hvor du vil forvrenge brukernavn og passord og vil sikre deg mot at angripere får tak i innloggingsinformasjonen
2. Det benyttes skrivefelter (forms) på nettsiden der brukerne oppgir personalia, for eksempel i forbindelse med bestillinger eller innlevering av konfidensielt materiale, og du vil sikre deg at denne informasjonen ikke kommer på avveie under innsending.
3. Det er mulig å sende inn e-post via nettsiden og du vil sikre deg mot at denne kommer på avveie og bli lest av uvedkommende.
4. Informasjonen brukerne av nettsiden utveksler ikke skal være mulig å lese for andre.
5. At websiden skal tas som seriøs (spesielt hvis det er en bedrift) og ta brukerne på alvor

12.3 Oppsett av SSL på en Webtjener på Windows XP

I dette eksemplet har jeg tatt følgende utgangspunkt:

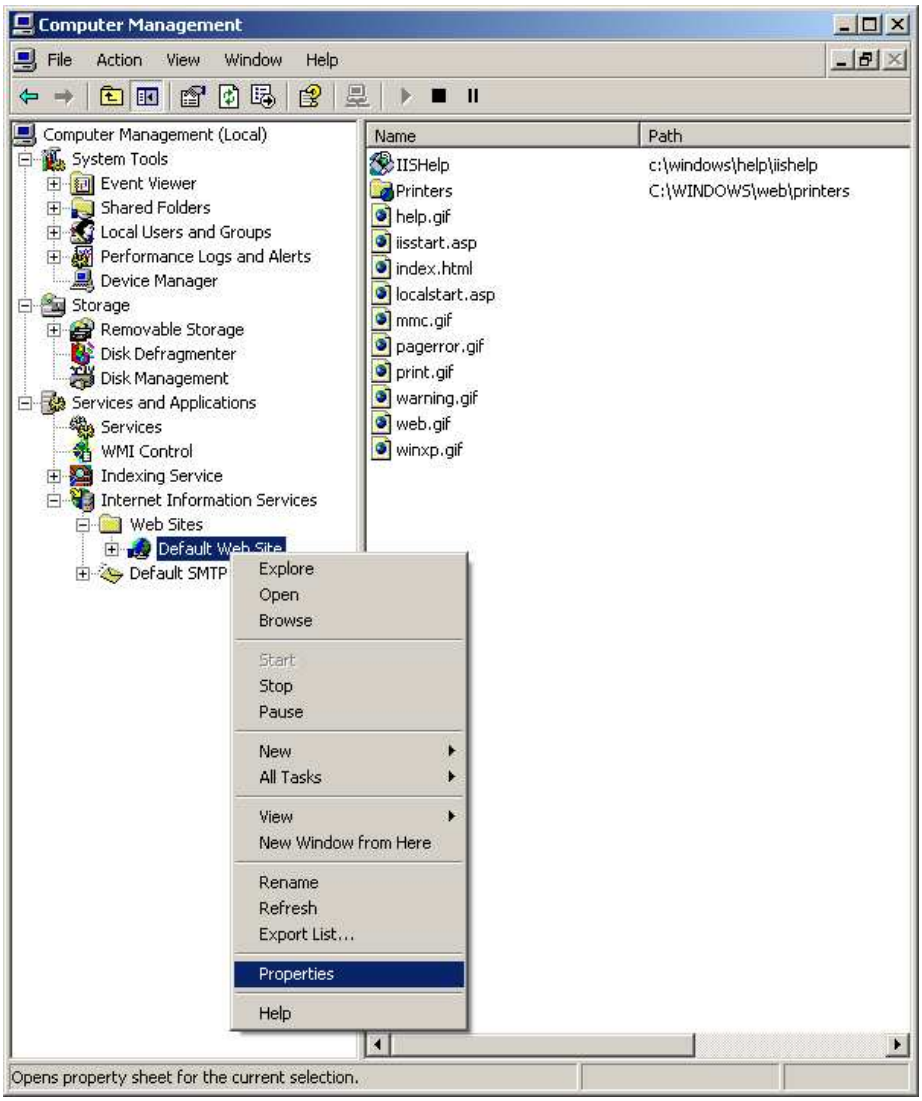
- Maskinen er en Windows XP Professional og har IIS (Internet Information Server) innstallert.

- Vi innstallerer her på en maskin som heter lab-laptop som har IP-adresse: 129.159.112.31
- Vi sender ikke inn sertifikatsforespørselen til en ekte CA, men signerer det selv.
- Windows versjonen i eksempelet er engelskspråklig, men på norsk versjon er knappene og dialogene like selv om ordlyden er litt forskjellig.

Start med å høyreklikke på *"My Computer"* og velg *"Manage"*:

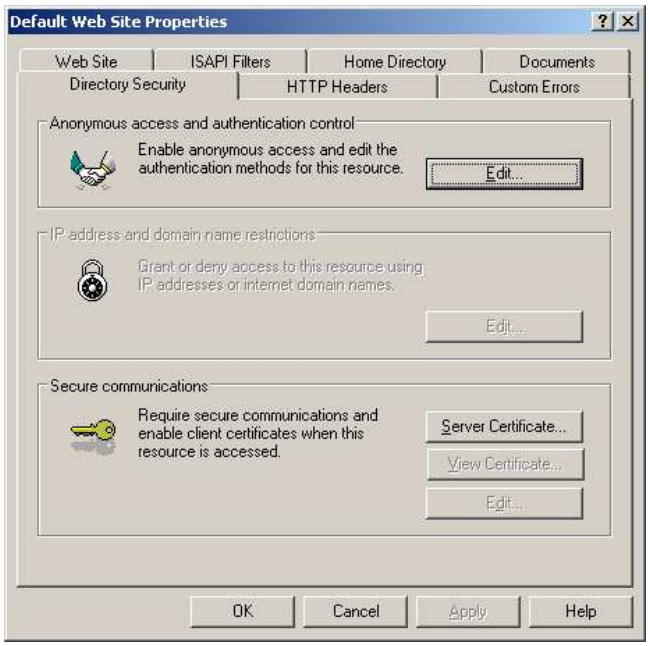
Figur 47: Starte administrering av maskinen

I den dialogen som da kommer opp, klikker vi oss ned til "*Computer Management*" –> "*Services and Applications*" –> "*Internet Information Services* –> "*Web Sites*" –> "*Default Web Site*" og høyreklikker på denne. (Se figur):



Figur 48: Egenskaper til Webtjeneren

I den dialogen som da kommer opp, velger vi fanen som er merket "*Directory Security*". Nederst på denne dialogen ligger innstillingene for sikker kommunikasjon til tjeneren. Klikk så på "*Server Certificate*":

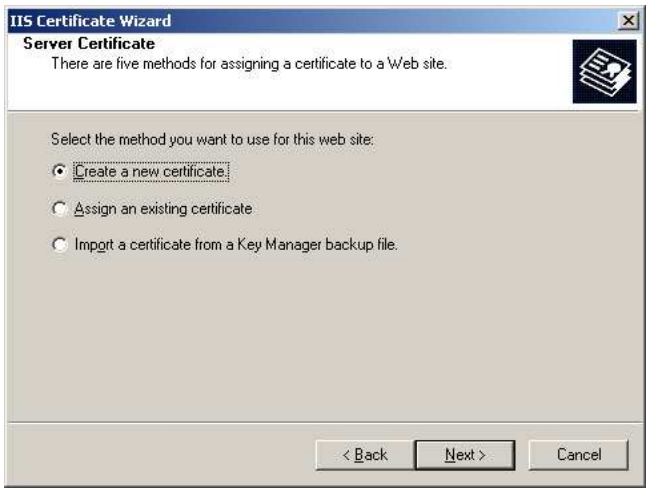


Figur 49: Webtjenerens sikkerhetssenter

Da kommer vi inn i en hjelper som vil guide oss gjennom opprettelsen og installasjonen av SSL sertifikat. Klikk så "*Next*":

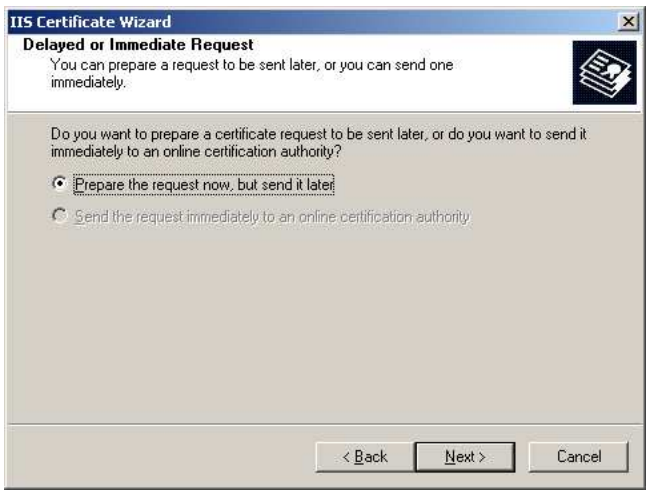
Figur 50: SSL hjelper. Steg 1

I den neste dialogen velger vi første gang "*Create a new certificate*". Klikk deretter "*Next*":



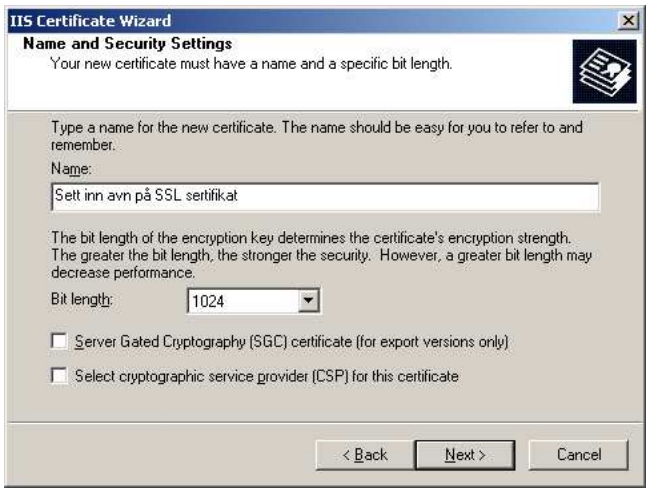
Figur 51: SSL hjelper. Steg 2

I den neste dialogen vil alltid opsjonen: "*Prepare the request now, but send it later*" være tiljengelig så vi lar den stå på denne. Den andre opsjonen: *"Send the request immidiately to an online certification authority"* vil kun være tilgejngelig dersom Web-serveren kan aksessere en Microsoft sertifikatstjener i sitt eget Windows-domene. Klikk deretter "*Next*"*:*



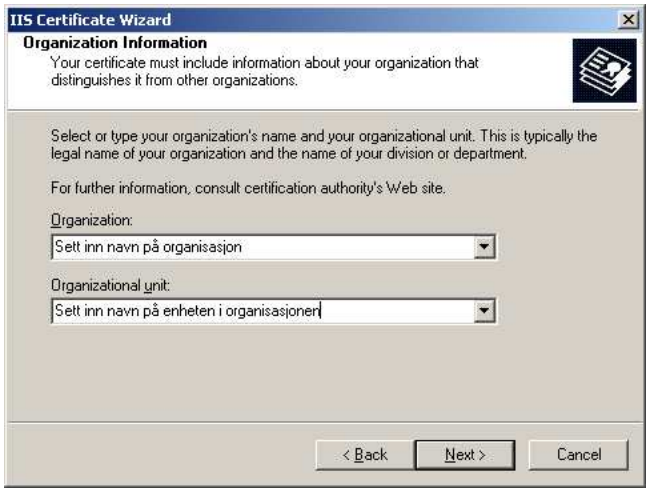
Figur 52: SSL hjelper. Steg 3

I denne dialogen setter vi inn et unikt navn for dette sertifikatet og bestemmer nøkkellengden. La nøkkellengden stå på "1024" og pass på at de to nederste alternativene ikke er krysset av (se figur):



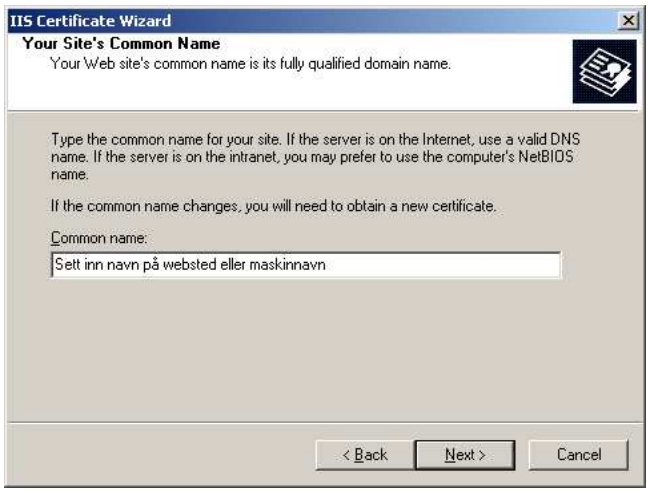
Figur 53: SSL hjelper. Steg 4

I den neste dialogen setter vi inn navn og enhet på organisasjonen som skal ha sertifikatet. Klikk deretter "*Next*":



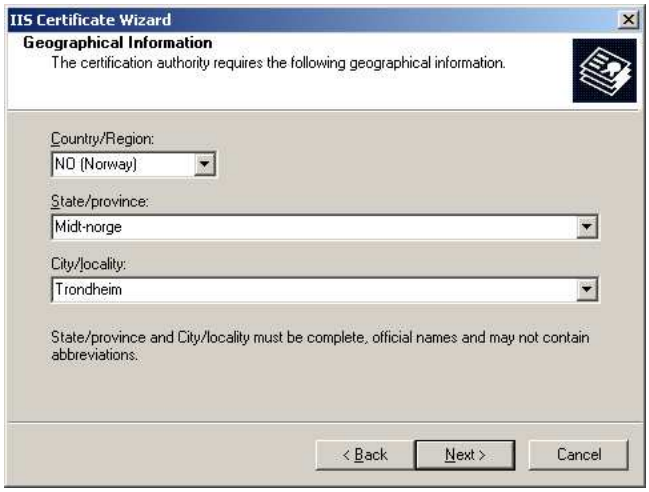
Figur 54: SSL hjelper. Steg 5

Her setter vi inn enten det offisielle navnet på webstedet eller maskinens navn. Klikk deretter "*Next*":



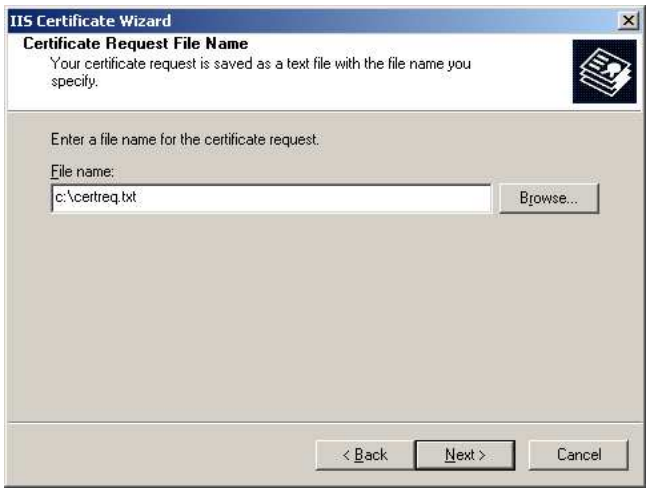
Figur 55: SSL hjelper. Steg 6

Fyll så inn den geografiske informasjonen for denne webtjeneren. Klikk deretter "*Next*":



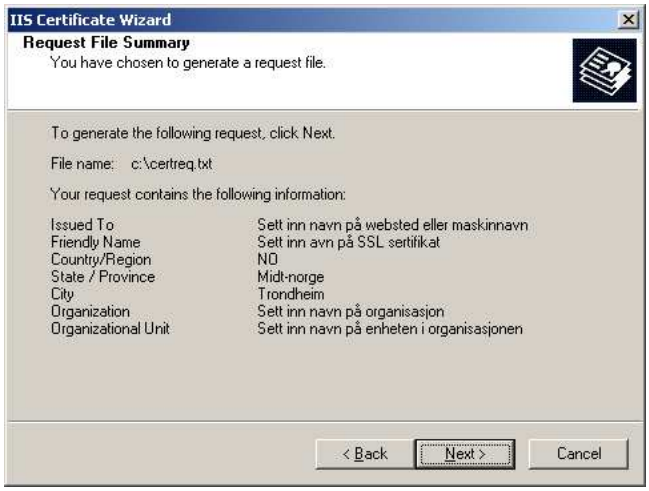
Figur 56: SSL hjelper. Steg 7

Deretter vil hjelperen spørre om hvor den skal lagre denne sertifikatsforespørselen. Her har vi valgt å lagre den som *C:\certreq.txt* Klikk deretter "*Next*":



Figur 57: SSL hjelper. Steg 8

I denne dialogen får du opp en oppsummering over forespørselen. Se over denne og forsikre at alt er riktig og trykk deretter "Next"*:*

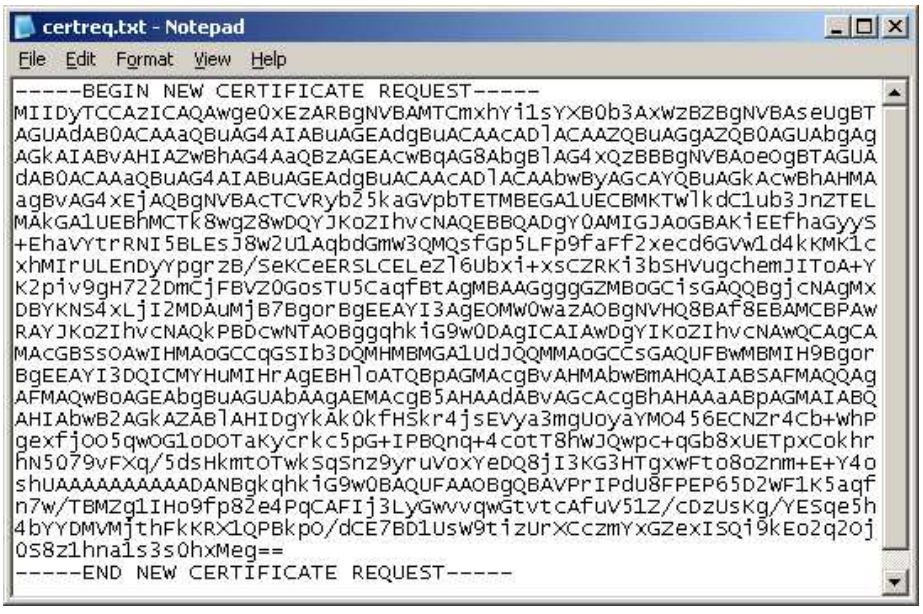


Figur 58: SSL hjelper. Steg 9

Dermed er stegene for for generering av en sertifikatsforespørselen ferdig. Klikk "*Finish*" for å fullføre:

Figur 59: SSL hjelper. Steg 10

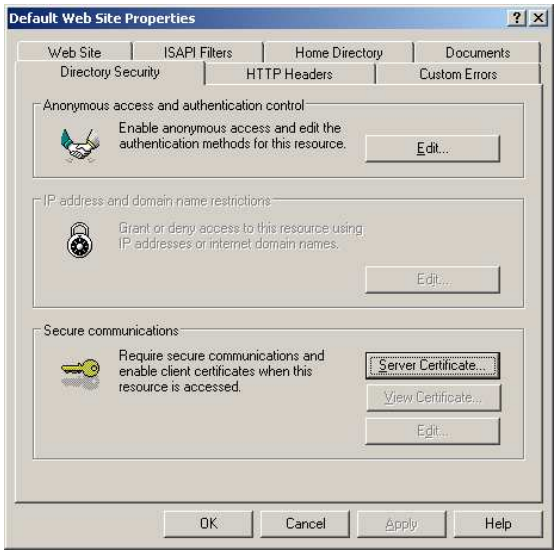
For å se hvordan en slik forespørsel ser ut kan vi åpne filen *C:\certreq.txt* i en editor:



Figur 60: Sertifikatsforespørselen i tekstformat

Det neste som må gjøres er å få signert dette sertifikatet av den CA (Certificat Authority). Den mest brukte CA'en er Verisign. Gå til `http://www.verisign.com/` å følg instruksjonene som ligger under `http://www.verisign.com/products-services/security-services/ssl/index.html`. Videre i det-

te eksemplet er det brukt et selvsignert sertifikat [17], noe som vil gjøre at selve teksten i sertifikatet vil variere noe fra et ekte sertifikat, men funksjonaliteten vil være lik. Når en så har fått signert sertifikatet og fått det tilbake fra CA'en åpner vi egenskaper til webtjeneren igjen og klikker på "*Server Certificate*":



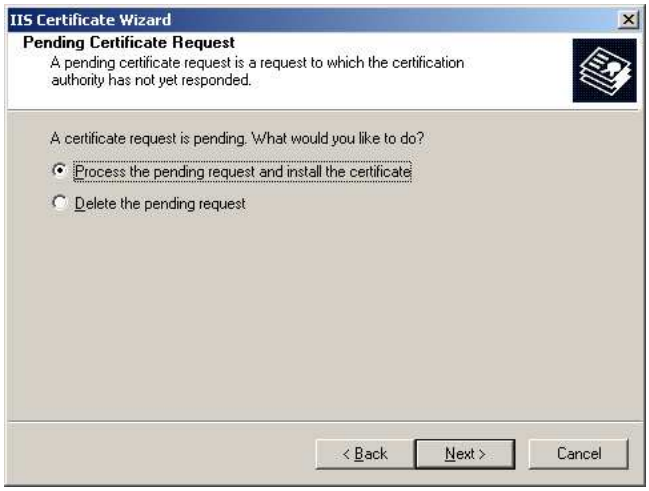
Figur 61: Webtjenerens sikkerhetssenter

Da starter hjelperen på nytt. Klikk så "*Next*":

Figur 62: Sertifikatsinstallasjon. Steg 1

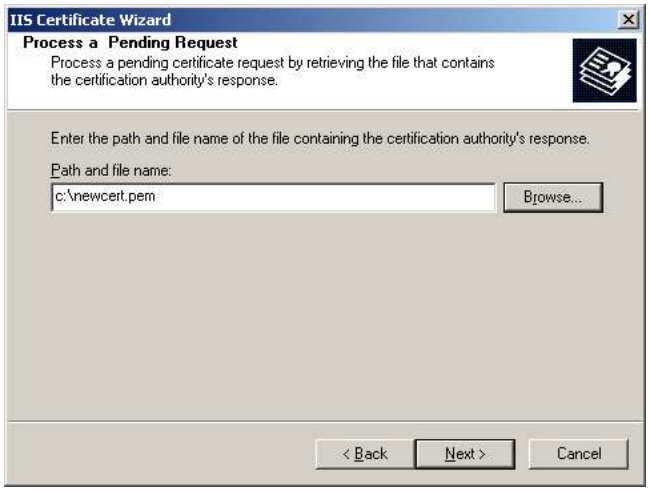
[17] Et selvsignert sertifikat er et sertifikat som vi signerer selv og ikke av en sertifikatsautoritet (CA). Dermed vil ikke dette serifikatet gi noen ekstra sikkerhet for brukeren siden ingen har verifisert denne web-tjeneren.

Den andre dialogen i hjelperen vil nå se lit annerledes ut. Her får du nå valget med å fortsette behandlingen av sertifikatsforespørselen. Kryss derfor av "*Process the pending request and install the certificate*" og klikk deretter "*Next*"*:*



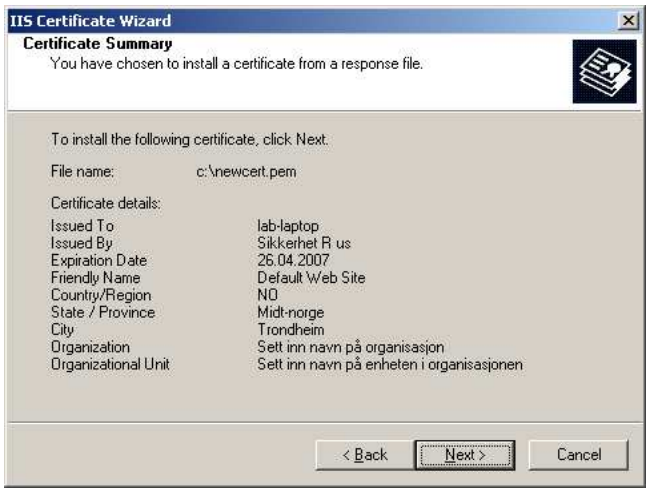
Figur 63: Sertifikatsinstallasjon. Steg 2

Da får du opp en dialog der du må velge den filen vi fikk tilbake fra CA'en. Her i dette eksemplet heter filen *C:\newcert.pem* Klikk deretter "*Next*"*:*



Figur 64: Sertifikatsinstallasjon. Steg 3

Da får du opp en dialog som viser en oppsummering av sertifikatet. Her står det hvem sertifikatet gjelder for, hvem som har godkjent det og hvor lenge det varer. Hvis alt er i orden klikk så *"Next"*:



Figur 65: Sertifikatsinstallasjon. Steg 4

Dermed er hjelperen ferdig. Klikk deretter "*Finish*":

Figur 66: Sertifikatsinstallasjon. Steg 5

For å teste det nye sertifikatet for å se om det virker kan vi nå se på forskjellen mellom krypterte og ikke-krypterte henvendelser mot webtjeneren. Her er et eksempel på å åpne nettsiden på standard ikke-kryptert vis. (Merk at vi her ser på nettsiden gjennom HTTP):

Figur 67: Webside uten SSL

Hvis vi istedet åpner siden med HTTPS, slik *https://129.159.112.31/index.html* i nettleseren vil vi første gang få opp følgende dialog

Figur 68: Nettleseren viser at vi er kommet til en SSL-kryptert tjener

Dette viser at webtjeneren har et sertifikat innstallert og har muligheten til å sende informasjon kryptert. Her i dette eksemplet ser vi at dialogen gir to advarsler samt at den sier at sertifikatet er gyldig. Den første feilmeldingen får du alltid når du for første gang åpner en side på en webtjener som har sertifikater innstallert. Det indikerer at du har ankommet en webtjener du ikke har godkjent og at du må bestemme deg om du vil stole på denne. Den andre feilmeldingen kommer av at jeg i dette eksemplet, med et selvsignert sertifikat, ikke har skrevet inn fullt navn på maskinen i sertifikatsforespørselen. Denne feilmeldingen vil ikke komme hvis sertifikatsforespørselen er riktig utfylt. (CA'en

vil sjekke sertifikatsforespørselen før de signerer den, slik at disse problemene ikke vil oppstå). Hvis vi nå velger "*View certificate*" kan vi se detaljer om sertifikatet:

Figur 69: Studering av sertifikatet fra nettleseren

Her kan vi se mange detaljer om sertifikatet som er innstallert på webtjeneren. Veldig mange brukere har aldri studert disse nå de har åpnet en webside på en webtjener med sertifikat. Dette kan lønne seg å studere, særlig når det gjelder nettbanker og andre sider der man sender fra seg sensitiv informasjon. Til slutt viser vi her hvordan siden ser ut via en SSL-kryptert forbindelse. Det ser ikke ut som den store forskjellen fra en ikke-kryptert siden. Det eneste du ser er at protokollen som benyttes er HTTPS

Figur 70: Webside med SSL

13 Epost sikkerhet / digitale signaturer

(Se kapittel 6 på side 46)

13.1 Når skal du bruke digitale signaturer

Digitale signaturer har i praksis mye av den samme rollen som vanlige underskrifter, nemlig å vise hvem som er opphavet til et dokument. De kan i tillegg brukes til å sjekke om dokumentet er endret siden det ble skrevet. I internettverden vil et dokument ofte være en e-post eller et vedlegg til e-post og digitale signaturer benyttes der til å verifisere at e-posten eller vedlegget virkelig kommer fra den personen den angivelig kommer fra og ikke har blitt endret.

Det er forholdsvis enkelt for en angriper og/eller virus å forfalske (spoofe) e-postadresser slik at det ser ut som om den tilhører noen andre. Det er av og til litt vanskelig å identifisere hvilke e-poster som er ekte, det vil si at de virkelig kommer fra den avsenderen som er oppgitt og at man kan stole på innholdet. Ektheten av innholdet i e-post er kanskje spesielt viktig i korrespondanse av mer privat eller økonomisk natur. I en melding som er digitalt signert kan man også oppdage om innholdet er endret fordi dette vil medføre at signaturen blir feil.

13.2 Eksempel på bruk av digital signatur

1. Generer en nøkkel ved å bruke programvare fra for eksempel PGP (Pretty Good Privacy) eller GnuPG (GNU privacy guard)

2. Nøkkelen din kan forsterkes ved at den signeres av andre som allerede har nøkler. Når nøkkelen din signeres av andre vil dette bekrefte at den tilhører deg og dermed verifiserer de identiteten din og indikerer at de stoler på nøkkelen din.

3. Last opp nøkkel din til en nøkkeltjener slik at de som mottar meldinger med din signatur kan verifisere med den digitale signaturen.

4. Signer de e-postene du sender. De fleste e-postklienter har funksjonalitet for å legge til digitale signaturer. PGP og GnuPG støtter i tillegg kryptering av meldingen ved bruk av nøklene til de som kommuniserer.

13.2.1 Oppsett av PGP for digitale signaturer på Windows XP

Forutsetninger for dette eksemplet er:

- Vi bruker i dette eksemplet PGP 6.58
- Brukeren er logget inn som administrator.
- Maskinen har nett-tilgang

Vi begynner med å laste ned PGP fra `http://www.pgpi.org/cgi/download.cgi?filename=PGPFW658Win32.zip` og unzip'er dette i en temporær katalog. Deretter eksekveres "*setup.exe*" fra denne katalogen. Man kommer da inn i et standard innstallasjonsprogram som innstallerer PGP på maskinen. Når programmet er ferdiginnstallert leter det etter eksisterende pgp-nøkler på maskinen. Hvis den ikke finner det starter den nøkkelgenererings hjelperen:

Figur 71: PGP hjelper. Steg 1

Trykk deretter "*next*" og vi kommer da til en dialog der vi fyller ut fullt navn og e-post adressen til den som skal ha nøklene:

Figur 72: PGP hjelper. Steg 2

I den neste dialogen velger vi hvilke nøkler vi vil generere, RSA eller Diffie-hellman/DSS. Vi lar det her stå på Diffie/Hellman:

Figur 73: PGP hjelper. Steg 3

I den neste dialogen velger vi nøkkellengden vi ønsker. vi lar det også her stå på den som allerede er valgt, 2048 bits og deretter "*next*":

Figur 74: PGP hjelper. Steg 4

I den neste dialogen velger vi varigheten på nøklene. Her velger vi "*Key pair never expires*" og deretter "*next*"*:*

Figur 75: PGP hjelper. Steg 5

I den neste dialogen skriver vi inn passordet vi vil beskytte nøklene med. Dette passordet må være minst 8 tegn langt og må gjentas to ganger. Trykk deretter "*next*":

Figur 76: PGP hjelper. Steg 6

I den neste dialogen genereres selve nøklene. Dette vil ta noen sekunder og når den er ferdig, trykk "*next*":

Figur 77: PGP hjelper. Steg 7

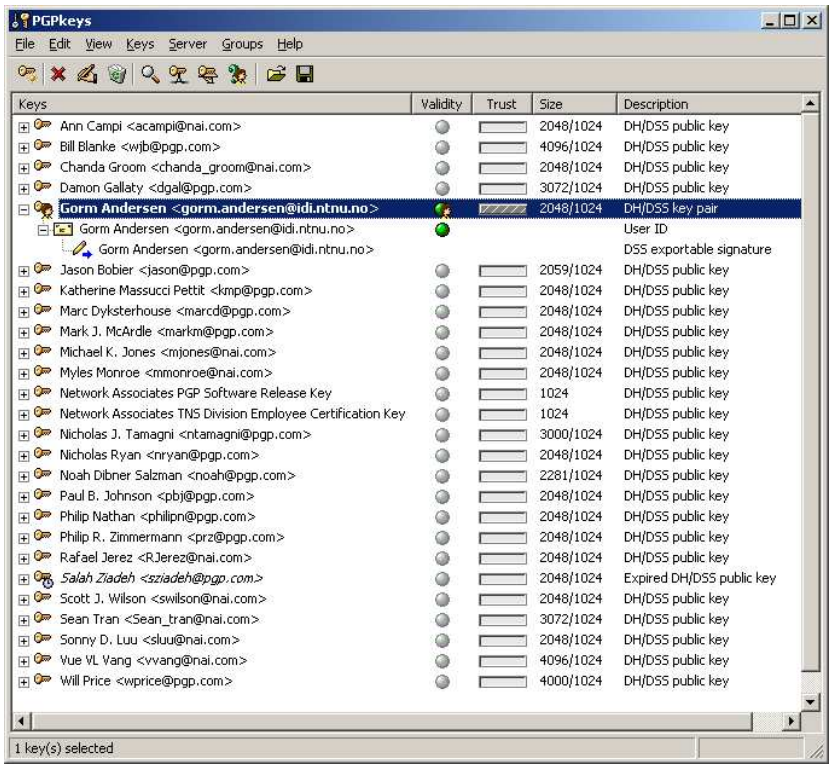
Deretter tilbyr hjelperen å laste opp nøklene som er generert til en nøkkelserver automatisk. Dette kan være problematisk, spesielt hvis du sitter innenfor en brannvegg eller proxy-server. Hvis dette punktet feiler så bruk metoden nedenfor. Trykk deretter "*next*":

Figur 78: PGP hjelper. Steg 8

Dermed har vi generert pgp-nøkler som vi kan bruke til å signere å kryptere meldinger og dokumenter. Klikk "*Finish*" for å avslutte:

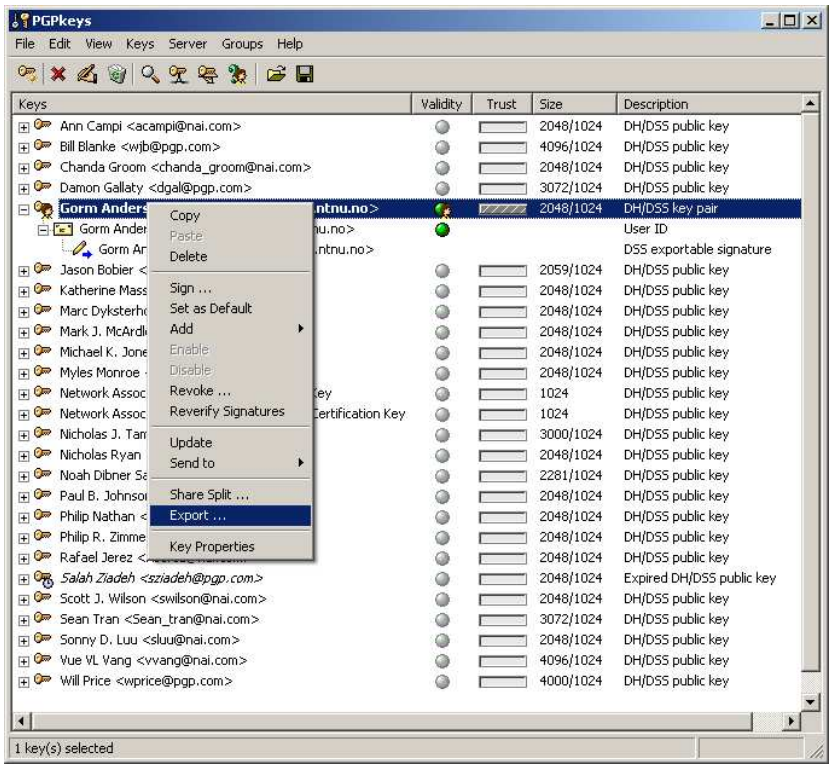
Figur 79: PGP hjelper. Steg 9

Deretter kommer det opp en dialog som viser nøklene dine samt de som har signert den for deg. Første gang vil dette kun være de som står bak PGP (se at de fleste e-post adressene slutter på *pgp.com* og *nai.com)*



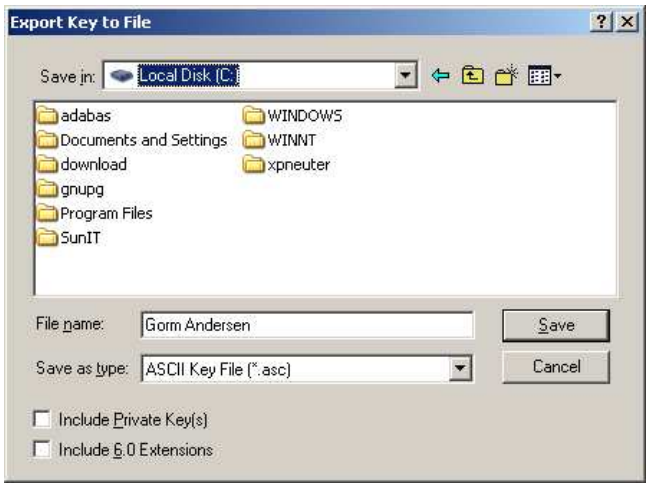
Figur 80: Lokalt tilgjengelige PGP nøkler

Alternativ metode for å laste opp nøklene til en nøkkeltjener er som følger: Høyreklikk på nøklene dine i nøkkeloversikten. Velg deretter "*export*"*:*



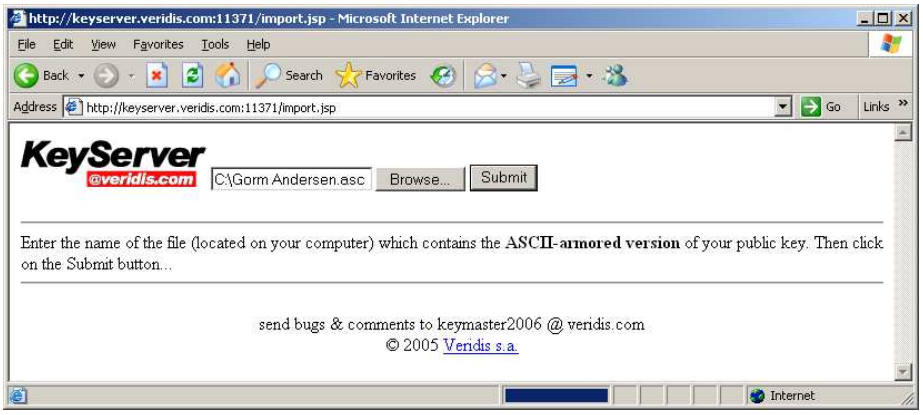
Figur 81: Eksportering av nøkkel

Lagre denne filen på lokal disk med et tilfeldig valgt navn.



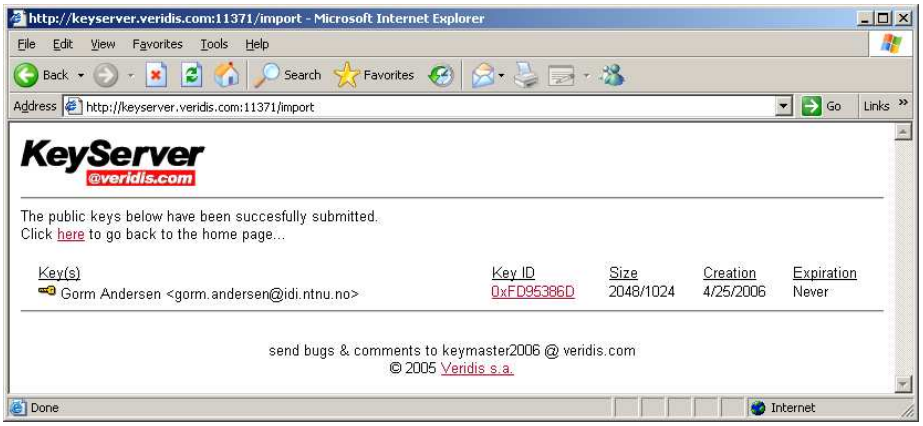
Figur 82: Lagring av den eksporterte nøkkelen

Åpne en nettleser og gå til nettstedet: `http://keyserver.verdis.com:11371/import.jsp`. Velg der "*Browse*" og gå til den filen vi nettopp lagret. (den skal ha endelsen *.asc*) og trykk deretter "*Submit*":



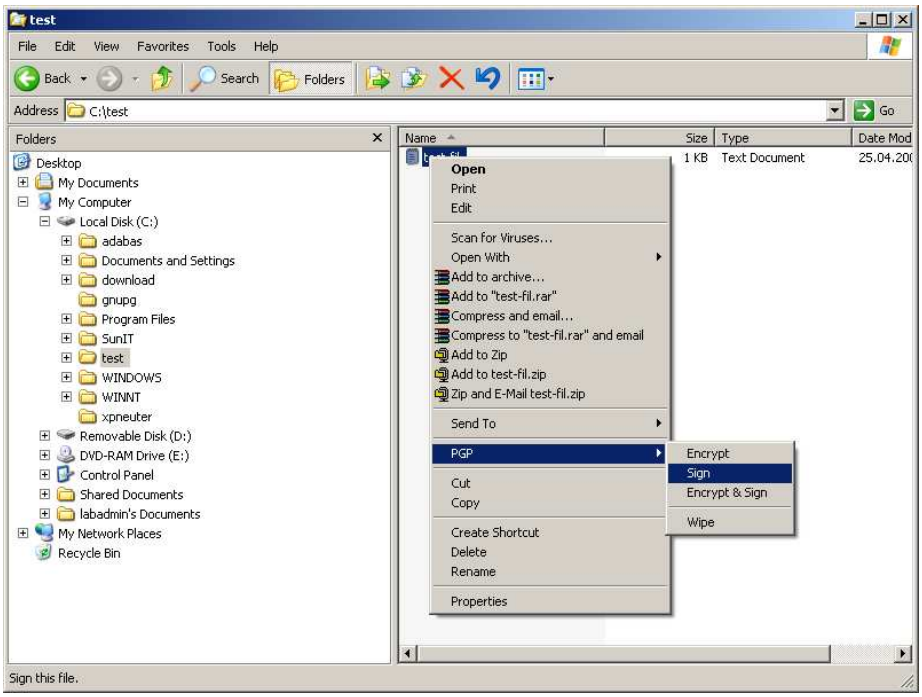
Figur 83: KeyServers hjemmeside

Deretter får du en bekreftelse på at nøkkelen er levert til tjeneren:



Figur 84: Nøklene lastet opp til nøkkeltjener

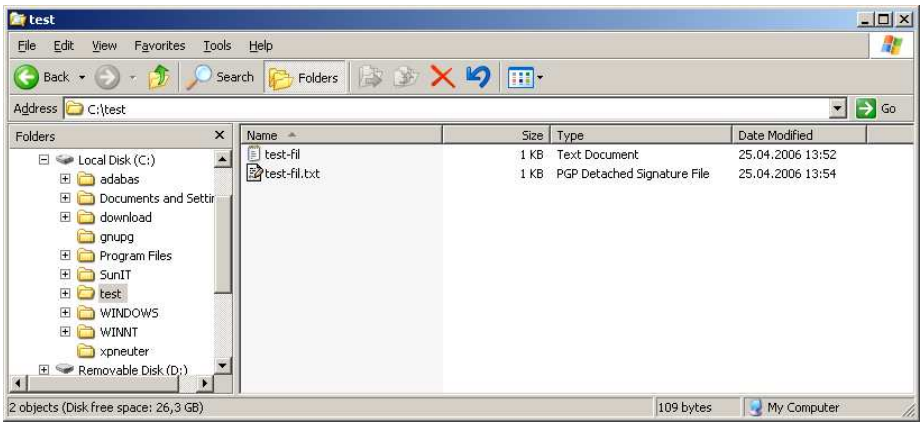
For å vise hvordan dette virker tar vi å oppretter en fil under C:\test\ som heter test-fil.txt
I denne filen skriver vi for eksempel "Dette er en test av signering". Deretter høyreklikker vi på filen, velger *"PGP"* og *"Sign"*.



Figur 85: Signering av dokument

Da kommer det opp en dialog som spør etter passordet til nøklene. Skriv så inn dette passordet og pass på at "*Detach Signature*" er krysset av. Dette vil opprette en ny fil i samme katalog som heter *test-fil.txt.asc* som er meldingsekstraktet av filen kryptert med pgp-nøklene:

Figur 86: Dialog ved signering med PGP

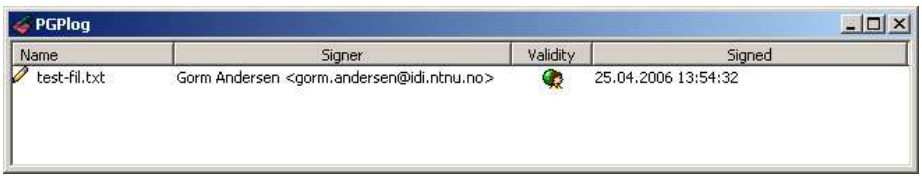


Figur 87: Filer etter signering

Start deretter programmet PGPtools fra "Start" –> "Programs" –> "PGP" –> "PGPtools"
Da vil følgende dialog komme opp:

Figur 88: Kappene til PGPtools

Velg i denne ikon nummer 5 fra venstre (den med åpen hengelås) og velg filen *test-fil.txt*
Hvis filen er uendret siden den ble signert vil dette bli vist:

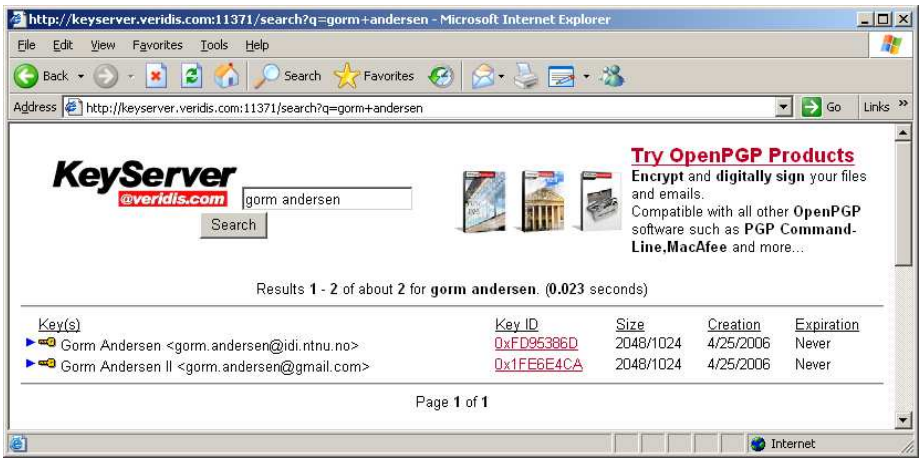


Figur 89: Gyldig signatur

Dersom filen har blitt endret etter den ble signert, vil dette bli vist:

Figur 90: Ugyldig signatur

Hvis vi nå tar et eksempel der vi mottar en fil fra en annen person (Gorm Andersen II) og vil verifisere denne. Det første man gjør da er å finne den offentlige nøkkelen til denne personen. Åpne en nettleser og gå til Keyserver `http://keyserver.verdis.com:11371/import.jsp` og bruk denne personens navn som søkeparametre. Vi finner da i dette eksemplet:

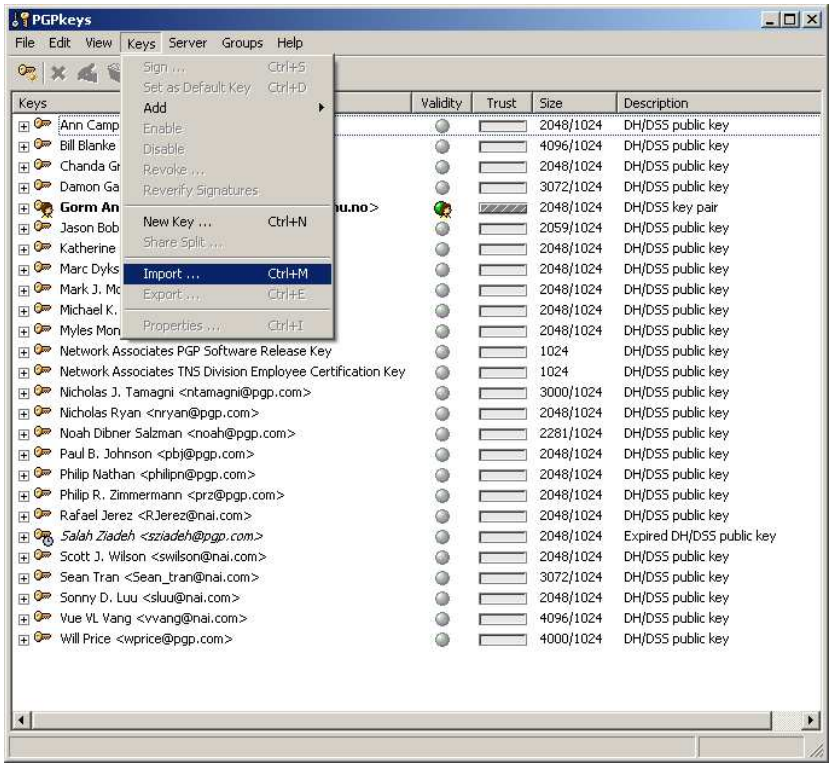


Figur 91: Nedlasting av offentlige nøkler fra nøkkeltjener

Klikk så på den gule nøkkelen på nettsiden og vi får så opp følgende dialog:

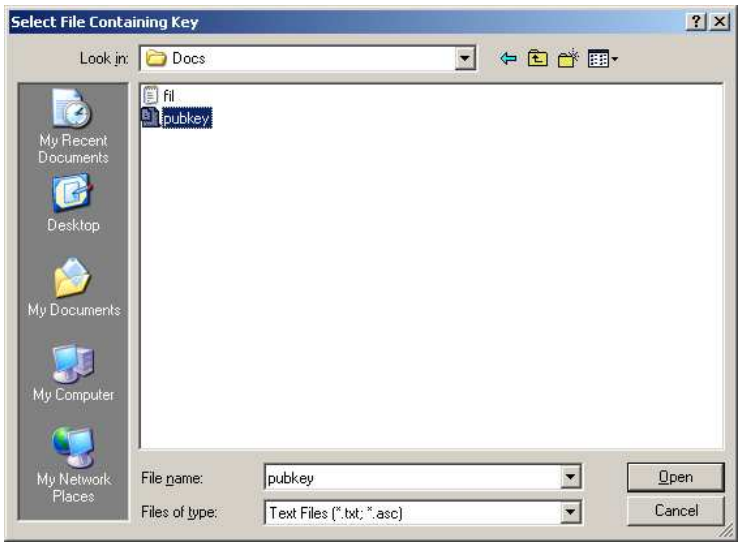
Figur 92: Lagring av andres offentlige nøkler

Lagre så denne nøkkelen til lokal disk. Deretter åpner vi "*PGPkeys*" fra *"PGPtools*" (ikonet med nøklene) og der går vi på menyen "*Keys*" og velger "*import*" *:*



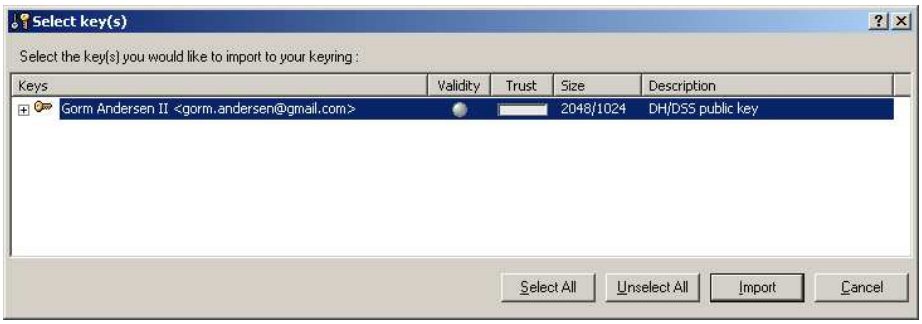
Figur 93: Importering av nøkler fra PGPtools

Deretter velger vi den offentlige nøkkelen vi nettopp lastet ned og trykker "*open*":



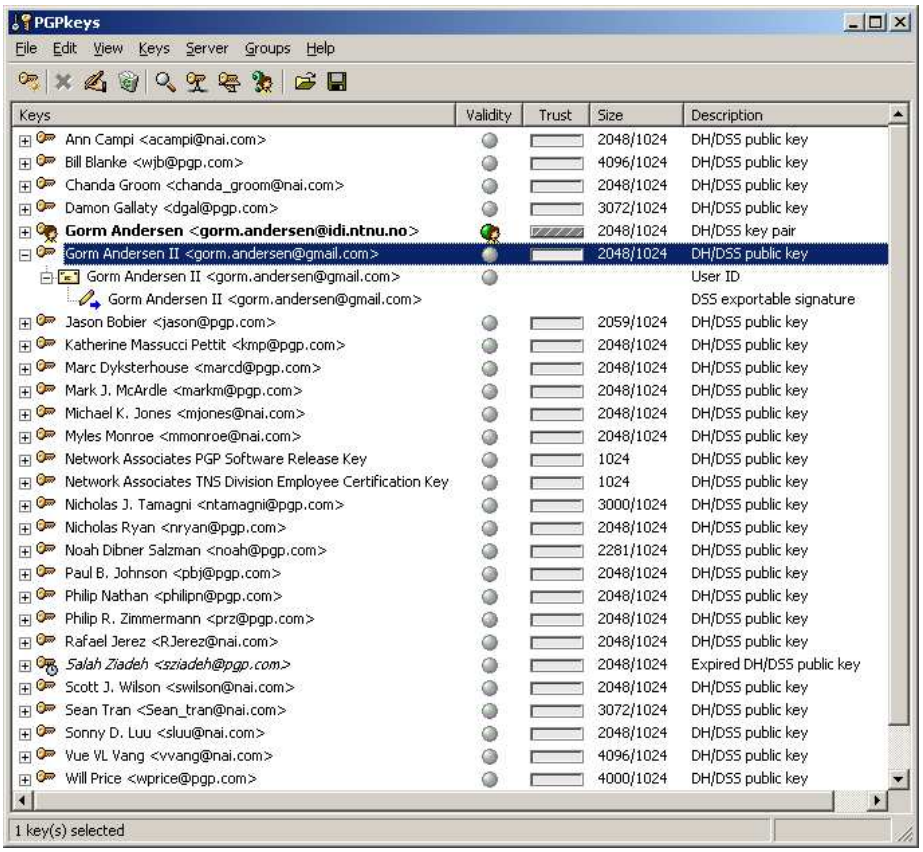
Figur 94: Velg nøkkelen som er lastet ned

Da får vi opp de nøklene vi vil importere. Trykk deretter "*Import*":



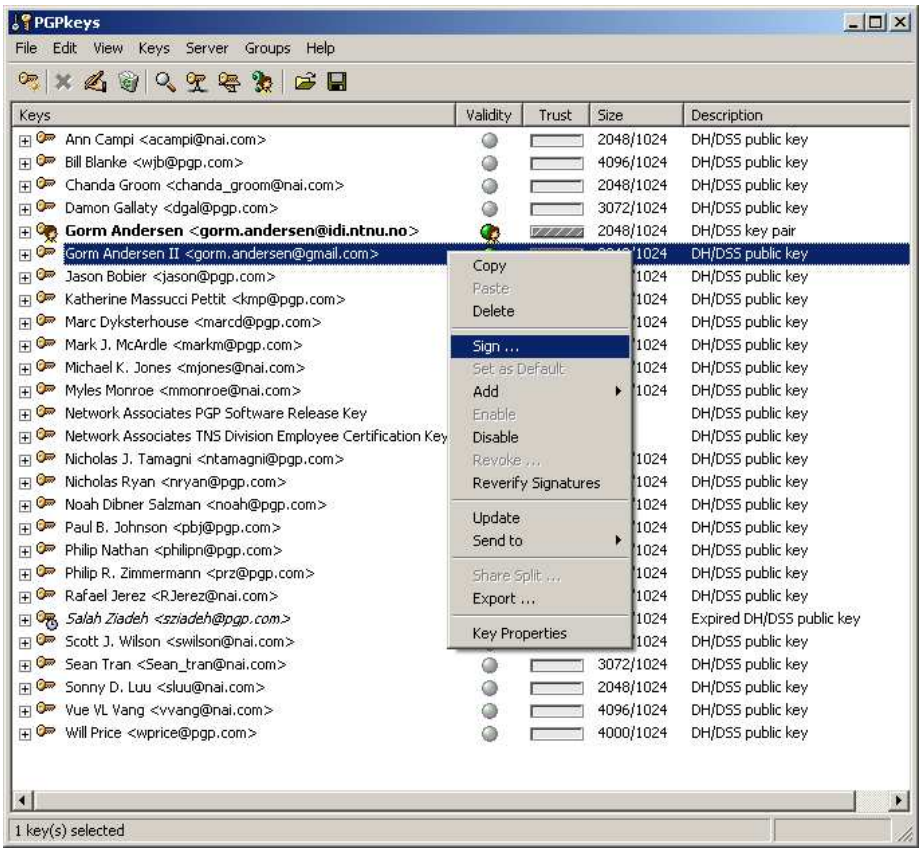
Figur 95: Visning av den importerte nøkkelen

Da får vi opp de nye nøklene i "*PGPkeys*":



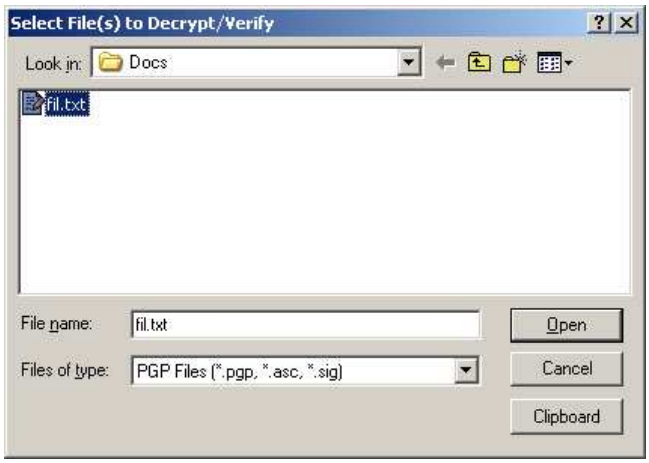
Figur 96: Alle lokalt tilgjengelige nøkler

Deretter høyreklikker vi på de nye nøklene og velger "*sign*" slik at vi viser at vi stoler på nøklene.



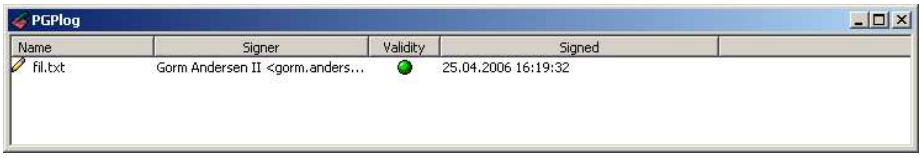
Figur 97: Signering av importert nøkkel

Lukk deretter *PGPkeys*-dialogen og trykk på ikon nummer 5 i *PGPtools* for å starte en verifikasjon. Velg så i denne dialogen den filen vi fikk fra den andre personen:



Figur 98: Åpne tilsendt dokument

Da vil vi se at denne filen er verifisert og at innholdet ikke er forandret siden det ble skrevet:



Figur 99: Gyldig filsignatur

Referanser

Krutz Cole and Conley. *Network Security Bible*. Wiley Publishing, Inc, 2005.

W. Diffie and M. Hellman. New directions in cryptography. In *Proceedings of the AFIPS National Computer Conference*, 1976.

Tom Karygiannis and Les Owens. Wireless network security. Technical report, NIST, National Institute of Standards and Technology, 2002.

Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key size. *Practice and Theory in Public Key Cryptography, PKC 2000*, pages 446–465, 2000.

Michael Palmer. *Guide to Operating Systems Security*. Course Technology, 2004.

William Stallings. *Cryptography and Network Security*. Prentice Hall, third edition edition, 2003.

A APPENDIKS

Dette er et eksempel på oppsett av IPSec mellom to maskiner som kjører operativsystemet Solaris 10. Oppsett av IPSec på Linux vil være forholdsvis likt dette , men det vil være noen forskjeller. For oppsett på linux henvises det til: `http://www.ipsec-howto.org/`

1. Maskinene i eksemplet her er:
   (a) maskin1 som har IP-adresse 10.10.0.1
   (b) maskin2 som har IP-adresse 10.10.0.2
2. Maskinene som skulle kommunisere via IPSec ble lagt inn i:
   (a) /etc/inet/ipnodes
   (b) /etc/inet/hosts (er ikke helt sikker på om det er nødvendig med ipnodes når det bare brukes IPv4)
3. Opprettet ipsecinit.conf på maskinene.
   /etc/inet/ipsecinit.conf
   Med innhold som er forskjellig på hver maskin. Denne definerer hva som skal gjøres avhengig av hvilke hoster maskinen kommuniserer med. Vi bruker delt SA i vårt oppsett.

   [root@maskin1]/: more /etc/inet/ipsecinit.conf {laddr maskin1 raddr maskin2} ipsec {auth_algs any encr_algs any sa shared}
   Forholdsvis selvforklarende: all trafikk fra maskin1 til maskin2 skal gjennom IPSec. Støtter så alle autoriseringsalgoritmer og krypteringsalgoritmer og har delt SA.
4. Laget self-signed Public Key sertifikat.
   ikecert certlocal -ks -m 1024 -t rsa-md5 -D "C=NO-maskin1, O=HADB-maskin1, OU=maskin1, CN=maskin1-ipsec" -A IP=10.10.0.1
   Dette gir et offentlig sertifikat til standard ut og genererer et privat sertifikat i etc/inet/secret/ike.privatekeys/0
5. La inn Public sertifikatene til de maskinene som skulle kommunisere sammen:
   ikecert certdb -a
   (cut & paste inn sertifikatet)
   ikecert certdb -l
   (for å se sertifikatene maskinen har) Eks: maskin1
   [root@maskin1]/: ikecert certdb -l

Certificate Slot Name: 0 Key Type: rsa (Private key in certlocal slot 0) Subject Name: <C=NO-maskin1, O=HADB-maskin1, OU=maskin1, CN=maskin1-ipsec> Key Size: 1024 Public key hash: 18E0F6C01E3D4E614D38B09DB0BB6B4F

Certificate Slot Name: 1 Key Type: rsa Subject Name: <C=NO-maskin2, O=HADB-maskin2, OU=maskin2, CN=maskin2-ipsec> Key Size: 1024 Public key hash: 0433E0D392267512898117B3D4F92791

6. Kofigurerte IKE

/etc/inet/ike/config
Laget en forbindelse fra hver boks til den andre

Eks maskin1:
Explicitly trust the following self-signed certs
Use the Subject Alternate Name to identify the cert
cert_trust "10.10.0.1"
cert_trust "10.10.0.2"
Parameters that may also show up in rules.
p1_xform
{ auth_method preshared oakley_group 5 auth_alg sha encr_alg des }
p2_pfs 5
{
label "maskin1 to maskin2" local_id_type dn local_id "C=NO-maskin1, O=HADB-maskin1, OU=maskin1, CN=maskin1-ipsec" remote_id "C=NO-maskin2, O=HADB-maskin2, OU=maskin2, CN=maskin2-ipsec" local_addr 10.10.0.1 remote_addr 10.10.0.2

7. Reboot maskinen

```
Resultatet av en nettverksniffing mellom de to maskinene:
Sender et ICMP echo fra maskin2 til maskin1
[root@maskin1]: snoop -v maskin2 Using device /dev/bge0 (promiscuous mode)
ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 1 arrived at 15:42:41.72786
ETHER: Packet size = 146 bytes
ETHER: Destination = 0:9:3d:0:87:26,
ETHER: Source = 0:9:3d:0:7d:3,
ETHER: Ethertype = 0800 (IP)
```

```
ETHER:
IP: ----- IP Header -----
IP:
IP: Version = 4
IP: Header length = 20 bytes
IP: Type of service = 0x00
IP: xxx. .... = 0 (precedence)
IP: ...0 .... = normal delay
IP: .... 0... = normal throughput
IP: .... .0.. = normal reliability
IP: .... ..0. = not ECN capable transport
IP: .... ...0 = no ECN congestion experienced
IP: Total length = 132 bytes
IP: Identification = 7548
IP: Flags = 0x0
IP: .0.. .... = may fragment
IP: ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live = 255 seconds/hops
IP: Protocol = 51 (AH)
IP: Header checksum = b840
IP: Source address = 10.10.0.2, maskin2
IP: Destination address = 10.10.0.1, maskin1
IP: No options
IP:
AH: ----- Authentication Header -----
AH:
AH: Next header = 50 (ESP)
AH: AH length = 4 (24 bytes)
AH: <Reserved field = 0x0>
AH: SPI = 0x5dc1fd89
AH: Replay = 11
AH: ICV = eb020c7cce53ae0a8f392301
AH:
ESP: ----- Encapsulating Security Payload ----
ESP:
ESP: SPI = 0x63da0d09
ESP: Replay = 11
ESP: ....ENCRYPTED DATA....
```

Register