Sikring av IP-basert managementtrafikk

Sammenlikning av SNMPv3 USM, IPSec og TLS

av

Ole Richard Valsgård

Hovedoppgave til mastergraden i
informasjons- og kommunikasjonsteknologi

Høgskolen i Agder

Grimstad, Juni 2003

Sammendrag

Sammendrag

Denne hovedoppgaven tar for seg ulike måter å sikre IP-basert nettverksmanagement trafikk. De mest vanlige metodene for nettverksmanagement, Simple Network Management Protocol (SNMP) og telnet, ble i sin tid utviklet uten særlig tanke for sikkerhetsproblematikk. SNMP har derfor vært gjenstand for flere revisjoner med hovedmål å tillegge sikkerhetsmekanismer. Kun den siste utgaven SNMPv3, har lykkes tilfredstillende i dette arbeidet. SNMPv3 brukes imidlertid i svært lite grad da den krever installasjon av dyre og komplekse komponenter på både nettverksutstyr og managementkontrollsentre.

Vi har derfor undersøkt alternativer til SNMPv3, for å sikre SNMPv1 protokolltrafikk med hensyn på konfidensialitet, integritet og autentisering. Disse alternativene er Internet Protocol Security (IPSec) og Transport Layer Security (TLS).

Tre bedømmelsesparametere har blitt lagt til grunn. Disse var sikkerhetsnivået for løsningene, hvor enkle løsningene var å konfigurere, og hvor bra ytelse de leverte.

Når det gjelder sikkerhetsnivå ble dette bedømt ut i fra et teoretisk grunnlag. Løsningene hadde omtrent samme sikkerhetsmekanismer implementert på ulikt nivå. Den største forskjellen lå i nøkkelhåndteringen, hvor SNMPv3 sin tilnærming var helt annerledes og vurdert som svakere enn IPSec og TLS sine løsninger. IPSec hadde i tillegg flere autentiseringsopsjoner enn TLS i sin nøkkelhåndteringsprosedyre.

Konfigurasjonen var kanskje den mest subjektive parameteren. Her falt TLS helt igjennom da denne ikke var mulig å konfigurere for kjernenettverksutstyr. Den ene IPSec implementasjonen var svært vanskelig å installere og gjorde derfor at IPSec kom dårligere ut enn SNMPv3 Security. Selve konfigureringen av SNMPv3 USM var overraskende enkel i forhold til forventningen om at denne skulle være svært kompleks. Den var nesten like grei å konfigurere som IPSec eksklusive installasjon.

Ytelse ble bedømt ut i fra målbare tester på tiden det tok å laste ned en bestemt mengde av data i sammenhengende operasjoner, og er i så måte den mest empiriske parameteren. Her kom IPSec desidert best ut for alle variasjoner av sikkerhetsnivå. TLS kom basert på tidligere publiserte tester relativt bra ut (var ikke mulig å konfigurere), mens SNMPv3 viste seg å være relativt treg i forhold til IPSec, særlig for jo flere sikkerhetsmekanismer som ble lagt til.

Det er mulig at vi ville få et annet ytelsesresultat dersom trafikktypen vi testet for og målene vi testet mot var mer spredt. Det er da tenkelig at vi for IPSec og TLS (hvis konfigurerbart) ville fått mer overhead grunnet stadig forhandling av sesjonsnøkler.

Vi kom til at Internet Protocol Security (IPSec) er den mest effektive løsningen for sikring av IP-basert managementtrafikk. Alternativt anbefaler vi SNMPv3 USM dersom stram aksesskontroll er en viktig parameter, men det er viktig at man da får et stort tap i ytelsen. TLS ble vurdert som dårligste alternativ på grunn av mangel på støtte i relevant utstyr.

Den viktigste konsekvensene av undersøkelsene er at vi har slått fast IPSec som et sterkt alternativ for å sikre SNMP trafikk. Dette gjør at SNMPv1 fortsatt kan brukes til nettverksmanagement dersom den kjøres over en generell sikkerhetsløsning som IPSec, og at managere da slipper å gjennomfør en oppgradering som anses for å være tung og kompleks.

Forord

Forord

Denne rapporten presenterer hovedoppgaven som avslutter mitt studie til mastergraden i informasjons- og kommunikasjonsteknologi ved Høgskolen i Agder (HiA), Fakultet for Teknologi i Grimstad. Prosjektet tilsvarer et halvt års arbeid.

Oppgaven har blitt utført ved Forsvarets Forskningsinstitutt, Avdeling for Elektronikk (FFIE). Problemstillingen kommer fra prosjektet 'Informasjonskrigføring i datanettverk', innen deres forskningsprogram for Elektronisk Krigføring (EK).

Jeg vil benytte anledningen til å takke mine veiledere, forsker Geir Hallingstad og forsker Ronny Windvik ved Forsvarets Forskningsinstitutt, for verdifull hjelp og inspirasjon. Videre vil jeg takke avdelingsingeniør Lars Hornfeldt ved Forsvarets Forskningsinstitutt for tålmodig hjelp med diverse tekniske utfordringer. Jeg ønsker også å takke, høgskoledosent Vladimir Oleshchuk ved Høgskolen i Agder for å håndtere prosjektets primære kontakt med Høgskolen, og studieleder Stein Bergsmark ved Høgskolen i Agder for verdifulle kommentarer under mitt arbeide.

Grimstad 11. juni 2003

Ole Richard Valsgård

Innholdsfortegnelse

Figurliste

For reproduserte figurere merket med forfatter og referanse er det innhentet spesifikk tillatelse.

Tabelliste

1 Innledning

Denne innledningen presenterer bakgrunn, tema og problemstillingen for hovedoppgaven. Videre tar den for seg status for problemområdet, målet med arbeidet og hvordan rapporten er oppbygd.

1.1 Bakgrunn tema og problemstilling

Vi vil her definer hva som menes med management av nettverk, sikkerhetskrav til nettverks management og problemet med oppfyllelsen av disse kravene.

1.1.1 Nettverksmanagement

Nettverksmanagement er et vel innarbeidet begrep som innbefatter konseptene styring, drift og vedlikehold. Siden det Norske språket ikke har noe godt uttrykk for dette, vil vi videre i rapporten bruke begrepet management. Et nettverkssystem består av forskjellige ressurser som endesystemer, mellomliggende systemer og subnettverk.

Management utføres gjennom kontrollering og overvåkning av ressursene som utgjør den aktuelle konfigurasjonen. Overvåkning angår det å observere og analysere status og adferd, mens kontrollering angår det å forandre parametere og fremkalle forandringer [1].

For å utføre management i et distribuert system, er det nødvendig å utveksle informasjon og kommandoer mellom de entiteter som utfører management og de som det skal utføres management på. Denne utvekslingen refererer vi til som management trafikk.

1.1.2 Sikkerhetskrav

For et nettverks management system, som et redskap for kontrollering og overvåking, er det viktig å oppfylle de krav som gjelder for data og nettverkssikkerhet.

Noen av de viktigste kravene er ifølge [2]:

- *Konfidensialitet* – krever at informasjonen i et datasystem ikke er tilgjengelig for lesing av andre en autoriserte entiteter
- *Integritet* – krever at ressurser i et datasystem ikke kan bli modifisert av uautoriserte entiteter uten at dette kan oppdages
- *Tilgjengelighet* – krever at datasystemressurser er tilgjengelige for autorisere parter.

Disse kravene gjelder også for management trafikk, som er fokuset i denne avhandlingen.

1.1.3 Problem

Dessverre er det dårlig samsvar mellom kravene og oppfyllelsen av disse i både spesifikasjoner og implementeringer av mange management systemer [5]. Med dårlige sikkerhetsmekanismer har managementsystemer primært blitt brukt i overvåkningsøyemed. Kontrolleringsfunksjoner har blitt koblet fra for å gjøre dem mindre sårbare, men dette har også gjort dem mindre nyttige. Andre mindre skalerbare løsninger som lokal- eller fjerninnlogging har blitt brukt isteden i konfigurasjonsøyemed.

1.2 Status

Den mest brukte nettverksmanagement løsningen er fortsatt Simple Network Management Protocol (SNMP). Men også Telnet protokollen er mye brukt for konfigurasjon via remote login.

Med basis i SNMP vil vi studere mulighetene for å forbedre sikkerheten til IP-basert nettverks management trafikk

SNMP har blitt revidert flere ganger for å utbedre mangelen på sikkerhetsmekanismer [3, 4]. Den første offisielle versjonen (SNMPv1) ble designet uten at det ble tatt noen større sikkerhetshensyn, da dette skjedde før vi fikk den massive bruken av distribuerte systemer vi har i dag. Den andre versjonen (SNMPv2) var et forbedringsforsøk som mislykkes i å implementere den planlagte sikkerhetsdelen. Til sist lyktes det i den tredje offisielle iterasjonen (SNMPv3) å implementere relativt gode sikkerhetsmekanismer. Men denne er fortsatt i liten grad tatt i anvendelse.

Det er flere grunner til dette, noen av dem kan være:

1. Ikke alle produsenter av nettverksutstyr har implementert denne nye versjonen
2. Det er mangel på, eller dyrt å oppgradere til, støtte for denne i overordnede management verktøy som f.eks. HP OpenView og andre.
3. Den har fått et rykte på seg for å være kompleks å konfigurere og kontrollere [5]
4. Nettverks administratorer finner det enklere å la være å oppgradere siden den første versjonen fortsatt virker, dog uten tanke for sikkerhetsbetraktninger.

Ved 'IFIP/IEEE Internasjonale Symposium om Integrert Nettverks Management i Seattle i Mai 2001 [5], ble det foreslått to alternative løsninger for sikring av SNMP protokollen. Den første var å kjøre SNMP trafikk over Transport Layer Security (TLS) [6] og den andre var å kjøre SNMP trafikk over Internet Protocol Security (IPSec) [5, 7]. Den første var testet og dokumentert og ble presentert som et bidrag til konferansen. Den andre ble i korthet nevnt som en mulig løsning, men var verken testet eller dokumentert.

På konferansen ble det også brakt på det rene at mange brukte Telnet protokollen for remote innlogging. Det ble påpekt at dette var en dårlig sikret løsning da denne f.eks sender innloggingspassordet i klartekst over nett. Et forslag om å alternativt bruke Secure Shell (SSH) ble da godt mottatt. Imidlertid mangler det kunnskap om omfanget av støtte for denne i sentralt nettverksutstyr.

1.3 Målet med arbeidet

Målet med arbeidet er å utforske hvordan man kan sikre konfidensialitet, integritet og autentisering av managementtrafikk i IP-baserte datanettverk.

Vårt fokus vil være Simple Network Management Protocol (SNMP), og vi ønsker å evaluere de tre alternativene SNMPv3 Security[1], Internett Protokoll Security (IPSec) og Transport Layer Security (TLS) for sikring av denne. Parametrene for evalueringen er nivå av sikkerhet, hvor enkle løsningene er å konfigurere og hvor god ytelse disse gir.

I tillegg vil vi ta en kort titt på remote innlogging, nærmere bestemt hvorvidt sentralt nettverksutstyr søtter Secure Shell (SSH) som et alternativ til Telnet.

[1] Med SNMPv3 Security menes de to sikkerhetsmodulene User-based Security Model (USM) og View-based Access Control Module (VACM).

1.4 Oppbygningen av rapporten

De resterende delene av rapporten er bygd opp på følgende mate:

Den første delen gir en teoretisk bakgrunn for oppgaven. Den tar for seg nettverksmanagement generelt og spesifikt for Simple Network Management protokollen, introduserer en sikkerhetsmodell og angir hvordan oppgaven skal angripes.

Den andre delen presenterer løsningene for å sikre nettverks management trafikk. Disse kapitlene beskriver løsningene og deres sikkerhetsnivå, hvor enkle de er å konfigurere og ytelsestest med resultater. Det er ett kapittel for hver hovedløsning og ett kollektivt kapittel med andre alternativer.

Til sist i del tre presenteres evalueringen av hovedløsningene. Resultatene sammenliknes og diskuteres, før konklusjonen for hele arbeidet trekkes.

2 Nettverksmanagement

Dette kapittelet er ment å gi litt teoretisk bakgrunn for oppgaven. Først introduserer vi nettverksmanagement og ser på det veldig generelt, så ser vi nøyere på SNMP protokollen.

2.1 Introduksjon

I denne delen vil vi se på hensikk, funksjonsområder og tilnærminger til nettverksmanagement.

2.1.1 Hensikt

Hensikten med nettverks management er å sørge for at systemet leverer det som det er spesifisert for. Dette blir oppnådd gjennom de to hovedoperasjonene overvåkning og kontrollering. Overvåkning dreier seg om å observere og analysere status og adferd, mens kontrollering dreier seg om å modifisere parametere og sørge for at handlinger utføres.

2.1.2 Funksjonsområder

Nettverksmanagement kan deles inn i flere funksjonsområder. ISO har trukket frem og standardisert fem hovedfunksjoner [8], som har oppnådd bred aksept:

- *Feilhåndtering* – deteksjon, isolasjon og korreksjon av uregelmessigheter og feil i nettverk
- *Konfigurasjonshåndtering* – registrere og vedlikeholde nettverkskonfigurasjon, og oppdatere konfigurasjonsparametere for å sikre normal operasjon av nettverket
- *Kontohåndtering* - brukerhåndtering og administrasjon, fakturering for bruk av nettverksressurser og tjenester.
- *Ytelseshåndtering* – besørge påliteligheten og kvaliteten av nettverksytelsen. Dette inkluderer tjenestekvalitetstilbud og regulering av parametere som gjennomløp, utnyttelsesgrad, forsinkelse, metningsnivå og pakketap.
- *Sikkerhetshåndtering* – besørge beskyttelse for alle sikkerhetstrusler mot nettverksressursene, deres tjenester og data. I tillegg å også ivareta konfidensialitet og kontrollere aksessrettigheter.

Ytelses-, feil- og kontohåndtering er primært *overvåkningsrelaterte*, mens konfigurasjons- og sikkerhetshåndtering primært er *kontrolleringsrelatert*

2.1.3 Tilnærminger

Managementsystemer har utviklets seg i takt med utviklingen av internett som går helt tilbake til ARPANET [9]. Med en rivende utvikling i størrelse, kompleksitet og tjeenstekrav til nettverkssystemers infrastruktur, har også tilnærmingene til måter å drive management på utviklet seg. En klar trend er en økende distribusjon av intelligensen i management systemene [10]. Vi vil her se på noen av tilnærmingene.

De enkleste metodene er vanlig *terminal- eller fjerninnlogging*. Dette ikke noen selvstendig management løsning, men likevel en mye brukt metode selv for management operasjoner. I dette tilfellet må man for hver enkelt entitet gjennomføre en egen innlogging for å kunne utføre kontrollerings og overvåkningsoppgaver. Telnet er et eksempel på en protokoll for fjerninnlogging.

Den andre tilnærmingen er å *aksessere en flat database* hos en agent for å lese og sette parametere i hver enkelt nettverksentitet fra ett eller flere sentrale punkter i nettverket. Med denne metoden kan det drives management på flere nettverksentiteter samtidig. Dette er en veldig forbedring fra det å separat logge inn på hver enkelt entitet. Simple Network Management Protocol (SNMP) er et eksempel på en slik løsning.

En tredje tilnærming er å bruke en mer *objektorientert database* på nettverks entitetene til de samme oppgavene. Dette muliggjør å kjøre funksjoner på objekter i entitetene. Et eksempel på en slik løsning er Telecommunication Management Network (TMN) [11].

En fjerde tilnærming er *policy-basert nettverks management*. Hovedtanken med dette er at regler styrer tilstander og oppførselen til nettverkssystemet. Regler omformes slik at de blir syntaktiske og verifiserbare og deretter igjen oversettes til entitetsavhengige konfigurasjoner. Disse distribueres til og håndheves så av gjeldende entiteter.

En femte tilnærming er en *distribuert prosessering*. Dette realiseres ved å bruke objektorienterte metoder for å lage distribuerte applikasjoner. En slik løsning kan f.eks. bruke Common Object Request Broker Architecture (CORBA).

Web-basert management er også en mulig realisasjon. Vi har tre typer av dette. Først har vi systemer som kun oversetter HTTP trafikk til andre management protokoller (f.eks. SNMP/CMIP). Dernest har vi web-baserte entitetsservere som prosesserer entitetsdata, bygger en HTML/XML presentasjon av disse og overfører dette. Til sist har vi web-baserte management plattformer, som har sin egen management protokoll, datamodell og arkitektur. Et eksempel på den sistnevnte er Web-Base Enterprise Management (WEBEM) [12].

Java-basert management har deltatt i flere paradigmer av nettverks management, fra distribuert prosessering, via web-basert management til intelligente agenter (se under). Det er også en lovende teknologi for mobil kode (se under).

Det er også mulig å bruke *mobil kode* [13] for nettverks management. Management oppgaver må da ikke lenger må gjøres sentralt. Sentrale entiteter bare lager mål og utkast til prosedyrer for oppgaver. Vi kan definere tre paradigmer for mobil kode [14], basert på samhandlingen mellom tjenester og ressurser. Dynamisk nedlasting av kode på forespørsel (Code On Demand), dynamisk opplasting av kode til agent med resultat tilbake (Remote Evaluation), og relokasjon av hele prosesserende komponenter (Mobile Agent).

Videre ser vi nå også begynnelsen til en løsning kalt *intelligente agenter* [15]. Disse er uavhengige entiteter som kan utføre komplekse handlinger og løse management problemer på egenhånd, og trenger kun å vite mål på høyere nivå. Vi har tre arkitekturer for intelligente agenter [15]: *rådslående agenter* som opererer på et sett av fysiske symboler for å lage overordnede intelligente handlinger, *reaktive agenter* som utfører handlinger basert på observasjoner av omgivelsene, og *hybride agenter* som er en blanding av de to.

Til sist har vi *aktive nettverk* [16, 17] hvor nettverksnoder som rutere og svitsjer utfører tilpassede beregninger over meldinger som flyter igjennom dem. Tilpassede tjenester for dette kan lastes ned dynamisk fra kodeservere (mobil kode) eller aktive pakker. To tilnærminger for dette eksisterer: nedlastning på en egen dedikert linje, eller nedlastning på trafikklinjen. Den siste medfølger store sikkerhetsutfordringer som ennå ikke har fått noen løsning.

Som vi ser er det en klar trend at intelligens flyttes fra sentrale entiteter og ut i de opererende nettverksentitetene. Likevel er løsninger som mobil kode, intelligente agenter og aktive nettverk fortsatt på et tidlig stadium, og har fortsatt en del uløste problemer særlig på sikkerhetssiden. Tradisjonelle løsninger som SNMP vil sannsynligvis fortsatt være aktuelle i lang tid.

2.2 Arkitektur

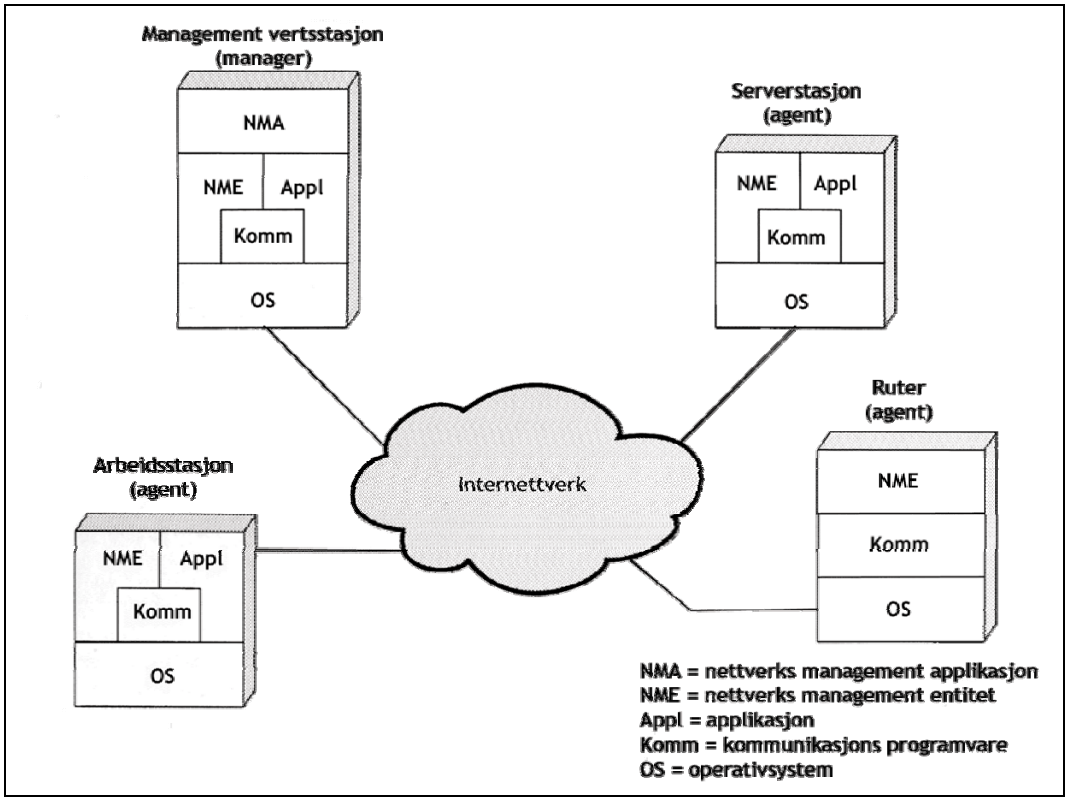
Vi vil i dette avsnittet se på den generelle arkitekturen til et tradisjonelt management system. Vi har delt denne gjennomgangen opp i systemelementer og programvare.

Et nettverksmanagementsystem er en samling av verktøy for nettverksovervåkning og -kontroll. Det består av hardware- og softwaretillegg, implementert i eksisterende nettverkskomponenter. Det har et enkelt brukergrensesnitt med et kraftig og brukervennlig kommandosett for å utføre management oppgaver.

Management systemet er utformet for å se hele nettverket som en enhetlig arkitektur, med adresser og etiketter utpekt for hvert punkt og spesifikke atributter for hvert element og hver link kjent for systemet. De aktive elementene i nettverket gir regelmessig tilbakemelding om status informasjon til nettverkskontroll senteret.

2.2.1 Systemelementer

Et tradisjonelt management system er bygget opp av to entitetstyper: agent og manager. Figur 1 viser sammenhengen mellom disse i et eksempel med fire forskjellige entiteter koblet sammen.



Figur 1: Elementer i et nettverks management system, Stallings [1]

Hver entitet inneholder en samling av dedikert nettverksmanagementprogramvare kalt en nettverksmanagemententitet (NME). I tillegg har minst en av entitetene også en funksjon som nettverksmanagement kontrollsenter, og innehar en samling av dedikert programvare kalt en nettverksmanagementapplikasjon (NMA). Entiteter som kun har NME programvare kalles for agenter, mens entiteter som i tillegg har NMA programvare kalles for managere.

Nettverksmanagemententitetene (NME) samler inn statistikk om kommunikasjon og nettverksrelaterte aktiviteter. Dette lagres lokalt i en database. Videre gjennomfører de ordrer fra nettverkskontrollsentret. Dette inkluderer ordrer om: overføring av innsamlede data, forandring av parametere, avlevering av statusinformasjon og generering av testtrafikk. De sender også

meldinger til nettverkskontrollsenteret med informasjon om endring av viktige parametere. I tillegg kan de også fungere som 'proxier' ovenfor kontroll senteret, for noder som har proprietære løsninger eller av andre grunner (f.eks. ytelse) ikke kan inneholde en NME. Proxien lagrer og vedlikeholder da statistikk på vegne av noden.

Nettverksmanagementapplikasjonen (NMA) gjennomfører brukerkommandoer forå vise informasjon fra nettverksmanagemententitetene, eller iverksette kommandoer til disse ut gjennom nettverket. Videre alarmerer applikasjonen brukere om mottatt informasjon om vesentlige endringer i entitetene.

All kommunikasjon mellom managere og agenter foregår over management protokoller på applikasjonsnivå, som anvender kommunikasjonsarkitekturen på samme måte som enhver annen distribuert applikasjon.

Isteden for den ovenfor skisserte ordningen, er det også mulig å ha en *distribuert tilnærming* til nettverks management. Dette gjennomføres ved å ha en overordnet NMA pluss en backup, med flere NMA-er imellom seg og NME-ene. De mellomliggende NMA-ene fungerer som 'proxier' og gir begrensede rettigheter til lokale administratorer. Den overordnede NMA-en har globale rettigheter og kan forvalte alle nettverksressursene via proxiene. Det er flere fordeler med en slik tilnærming; mye trafikk avgrenses lokalt og overhead unngås, systemet får bedre skalerbarhet og utvidelser er både billig og relativt enkel, 'single-point-of-failure' unngås.

2.2.2 Programvarearkitektur

Arkitekturen til programvaren for nettverks management i managere og agenter varierer med funksjonaliteten til plattformen og med omfanget og kvaliteten på tjenestene.

Vi har tre hovedgrupper av management programvare [1]:

- Brukerpresentasjonsprogramvare
- Nettverksmanagementprogramvare
- Kommunikasjons- og databasestøtteprogramvare

Brukerpresentasjonsprogramvaren
Et enhetlig brukergrensesnitt for interaksjon mellom brukere og programvare muliggjør en enkel kontrollering og overvåkning. For å unngå å overlesse brukeren med informasjon, bør det ha verktøy for å organisere, oppsummere og forenkle informasjons presentasjonen.

Nettverksmanagementprogramvare
Nettverksmanagemnetprogramvare er en samling av programvare som tilbyr de tjenestene som brukerne behøver.

Denne programvaren kan igjen kommunisere med flere applikasjonselementer. Disse elementene implementerer enkle verktøy som brukes av en eller flere av applikasjonene.

Under applikasjonselementene har vi en transporttjeneste for nettverksmanagementdata. Denne modulen består av en nettverksmanagementprotokoll for å utveksle managementinformasjon mellom managere og agenter, og et tjenestegrensesnitt mot applikasjonselementene. Tjenestegrensesnittet gir typisk funksjoner for å innhente informasjon, sette informasjon og generere varsler.

Kommunikasjons- og databasestøtteprogramvare
Nettverks management programvaren trenger aksess til en lokal managementinformasjonsbase (MIB) og remote agenter og managere for å utføre sine påtenkte funksjoner.

En lokal MIB i en agent inneholder nettverksmanagementinformasjon, inklusive konfigurasjon og adferdsreflekterende informasjon og kontrolleringsparametere. En lokal MIB i en manager inneholder i tillegg oppsummeringsinformasjon om agenter under dens kontroll. En lokal MIB aksessmodul må inneholde programvare for grunnlegende filhåndtering for å kunne aksessere

MIB-en, og kan i tillegg også trenge å konvertere informasjon til formater standardisert over nettverksmanagementsystemet.

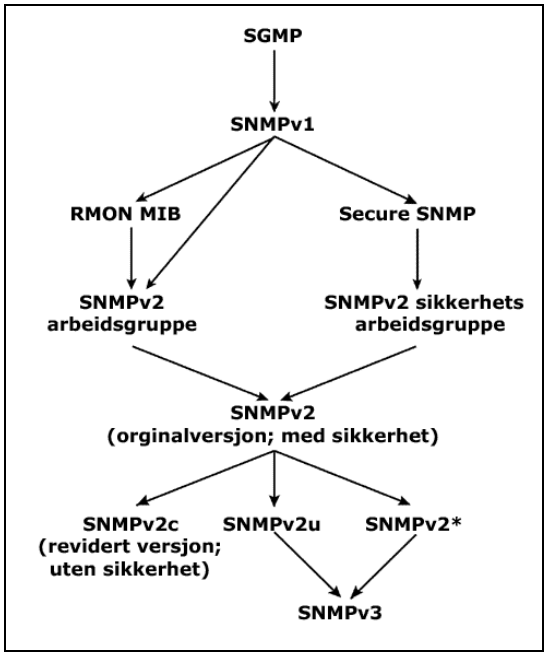
Kommunikasjonen mellom nodene (root-managere, managere, agenter og proxier) er avhengig av en kommunikasjonsprotokollstakk, som bærer av nettverksmanagementprotokollen som opererer på applikasjonsnivå.

2.3 Simple Network Management Protocol

I dette kapittelet vil vi se nærmere på Simple Network Management Protocol (SNMP). Først ser vi litt på utviklingen av protokollen og dens ulike versjoner. Så tar vi for oss litt generelt om lagdeling og operasjonene. Deretter ser vi på den overordnede arkitekturen og meldingsformatene for de tre versjonene.

2.3.1 Utvikling

Simple Network Management Protocol (SNMP) har utviklet seg (Figur 2) til tre offisielle iterasjoner kjent som SNMPv1, SNMPv2c og SNMPv3.



Figur 2: Utviklingen av SNMP, Stallings [1]

Forstadiet til SNMP var Simple Gateway Monitoring Protocol (SGMP) [18], definert høsten 1987. SGMP var det første spesifikke management verktøy, og gav enkle operasjoner for management av gatewayer.

I 1988 bestemte Internet Architecture Board (IAB) seg for å utvikle et forslag kalt Simple Network Management Protocol (SNMP), som en kortsiktig løsning for TCP/IP basert management i påvente av at TCP/IP installasjoners overgang til OSI baserte protokoller, og som en erfaringsbase for utvikling av management protokoller

Arbeidsgruppen for Simple Network Management Protocol (SNMP) [19] presenterte sitt resultat i mai 1990 [20, 21, 22, 23]. SNMP var fundamentert på SGMP og gav enkle men kraftige metoder for overvåkning og kontrollering av nettverkselementer ved hjelp av strukturer av management informasjon (SMI), management informasjonsbaser (MIB) og protokoll. Ettersom senere versjoner ble definert ble denne kalt SNMPv1.

SNMPv1 hadde imidlertid en del mangler. En av dem var mangelen på muligheter for å overvåke nettverk, ikke bare entiteter. Dette ble håndtert av arbeidsgruppen for 'Remote Network Monitoring' [24] som i november 1991 utvidet definisjonen for informasjonsbasen til SNMP (MIB) til også å dekke nettverk [25]. En annen svakhet var total mangel på sikkerhetsinnretninger. Arbeidsgruppen for 'Secure SNMP' [26] publiserte i juli 1992 et forsøk på å utbedre dette [27, 28, 29]. Denne definerte noe som het 'party-based security module'. Begge disse forbedringene var med som input i definisjonen av neste generasjon av SNMP.

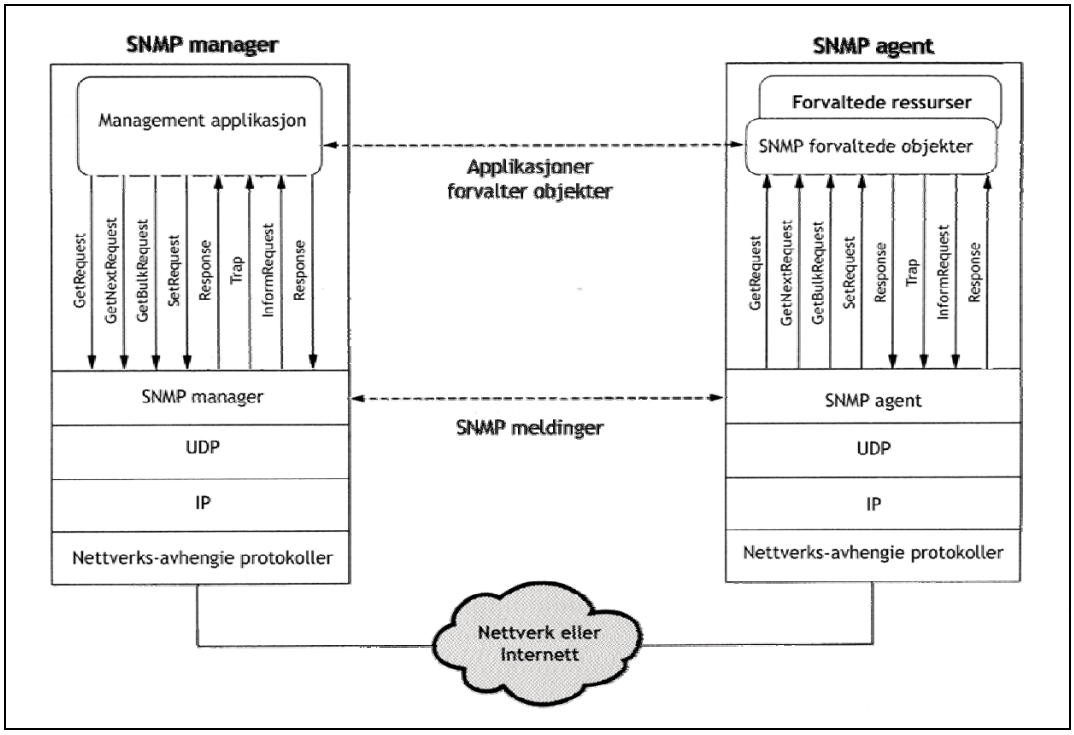
Arbeidsgruppene for andregenerasjons SNMP (SNMPv2) [30], var delt opp to deler: en for sikkerhetsaspekter og en for alt annet. Det samlede resultatet ble fremlagt i mars 1993. På grunn av manglende overensstemmelse om hva SNMPv2 sikkerhet skulle innebære, bortfalt imidlertid sikkerhetsdelen (party-basert) i en revidering i 1996 [31, 32, 33, 34, 35, 36, 37, 38]. Denne nye revisjonen ble kalt community-basert SNMPv2 eller SNMPv2c, siden man gikk tilbake til en community basert meldingshåndtering fra SNMPv1.

For å utbedre sikkerhetsmanglene i SNMPv2c, begynte uavhengige grupper å arbeide på sikkerhetsutvidelser til SNMPv2c. To konkurrerende tilnærminger stakk seg ut som favoritter: SNMPv2u [39, 40] og SNMPv2*(Internet Draft, 1995). Disse to tilnærmingene tjente som input til neste generasjon SNMP.

I mars 1997 ble arbeidsgruppen for SNMPv3 [41] opprettet. Den publiserte et sett av dokumenter [42, 43, 44, 45, 46, 47] i januar 1998. Dokumentene definerer et rammeverk for å innlemme sikkerhetstjenester i en overordnet kapabilitet som enten inkluderer SNMPv1 eller SNMPv2c (standard) funksjonalitet. I tillegg spesifiserer dokumentene spesifikke sett av kapabilteter for nettverkssikkerhet og tilgangskontroll.

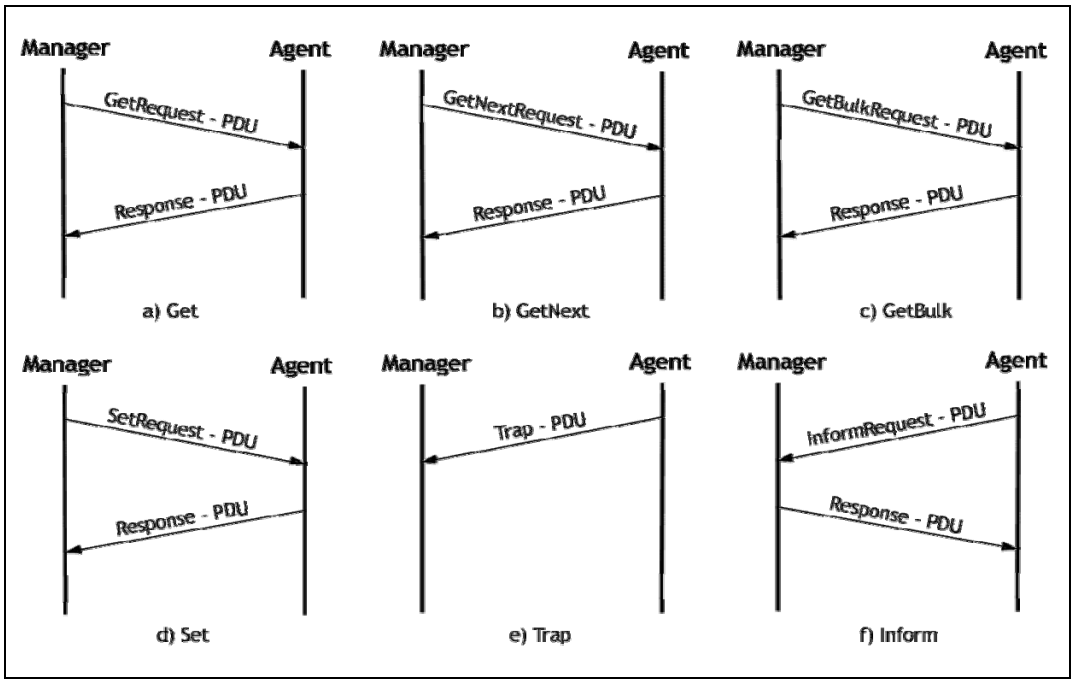
2.3.2 Generelt

SNMP opererer som managere og agenter. Figur 3 viser sammenhengen mellom disse to på ulike kommunikasjonsnivå.



Figur 3: SNMP roller, Stallings [1]

I SNMP agenten har vi en del forvaltede ressurser og parametere, hvis status avbildes i objekter i en database. Management applikasjonen i manageren opererer mot disse objektene med en logisk syntaks. Disse operasjonene utføres over SNMP meldinger mellom manager og agent programvaren. SNMP meldingene kommuniserer over en UDP/IP forbindelse.



Figur 4: SNMP operasjoner, Stallings [1]

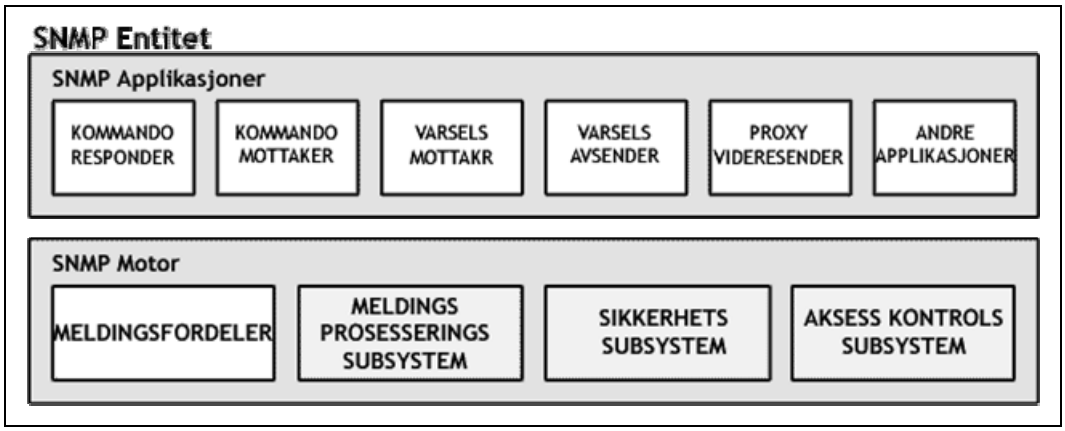
Figur 4 illustrerer operasjonene som protokollen kan utføre. De tre første operasjonene omhandler innhenting av informasjon (objektverdier) fra en agent, henholdsvis innhenting av enkeltobjekter, suksessive objekter, og større grupper av objekter. Den fjerde går på å sette verdier for ressurser (objekter representerer ressurser), mens de to siste operasjonene går på varsling fra en agent til en manager om hendelser, henholdsvis uten og med tilbakemelding.

Objektene som det opereres på ligger i en informasjonsdatabase kalt managementinformasjonbase 2 (MIB2). Syntaksen på objektene følger Advanced Syntax Notation 1 (ASN.1), og er bygget opp med en hierarkisk struktur som muliggjør leksografisk gjennomgang av basen.

2.3.3 Arkitektur

Et rammeverk for å beskrive SNMP arkitekturen, ble først definert i SNMPv3 [43]. Dette rammeverket passer imidlertid også for å beskrive de forutgående versjonene.

SNMP entitet
En SNMP entitet (Figur 5) består av en SNMP *motor*, og en eller flere assosierte *applikasjoner*.



Figur 5: SNMP Entitet

Motoren
SNMP motoren er ansvarlig for å forberede meldinger for overføring, pakke ut innkommende meldinger for levering til applikasjoner, og for å gjøre sikkerhetsrelatert prosessering for utgående og innkommende meldinger. Til disse operasjonene har den en meldingsfordeler og subsystemer for meldingsprosessering, sikkerhet og aksesskontroll.

Applikasjonene
Vi har fem forskjellige applikasjoner som besørger ulike managementoperasjoner.

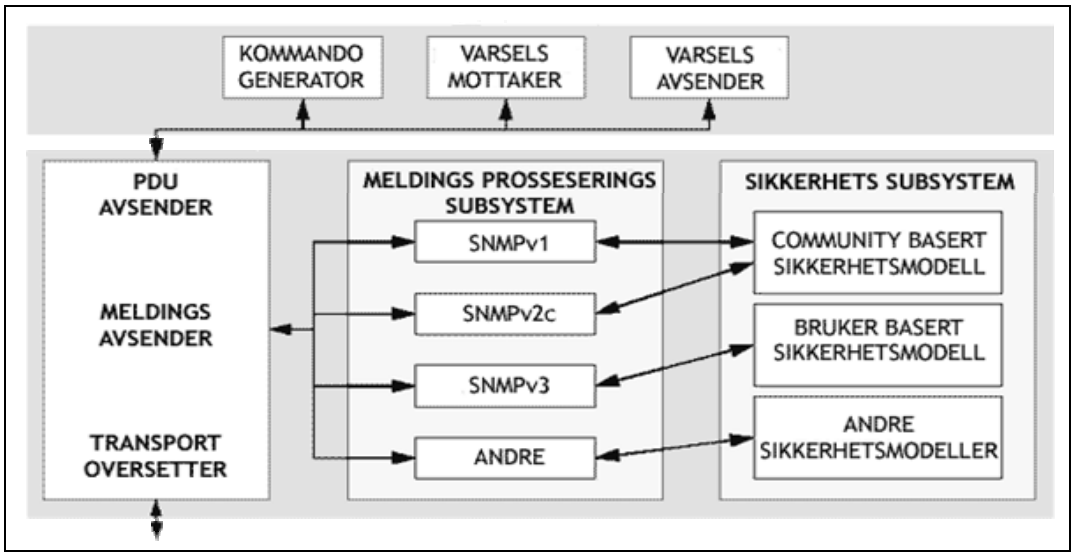
- To av dem brukes til å generere og respondere på kommandoer for overvåkning og kontrollering. (kommandogenerator og -responder)
- To andre brukes til å avsende og motta notifikasjoner om statusendringer i forvaltede objekter. (varselsavsender og -mottaker)
- Den femte brukes til å videresende meldinger på vegne av andre entiteter (proxy videresender)

Applikasjonene bruker tjenestene til motoren for å sende og motta meldinger med et gitt sikkerhetsnivå og for å utføre aksesskontroll for objektoperasjoner.

Ulike entiteter
Rollen til en entitet defineres av hvilke moduler som denne har implementert. Det er ikke gitt at en entitet må ha ett spesielt sett av moduler. Følgende stereotypene er nevnt i rammeverket:

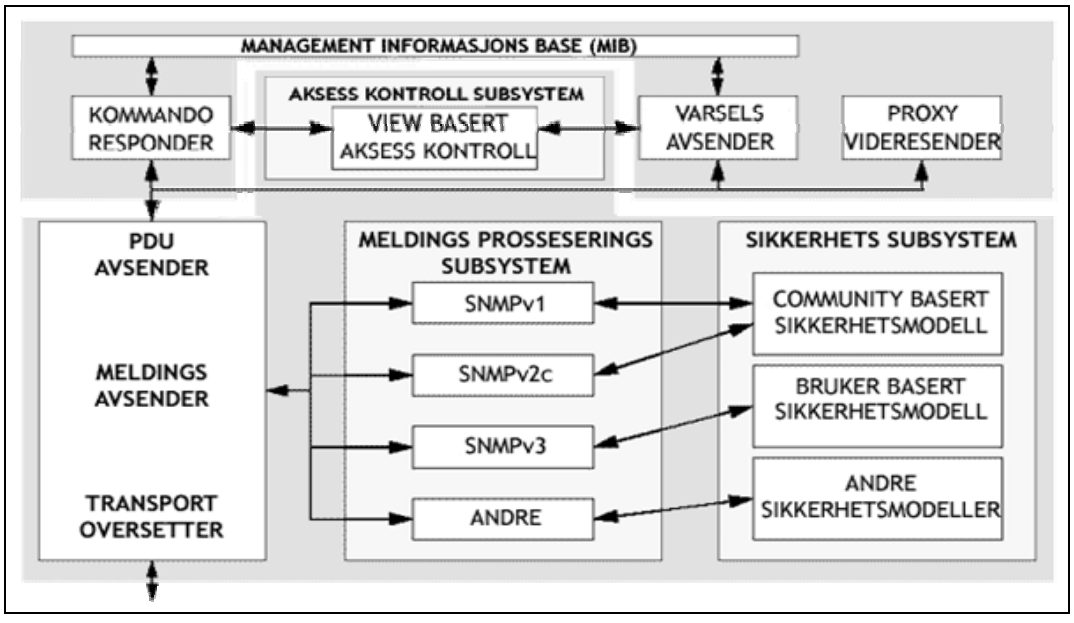
- SNMP agent – kommando responder og/eller varsels avsender applikasjon
- SNMP proxy agent – SNMP agent pluss proxy forwarder applikasjon.
- SNMP kommandoline manager – kommando generator og/eller varsels mottaker
- SNMP mellomnivå manager – SNMP kommandolinje manager pluss kommando responder og/eller varsels avsender
- Nettverks management stasjon – SNMP manager muligens med andre applikasjonstyper, for å forvalte et potensielt veldig stort antall av noder.

Eksempel på manager og proxy agent
Figur 6 og Figur 7 basert på figurer i RFC2271 [43], viser eksempler på en tradisjonell SNMP mellomnivå manager og SNMP proxy agent.



Figur 6: SNMP manager

Manageren (Figur 6) inneholder applikasjoner for å generere kommandoer til agenten, og for å sende og å motta varsler fra denne. Meldingene håndteres av en meldingsfordeler (dispatcher), som sender både innkommende og utgående meldinger til riktig subsystem for meldingsprosessering. Dette subsystemet sender også meldingen videre for prosessering i et sikkerhetssubsystem med en gitt sikkerhetsmodell. Vi har et meldingsprosesseringssubsystem for hver av SNMP versjonene pluss mulighet for å definere fremtidige, og hver av disse kan igjen bruke inntil flere subsystem for sikkerhet. Meldingsfordeleren tar seg også naturlig nok av meldingshåndteringen mot applikasjonene og transportlaget.

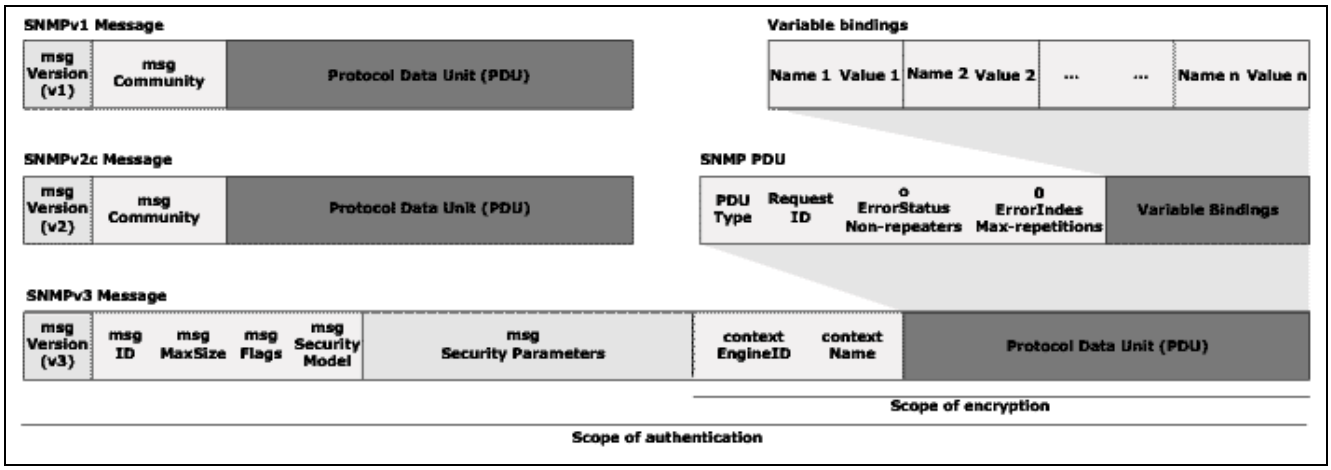


Figur 7: SNMP proxy agent

Agenten (Figur 7) inneholder applikasjoner for å respondere på kommandoer, generere varsler og å videresende meldinger på vegne av andre entiteter. Vi ser at agenten opererer mot en management informasjonsbase og at et subsystem styrer aksesskontrollen til denne. Også her går alle meldingene gjennom et subsystem for meldingsprosessering som igjen bruker et subsystem for sikkerhet.

2.3.4 Meldingsformater

Figur 8 illustrerer meldingsformatene for de ulike SNMP versjonene. Det første feltet i alle meldingene er versjonsnummeret, som sørger for at meldingene kommer til riktig meldingsprosesserings subsystem. Videre inneholder alle meldingene ett eller flere headerfelt, og til sist en Data Protocol Unit (PDU). En slik PDU inneholder en SNMP operasjonsmelding som består av flere headerdata og til sist innholdsdata (variabelbindinger). Disse igjen består av navn og verdier på sett av objekter.



Figur 8: SNMP meldingsformater

Vi ser at meldingsformatene til SNMPv1 og SNMPv2c er helt like. Disse inneholder et 'community' felt som fungerer som en identifikator som brukes som en veldig svak autentisering og gir mulighet for differensiert aksesskontroll. Meldingene har ellers ingen autentisering eller konfidensialitet av data.

SNMPv3 meldingsformatet er helt annerledes og bruker ikke noe community konsept. Meldingen består av globale data, sikkerhetsdata og meldingsdata.

De globale dataene notert *'msgGlobalData'*, er også kalt for header data. Disse feltene inneholder en unik meldingsidentifikator, beskjed om maksimal meldingsstørrelse den vil ha tilbake, meldingsflagg og sikkerhetsmodell for meldingen. Det er tre typer meldingsflagg: et rapportflagg som forteller om meldingen krever svar, ett konfidensialitetsflagg og ett autentiseringsflagg som indikerer hvilke av disse mekanismene som brukes. Sikkerhetsmodellfeltet sier hvilken sikkerhetsmodell som brukes og kommer data for i de påfølgende feltene.

De påfølgende sikkerhetsdataene notert *'msgSecurityParameters'*, er spesifikke for hvilken sikkerhetsmodell som brukes. Vi vil komme tilbake til disse i gjennomgangen av Bruker-basert Sikkerhetsmodell i kapittel 5 om SNMPv3 sikkerhet.

Til sist følger meldingsdataene notert *'msgData'*, som også på engelsk blir kalt scoped PDU. I tillegg til PDU-en inneholder denne også en kontekst motor identifikator og et kontekst navn, som brukes av subsystemet for aksesskontroll: View-based Access Control Module (VACM).

Vi ser av figuren at SNMPv3 meldingsdataene er gjenstand for en eventuell kryptering, mens autentisering eventuelt skjer for hele meldingen.

Det er litt forskjell på PDU formatene til SNMPv1 og SNMPv2c. SNMPv1 har et eget PDU format for 'trap' operasjonen, og ikke noen PDU for 'getBulk' operasjonen. Det er også viktig å forstå at SNMPv3 ikke definerer egne PDU formater, men bruker formatene til enten SNMPv1 eller SNMPv2c (default). Sånn sett er SNMPv3 bare et rammeverk for SNMPv1 eller SNMPv2 operasjoner over et nytt meldingsformat med funksjoner for trafikksikkerhet og aksesskontroll.

3 Sikkerhet

Dette kapittelet tar for seg nettverkssikkerhet, som en bakgrunn for senere vurdering av sikkerhetsløsninger for IP-basert managementtrafikk.

Rammeverk
Vi tar utgangspunkt i sikkerhetsarkitekturen for OSI baserte systemer definert i ISO/IEC 7498-2 [2]. Denne referansemodellen har også stor relevans for TCP/IP baserte systemer.

Referansemodellen er bygget opp med utgangspunkt i at det først defineres en sikkerhetspolicy. En sikkerhetspolicy kan f.eks. bestå av krav om konfidensialitet, integritet og tilgjengelighet. Vi identifiserer derfra trusler, tjenester og mekanismer i henhold til policyen.

Sammenhengen mellom trusler, tjenester og mekanismer er:
- En *sikkerhetstrussel* er et mulig middel ved hvilket en sikkerhets policy kan bli brutt
- En *sikkerhetstjeneste* er et tiltak som kan bli iverksatt for å håndtere en trussel
- En *sikkerhetsmekanisme* er et middel som tilveiebringer en eller flere tjenester

Vi vil videre se på hver av disse punktene med spesifikk tanke for management.

3.1 Sikkerhetstrusler

Nettverks management trafikk står ovenfor de samme generelle truslene mot *konfidensialitet, integritet og tilgjengelighet*, som all annen kommunikasjon av informasjon mellom en kilde og en destinasjon.

Disse truslene kan vi dele inn i fire hovedkategorier:
- *Avlytting* – En uautorisert part får tilgang til en ressurs (konfidensialitet).
- *Avbrytelse* – En ressurs i systemet er ødelagt eller blir utilgjengelig eller ubrukelig (tilgjengelighet).
- *Modifikasjon* – En uautorisert part ikke bare får tilgang til, men også tukler med en ressurs (integritet).
- *Fabrikasjon* – En uautorisert part setter inn falske objekter inn i systemet (integritet).

For en informasjonsflyt er det vanlig å skille mellom aktive og passive trusler:
- *Passive trusler* innebærer å finne ut eller utnytte informasjon fra systemet (avlytting) uten å påvirke disse. Dette gjør at passive trusler er svært vanskelig å oppdage.
- *Aktive trusler* involverer endring (avbrytelse, modifikasjon eller fabrikasjon) av systemressurser for å forandre på deres operasjon.

Spesifikke trusler
Flere kilder [1, 26, 30, 41, 43] har identifisert følgende *spesifikke trusler* for meldinger utvekslet mellom manager og agent (nettverksmanagement trafikk):
- *Avsløring av meldingsinnhold (avlytting)* – En entitet avlytter meldingsutvekslingen og er i stand til å tolke og trekke ut informasjon som sendes. Dette er særlig risikabelt dersom det medfører avsløring av autentiseringsinformasjon.
- *Trafikkanalyse (avlytting)* - En entitet avlytter meldingsutvekslingen og er i stand til å avlede informasjon fra karakteristikken til trafikken. Informasjon om lokalisasjon, identitet til kommuniserende parter, frekvens og lengde på meldingene, kan gjøre det mulig å gjetting av hva slags kommunikasjon som foregår, selv med konfidensialitetsbeskyttede meldinger.
- *Tjenestenektelse (avbrytelse)* – En entitet hindrer eller umuliggjør normal bruk av kommunikasjonsutstyr. Dette kan gå mot enkeltenheter (manager eller agent) eller hele nettverk, f.eks. ved å frakoble eller overbelaste dem.
- *Modifikasjon av informasjon (modifikasjon)* – En entitet forandrer en melding som er underveis fra en autorisert entitet, for å forårsake uautoriserte management operasjoner, inklusive endring av objektverdier.

- *Modifikasjon av meldingsstrømmen (modifikasjon)* – En del av en legitim meldingsstrøm forsinkes, gjentas eller omstokkes, for å skape en uautorisert effekt. Dette er særlig relevant for SNMP som default går over den forbindelsesløse transportprotokollen UDP.
- *Maskerade (fabrikasjon)* – En entitet får tilgang til og utfører uautoriserte management operasjoner, ved å late som den er en autorisert entitet. Dette skjer som regel i sammenheng med andre angrepstyper. For eksempel ved å avlytte meldingsutvekslingen, identifisere en autentiseringssekvens og gjentakelse av denne for å påstå en autorisert identitet

Av disse er kanskje modifikasjon av informasjon og maskerade de to viktigste truslene, mens avsløring av meldingsinnhold og modifikasjon av meldingsstrøm er sekundære trusler. Tjenestenektelse er vanskelig å skille ut fra vanlige nettverksfeil og omfatter sannsynligvis all kommunikasjon og kan derfor håndteres av generelle sikkerhetsmekanismer, istedenfor mekanismer spesifikke for management sikkerhet. Når det gjelder trafikkanalyse er mye av trafikkmønsteret forutsigbart, og det hat derfor liten nytteverdi å beskytte mot dette.

3.2 Sikkerhetstjenester

Sikkerhetsarkitekturen [2] definerer følgende tjenester for å motvirke truslene:

- *Autentisering* - forsikre seg om at brukere er de entiteter de påstår å være. Det er mulig å autentisere hvor data kommer fra, og hvem den du kommuniserer med.
- *Aksesskontroll* - forsikre seg om at brukere bare aksesserer de ressurser og tjenester som de har rett til, og at kvalifiserte brukere ikke blir nektet tilgang til tjenester de lovlig forventer å motta.
- *Datakonfidensialitet* - forsikre seg om at data er utilgjengelig for andre enn autoriserte brukere.
- *Dataintegritet* - forsikre seg om at data ikke har blitt modifisert av uautoriserte brukere.
- *Tilgjengelighet* - forsikre seg om at et system er operasjonelt og funksjonelt ved et gitt tidspunkt.
- *Ikke-benektelse* - forsikre seg om at f.eks. en avsender av en melding ikke kan nekte for at de faktisk sendte meldingen.

Av disse er tjenestene autentisering, dataintegritet og datakonfidensialitet de mest relevante med tanke på å motvirke trusler mot managementtrafikk.

3.3 Sikkerhetsmekanismer

Sikkerhetsarkitekturen [2] definerer følgende spesifikke mekanismer som implementerer av en eller flere av tjenestene:

Kryptering
Kryptering kan tilby konfidensialitet for data eller trafikkflytsinformasjon, og kan være en del av eller utfylle en hel del andre sikkerhetsmekanismer.

Krypteringsalgoritmer kan være reversible eller irreversible. Vi har to generelle klasser reversible krypterings algoritmer:

- *Symmetriske* (dvs hemmelig nøkkel) krypteringsalgoritmer, bruker samme nøkkel til både kryptering og dekryptering.
- *Asymmetriske* (f.eks. offentlig nøkkel) krypteringsalgoritmer, bruker forskjellige nøkler for kryptering og dekryptering. En privat nøkkel brukes til kryptering og en offentlig som ikke kan utlede den private til dekryptering.
- *Irreversible* krypteringsalgoritmer kan, men må ikke bruke nøkler. Ved bruk av nøkkel kan denne være offentlig eller hemmelig.

Bruk av krypteringsalgoritmer forutsetter bruk av *mekanismer for nøkkelhåndtering*, unntatt for noen irreversible krypteringsalgoritmer.

Digitale Signaturer
Mekanismer for digital signatur gir tjenester som ikke-benektelse for avsender og autentisering av avsender eller dataintegritet.

Digitale signaturer benytter seg av asymmetriske krypteringsalgoritmer, og består av to prosedyrer: signering av en dataenhet og verifisering av en signert dataenhet.

- *Signeringsprosessen* bruker informasjon som er privat (dvs unik og konfidensiell) for den som signerer, og involverer enten kryptering av en dataenhet eller produksjon av en kryptografisk sjekksum over dataenheten.
- *Verifikasjonsprosessen* bruker offentlige prosedyrer og informasjon hvorfra den private informasjonen ikke kan utledes, for å bestemme om signaturen var produsert med avsenderens private informasjon.

Siden en signatur ikke kan ha vært laget av andre enn innehaveren av den private informasjonen, kan en verifisert signatur senere også brukes som bevis ovenfor en tredjepart, at innehaveren av den private nøkkelen virkelig har sent informasjonen (ikke-benektelse).

Aksesskontroll
Mekanismer for aksesskontroll kan bruke den autentisert identiteten til en entitet, informasjon om entiteten eller egenskaper til entiteten, for å bestemme og håndheve aksess rettighetene til denne.

Hvis en entitet prøver å aksessere ressurser den ikke har aksess til eller feil aksessrettigheter for, vil en aksesskontrollsfunksjon avvise forsøket. I tillegg kan funksjonen også eventuelt informere definerte entiteter eller brukere om det uautoriserte aksessforsøket.

Aksesskontrollmekanismer kan for eksempel være basert å en eller flere av de følgende:

- *Aksesskontrolls informasjonsbaser*, hvor aksessrettigehetene til likestilte entiteter vedlikeholdes. Denne informasjonen kan vedlikeholdes av autorisasjons sentre eller av entiteten som blir aksessert, og forutsetter autentisering av kommunikasjonsmotparten.
- *Autentiseringsinformasjon* som passord, besittelser eller følgende presentasjon av besittelser som beviser den aksesserende entitetens autorisasjon.
- *Egenskaper*, besittelser eller følgende presentasjon av besittelser som beviser retten til å aksessere entiteten eller ressursen definert av egenskapen.
- *Sikkerhetslabler*, som når assosiert med en entitet kan brukes for å bevilge eller nekte aksess, normalt i henhold til en sikkerhetspolicy.
- *Tiden* for aksessforsøket
- *Ruten* for aksessforsøket
- *Varigheten* av en aksess

Mekanismer for aksesskontroll kan anvendes i begge ender av en kommunikasjonsassosiasjon eller på et mellomliggende punkt.

Dataintegritetsmekanismer
Dataintegritet går ut på å beskytte data mot modifikasjon. Vi har to typer dataintegritets mekanismer. De som brukes for å beskytte integriteten til en enkelt dataenhet eller datafelt, og de som i tillegg beskytter sekvensen til hele strømmen av dataenheter eller –felt i en forbindelse.

Dataintegritet for en enkelt dataenhet involverer to prosesser, en hos sender og en hos mottaker. Senderen tillegger en unik mengde til dataene som selv er en funksjon av dataene. Denne mengden kan være tilleggsinformasjon som en sjekkblokk kode eller en kryptografisk sjekkverdi og kan selv være kryptert. Mottakeren genererer en tilsvarende mengde fra dataene og sammenlikner det med den mottatte for å bestemme om dataene har blitt forandret under transporten. Mekanismen vil i seg selv ikke beskytte mot avspilling av en dataenhet. Men deteksjon av manipulasjonen kan føre til gjenoppretting via retransmisjon eller feiloppretting.

For forbindelsesorientert dataoverføring, kreves i tillegg en eller annen form for eksplisitt rekkefølge som sekvensnummerering, tidsstempel eller kryptografisk kjeding, for å beskytte

integriteten til en sekvens med dataenheter (f.eks. beskyttelse mot feil i rekkefølge, tap, avspilling og innsetting eller modifikasjon av data).

For forbindelsesløs dataoverføring kan tidsstempel brukes for å oppnå en begrenset form for beskyttelse mot avspilling av individuelle dataenheter.

Autentiseringsutveksling
Mekanismer for autentiseringsutvekslinger besørger autentisering av kommunikasjonsmotpart.

Noen av teknikkene som kan brukes til autentiseringsutvekslinger er:
- Bruk av autentiserings informasjon slik som passord levert fra senderen og kontrollert av mottakeren.
- Kryptografiske teknikker
- Bruk av karakteristikker eller besittelser til entiteten

Kryptografiske teknikker kan kombineres med håndtrykk for å også gi avspillingsbeskyttelse.

Valg av teknikk for autentiseringsutveksling avhenger av omstendighetene de skal brukes under. Noe av det som da er viktig å ta stilling til er bruk av:
- Tidsstempel og synkroniserte klokker
- To og treveis håndtykk henholdsvis for ensidig og gjensidig autentisering
- Tjenester for ikke-benektelses ved hjelp av digital signatur eller notarisasjonsmekanismer

Trafikk utfylling
Mekanismer for trafikkutfylling besørger konfidensialiteten til trafikkflyten, som gir en varierende grad av beskyttelse mot trafikkanalyse. Mekanismene er bare effektive når den i tillegg beskyttes av en konfidensialitetstjeneste.

Ruting kontroll
Mekanismer for ruting kontroll benyttes for å besørge konfidensialiteten til en forbindelse (både forbindelsesorientert og forbindelsesløs) og trafikkflyten i forbindelsen. Dette gjør den ved å kontrollere (dynamisk eller forhåndskonfigurert) trafikkruten slik at det kun benyttes fysisk sikre ruter.

Endesystemer kan ved deteksjon av vedvarende manipulsasjonsangrep, be nettverksleverandøren om etablere en forbindelsen over en annen rute

En sikkerhetspolicy kan forby data med visse sikkerhetsmerker å passere over bestemte ruter. Den som oppretter en forbindelse (forbindelsesløs) kan også spesifisere advarsler som sier at spesifikke ruter ikke bør brukes.

Notarisasjon
En mekanisme for notarisasjon kan garantere for egenskaper ved data kommunisert mellom to entiteter, slik som dataenes integritet, opphav, tid og destinasjon. Mekanismene kan altså besørge tjenesten ikke-benektelse for både avsender og mottaker.

En tiltrodd tredjepart som innehar den nødvendige informasjonen for å angi den forespurte bevitnelsen på en sikker måte, står som garantist

Hver kommunikasjonsinstans kan bruke digital signatur, kryptering, og integritetsmekanismer alt etter tjenesten levert av den tiltrodde tredjeparten. Når en notarisasjonsmekanisme er igangsatt, blir dataene kommunisert mellom entitetene via tredjeparten og dens beskyttelses mekanismer.

4 Oppgavetilnærmingen

Målet med oppgaven er å evaluer løsningene 'SNMPv3 Security', 'Internett Protocol Security (IPSec)' og 'Transport Level Security (TLS)', for å sikre konfidensialiteten, integriteten og autentiseringen til Simple Network Management Protokcol (SNMP) trafikk. Videre var det også et tilleggsmål å undersøke støtten for remote innlogging via 'Secure Shell (SSh)' i sentrale nettverksenheter.

Det finnes lite offentlig tilgjengelig av relevant eksisterende arbeide som vi kan bygge på, utenom det tidligere nevnte paperet om implementasjon og ytelsesanalyse av SNMP over 'Transport Layer Security' og 'Transport Control Protocol' [6]. I tillegg til dette vil vi bygge på forskjellige beskrivelser av løsningene som ikke nødvendigvis er spesifikke for transportering av management trafikk. Da det ser ut til å ikke finnes mye liknende arbeider, vil undersøkelsene som denne oppgaven søker å utføre vil bringe forskningen på dette området ett hakk videre.

Vi vil i dette kapittelet beskrive hvordan vi vil gå frem for å løse problemstillingen. Først vil vi skissere et testsenario, dernest utdype hva vi legger i de forskjellige evalueringsparameterene.

4.1 Testscenario

Hensikten med et testscenario er å få erfaring med hvordan det er å konfigurere de forskjellige løsningene med forskjellige sikkerhetsnivå, samt å kjøre sammenliknbare ytelsestester for disse.

For at testene skulle bli mest mulig sammenliknbare, valgte vi ett svært enkelt oppsett med en manager og en agent med en krysset trådparkabel i mellom. På denne måten unngikk vi eksterne parametere som ville påvirket testene. I tillegg søkte vi at de to kommunikasjonspartene skulle være mest mulig statiske, slik at vi fikk et konformt testmiljø. I tillegg var dette metoden som var brukt i tidligere tester med Transport Layer Security [6].

Vi valgte å kjøre manageren på en laptop. Agenten kjørte vi på en Cisco ruter, som er noe av det mest vanlige[2] kjernenettverksutstyret. Dette skulle etterhvert vise seg å medføre en del utfordringer, men vi valgte likevel å fortsette oppgaven med denne løsningen.

Det problematiske bestod i:

1. Leveransen av ruteren to ekstremt lang tid
2. Leverandøren gav oss først software som ikke støttet det vi hadde spesifisert.
3. I et møte med Cisco Systems Norge, fikk vi vite at:
    a. Deres implementasjon av Simple Network Management Protocol (SNMP), ikke lenger støtter Transport Control Protocol (TCP), som må brukes for å kunne kjøre Transport Layer Security (TLS).
    b. Det ikke finnes generell støtte for Transport Layer Security i nettverksenhetene da protokollstakken utenom til enhetens management applikasjoner, kun går opp til nettverkslaget (ruterfunksjonalitet).

Dette førte til at vi utelot å konfigurere og teste Transport Layer Security (TLS) og istedenfor baserte oss på resultatene fra tidligere rapporter [6].

Vi kunne gått over til å gjøre testene med en vanlig vertsmaskin som agent slik som ble gjort i [6], men valgte å fortsette med kjernenettverksutstyr da vi anså dette for mest relevant.

Vi hadde jo i hvert fall avdekket at løsningen med TLS ikke var gjennomførbar for kjerneutstyr.

[2] I følge markedsrapporter fra Garthner Brooks har Cisco Systems 91,2% markedsandel for low-end rutere i tredje kvartal 2002.

4.1.1 Manager

Manageren var som nevnt en laptop, nærmere bestemt en IBM Thinkpad med Pentium 3 prosessor, 256 Mb RAM og et 10/100 Mb nettverkskort. Til operativsystem brukte vi RedHat Linux 8.0 med en 2.4.18-14 kjerne (default).

Til Simple Network Management Protocol (SNMP) brukte vi programpakken net-snmp 5.0.1-6, som kommer default med RedHat 8.0 distribusjonen. Denne er en videreføring av ucd-snmp prosjektet, hvis pakke ble brukt i [6]. Programvaren og dokumentasjonen for begge disse løsningene kan lastes ned på net-snmp prosjektets hjemmeside [48].

For Internet Protocol Security (IPSec) brukte vi den populære linux implementasjonen FreeS/WAN 1.99. Programvaren og dokumentasjon er tilgjengelig på freeswan prosjektets hjemmeside [49]. Nærmere bestemt så brukte vi en patchet implementasjon kalt Super-FreeS/WAN [50] som støtter 1des og nullkryptering, i en ferdigpakket versjon (både kompilert kernel og userspace) [51].

Siden agenten som nevnt ikke støtter SSL eller TLS, så vi heller ikke på dette for manageren. Men RedHat 8.0 distribusjonen kommer uansett som standard med OpenSSL 0.9.6b installert. Dokumentasjon og programvare for denne finnes på OpenSSL sin hjemmeside [52].

For Secure Shell (SSH), brukte vi OpenSSH pakken versjon 3.4p1 som kommer med Redhat 8.0 distribusjonen. Denne støtter både SSH versjon 1.5 og SSH versjon 2.0. Dokumentasjon og programvare for denne finnes på openssh prosjektets hjemmeside [53].

4.1.2 Agent

Agenten var som nevnt en Cisco ruter, nærmere bestemt en Cisco 2621 ruter av 'low-end' kategori med Cisco IOS Software 12.2(11) T2. Softwaresettet er av type IP-pluss/IPSec/3DES, som kreves for å støtte SNMPv3 med kryptering, IPSec, og SSH versjon 1. På Cisco Systems sine hjemmesider [54] finnes dokumentasjonen for ruteren [55] og programvaren [56], og i tillegg mer generell dokumentasjon om SNMP [57, 58], IPSec [59, 60, 61] og SSH [62].

I programvaren er det som nevnt ikke støtte for SNMP over TCP. Dette har tidligere kun eksistert i 'IOS Software 12.0' versjonen, men ble tatt bort i neste versjon. Det er heller ikke lagt inn støtte for SSL eller TLS på ruteren. Det er lite sannsynlig at andre dedikerte rutere har støtte for dette, derfor anser vi en slik løsning for mindre relevant.

4.2 Evalueringsparametere

I dette kapittelet vil vi utdype de tre parametrene sikkerhetsnivå, konfigurasjon og ytelse, som brukes for å evaluere de foreslåtte løsningene.

De påfølgende hovedkapitlene i rapporten vil inneholde et underkapittel for hver av disse parametrene, som til slutt blir sammenliknet og diskutert i kapittel 9.

4.2.1 Sikkerhetsnivå

Vi ønsker å bruke følgende parametere for evaluering av sikkerhetsnivå for de forskjellige løsningene:

Protokollag
De forskjellige protokollagene behandler data til og fra det overliggende laget. Hvor i laghierarkiet protokollen befinner seg, får derfor betydning for hvilke deler av den overførte meldingen (inkluderer alle involverte protokoller) som sikkerhetsmekanismene opererer på.

Nøkkelhåndtering
Metoder for distribusjon og oppdatering av nøkler danner grunnlag for de fleste kryptografiske sikkerhetsmekanismer. Nøkler kan forhåndsutveksles manuelt eller forhandles etter gjensidig autentisering. Den mest vanlige metoden å utveksle og oppdatere nøkler på er ved hjelp av asymmetriske krypteringsalgoritmer.

Dataautentisering
Begrepet data autentisering innebefatter både dataintegritet og autentisering av dataopprinnelse. Til denne kombinasjonen av tjenester er det mest vanlig å bruke en kryptografisk mekanisme kalt Message Authentication Code (MAC).

Avspillingsbeskyttelse
Beskyttelse mot avspilte meldinger er også en viktig parameter for sikkerhetsnivået. Dette er en trussel mot integriteten til en forbindelse, og er vanlig å beskytte med en eller annen form for teller.

Datakonfidensialitet
De aktuelle løsningene for konfidensialitet baserer seg hovedsakelig på symmetrisk kryptografi. De forskjellige algoritmene er forskjellig oppbygd, og har dermed forskjellige egenskaper og karakteristikk. Mest sentralt er den kryptografiske styrken på algoritmen. Den viktigste parameteren som gir en indikasjon på denne styrken er nøkkellengden.

Aksesskontroll
Aksesskontroll er en viktig del av totalsikkerheten for ett system. Denne hever seg over selve trafikksikkerheten og foregår i så måte på applikasjonslaget.

Kombinasjoner
Hvilke kombinasjoner av sikkerhetsnivå som er tilgjengelig er også vesentlig. Vi er primært interessert i løsningene hvor vi har bare autentisering, og både konfidensialitet og autentisering.

Totalvurdering
Den totale sikkerheten til en løsning vil aldri være sterkere enn det svakeste leddet. Derfor er det viktig at alle mekanismene gir omtrent like god beskyttelse.

4.2.2 Konfigurasjon

Vi ønsker å bruke følgende parametere for å evaluere hvor enkelt det er å konfigurere de forskjellige løsningene:

Brukergrensesnitt
Enhetene i testscenariet har forskjellig brukergrensesnitt. Et brukergrensesnitt kan delvis bestemme hvor enkelt det er å konfigurere en løsning for operatøren. Grensesnittene kan enten være grafiske, kommandolinjebaserte eller tekstbaserte innen en eller flere konfigurasjonsfiler.

Installasjon
Installasjonen er en viktig del av forarbeidet til konfigurasjonen. Dette kan skje på ulike måter for de forskjellige sikkerhetsløsningene. Parametere som kan gi en indikasjon på kompleksiteten av installasjonen kan for eksempel være tilgang til komplett programvare, og tiden for hele installasjonen.

Konfigurasjonsparametere
Antall parametere som må konfigureres og deres gjensidige avhengighet gir også en indikasjon på kompleksiteten til konfigurasjonen. Det er ikke alltid gitt at det er enkelt å sette en hel del med parametere eksempelvis i forskjellige konfigurasjonsfiler og med ulike kommandoer.

Dokumentasjon
Erfaring tilsier at det ofte trengs det dokumentasjon i en eller annen form for å forstå en større sammenheng eller enkeltbrikker, som er nødvendig for å få til en konfigurasjon. Derfor er det vesentlig hvor lett det er å finne passende dokumentasjon. Kilder til dette kan være kommandolinjehjelp, kommentarer i konfigurasjonsfiler, online manualer, online lister med ofte spurte spørsmål, e-postlister, andre former for brukersamfunn, og supportavtaler.

Skalerbarhet
For større organisasjoner er det også vesentlig hvor skalerbar løsningen er. Dette kan forebygge høye fremtidige investerings- og vedlikeholdskostnader.

Vedlikehold
Vedlikehold av systemene er som oftest kritisk for en administrator. Aktuelle problemstillinger er blant annet oppdatering av nøkler og programvare. Muligheter for å kjøre skript i løsningene er også interessant da dette kan lette manuelle vedlikeholdsoppgaver.

4.2.3 Ytelse

For å måle ytelsen til de forskjellige løsningene med forskjellig sikkerhetsnivå, måler vi responstiden for management operasjonen 'get' over hele management informasjonsbasen et gitt antall ganger.

Management informasjonsbasen på Cisco ruteren inneholder i overkant av 700 objektvariable når alle løsningene er konfigurert ferdig.

For denne oppgaven har vi laget et skript som genererer en liste med get operasjoner mot objekter, ved hjelp av en suksessiv gjennomgang av informasjonsbasen med getnext operasjonen. Manageren kjører denne listen med operasjoner mot agenten, og venter til applikasjonen har fått svar før den utføre neste operasjon

For å få en god statistisk signifikans har vi valgt å gjøre 30 repetisjoner for hver løsning med et gitt sikkerhetsnivå. Gitt at responstiden er normalfordelt gir dette antallet repetisjoner god nok signifikans for resultatene. Dette fremgår av teoremet om konfidensintervall for gjennomsnittet av et tilfeldig utsnitt av en normalfordeling med ukjent standardavvik [63] (teorem 11-3 og tabell IV)

De forskjellige sikkerhetsnivåene vi ønsker å måle er:

- Ingen sikkerhetsbeskyttelse
- Dataautentisering (HMAC-MD5)
- Konfidensialitet (1DES-CBC) pluss dataautentisering.

Ved å måle med samme autentiserings og konfidensialitetsalgoritmer, blir testresultatene mest mulig sammenliknbare.

5 SNMPv3 Security

Dette kapittelet tar for seg sikkerhetsløsningene i SNMPv3. Det begynner med en presentasjon av løsningen, og tar så for seg hver av de tre evalueringsparametrene.

5.1 Protokoll

Sikkerhetsløsningene til SNMPv3 finner sted på applikasjonslaget. Disse besørger sikkerhet fra applikasjon til applikasjon, uavhengig av andre applikasjoner og underliggende lag.

Arbeidsgruppen for SNMPv3 definerer som nevnt SNMP Arkitekturen [43], et rammeverk med subsystemer for sikkerhet og for aksesskontroll. For SNMPv3 finnes det for tiden kun en implementasjon av hver av disse. Dette er henholdsvis Bruker-basert Sikkerhetsmodul (USM) [46] og View-basert Aksesskontrollmodul (VACM) [47].

Den bruker-baserte sikkerhetsmodulen (USM) besørger autentisering og konfidensialitet på nivået for prosessering av meldinger Den definerer også prosedyrer for nøkkelhåndtering og en tidslinjefunksjon for å beskytte mot avspilling av meldinger.

Den view-baserte aksesskontrolsmodulen (VACM) opererer på nivået for prosessering av 'Protokoll Data Enheter' (PDU) og bestemmer hvorvidt en gitt bruker har adgang til gitte MIB objekter for å utføre gitte funksjoner (read, write, notify).

Vi vil i dette kapittelet konsentrere oss primært om bruker-basert sikkerhetsmodul (USM) for SNMPv3, introdusert i SNMPv2u og SNMPv2*. I tillegg til denne har vi community-basert sikkerhetsmodul for SNMPv1 og SNMPv2c, og party-basert sikkerhetsmodul for SecureSNMP og SNMPv2. Denne siste modulen er ikke lengre en offisiell standard.

Siden subsystemene er organisert i moduler, er det mulig å ha flere sikkerhetsmoduler med ulike sikkerhetsparametere for en SNMP versjon. Det er derfor godt mulig at vi i fremtiden vil se flere sikkerhetsmoduler for SNMPv3 eller nyere versjoner.

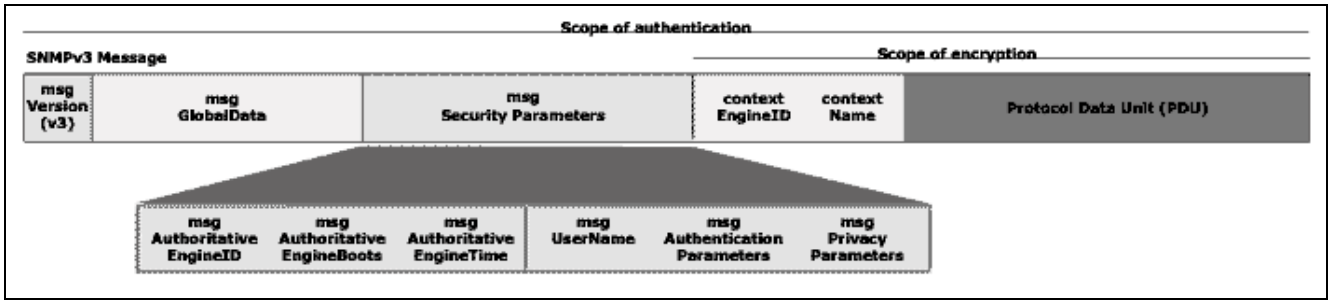
5.1.1 User-based Security Model (USM)

Spesifikasjonen av bruker-basert sikkerhetsmodul (USM) omfatter [4]:

- Meldingsformatet: Definerer formatet til msgSecurityParameters, som støtter funksjonene for tidslinje, autentisering og konfidensialitet.
- Tidslinje: Beskytter mot forsinkelse og avspilling av meldinger
- Autentisering: Besørger dataintegritet og autentisering av dataopprinnelse. Dette skjer ved hjelp av 'Meldingsautentiseringskode' (MAC) med hash funksjonene MD5 eller SHA-1 for autentisering.
- Konfidensialitet: Beskytter mot avsløring av meldingsinnhold. Dette skjer ved hjelp av kryptering i Cipher Block Chaining (CBC) modus av 'Data Encryption Standard' (DES).
- Nøkkelhåndtering: Definerer prosedyrer for generering, oppdatering og bruk av nøkler.

Meldingsformatet

Figur 9 illustrerer meldingsformatet for SNMPv3 med parametrene for bruker-basert sikkerhetsmodul (USM) fremhevet. De tre første parametrene (Authoritative Engine ID, Boots og Time) er relatert til tidslinjefunksjonen, og de tre siste (UserName, AuthenticationParametres og PrivacyParameters) er brukerspesifikke.



Figur 9: SNMPv3 meldingsformat med USM

Tidslinjefunksjonen

Tidslinjefunksjonen beskytter mot forsinkelse og avspilling av meldinger. Denne spesifiserer en såkalt 'autoritativ motor'. Dette er den motoren av kommuniserende manager og agent som svarer på forespørsler (mottar get*, set og inform) og sender meldinger som ikke krever svar (trap og response). Dette identifiseres ved hjelp av rapport flagget (msgFlags i msgGlobalData).

Hver gang en *autoritativ motor* sender en melding til en ikke-autoritativ motor, sender den også med sin egen motor identifikator (EngineID), antall oppstarter (EngineBoots) og tiden som har gått siden siste oppstart (EngineTime). Disse parametrene vedlikeholdes lokalt hos en ikke-autoritativ motor som oppdaterer tiden (EngineTime).

Hver gang en *ikke-autoritativ motor* sender en forespørsel, sender den også med den kalkulerte verdi av den autoritative motoren sin tid siden oppstart (EngineTime). Meldingen blir akseptert hvis tidsverdien ligger innefor et visst tidsvindu. Dette fører til at meldinger som blir forsinket eller avspilt vil være utenfor tidsvinduet til den autoritative motoren. Disse meldingene forkastes.

Den ikke-autoritative motoren vedlikeholder også den høyeste mottatte motortiden fra den autoritative motoren. Dette forebygger avspillingsangrep som søker å forhindre den ikke-autoritative motoren sin notasjon av tiden til den autoritative motoren fra å avanserer.

Brukerspesifikke parametere

De tre siste sikkerhetsparametrene er som nevnt brukerspesifikke. Brukernavnet identifiserer en unik bruker med påfølgende spesifiserte parametrene for autentisering og konfidensialitet.

Separate verdier på nøkler for konfidensialitet og autentisering vedlikeholdes for følgende brukere:

- Lokale brukere: Enhver i SNMP motoren hvor management operasjoner er ønsket.
- Remote brukere: Enhver i remote SNMP motor hvor kommunikasjon er ønsket.

Disse verdiene lagres for hver bruker og er ikke mulig å aksessere via SNMP.

Autentisering

USM tilbyr en av to alternative protokoller for autentisering: HMAC-MD5-96 and HMAC-SHA-96. HMAC bruker en sikker hash funksjon og en hemmelig nøkkel for å lage en meldings autentiserings kode (MAC).

HMAC er veldig vanlig i internett-baserte applikasjoner og definert i RFC 2104 [64] For HMAC-MD5-96, blir HMAC brukt med MD5 [65] som den underliggende hash funksjonen. En 16 oktetts (128-bit) autentiseringsnøkkel brukes som innput til HMAC algoritmen. Algoritmen produserer ett 128-bits resultat, som blir avkortet til 12 oktetter (96-bit). For HMAC-SHA-96, brukes SHA-1 [66] som den underliggende hash funksjonen. Denne har en 20 oktetters autentiseringsnøkkel og produserer et 20-oktetters resultat som avkortes til 12 oktetter.

Konfidensialitet
USM bruker cipher block chaining (CBC) modus av Data Encryption Standard (DES) til kryptering.

En 16 oktetters (128 bit) konfidensialitetsnøkkel (privKey) fungerer som input til krypterings algoritmen. De første 8 oktettene (64-bit) av nøkkelen brukes som DES nøkkel. Siden DES kun trenger en 56-bits nøkkel, ignoreres det minst signifikante bittene i hver oktett.

En 64-bits initialiseringsvektor (IV) trengs for CBC modus. Denne lages på følgende måte:

- De siste 8 oktettene av en 16-oktetters privKey brukes som pre-IV.
- For å forsikre at to forskjellige IV-er for to like klartekster kryptert med same nøkkel, brukes en saltverdi. Saltverdien tilsvarer den nåværende verdien av smpEngineBoots (4 oktetter) og ett lokalt 4 oktetters heltall som vedlikeholdes av den lokale krypteringsprotokollen. Denne integeren er implemenatsjonsavhengig og bør forandres etter hver gang den brukes.
- Saltverdien XOR-es bitvis med preIV for å produsere IV.

Saltverdien settes inn i feltet med msgPrivacyParameters i SNMPv3 meldigen for å gjøre det mulig for mottakeren å beregne rett IV.

Denne metoden oppnår følgende:

1. Siden saltverdien forandres for hver bruker brukes forskjellige IV-er for forskjellige klartekster.
2. Bare saltverdien overføres over SNMP, hvilket forhindrer an angriper i å bestemme IV-en.

Nøkkelhåndtering
Management av konfidensialitets- og autentiseringsnøkler er vitalt for sikkerheten til SNMP. Vi vil her ta en kort titt på hvordan nøklene genereres, lokaliseres og oppdateres.

Generering av nøkler fra passord
Nøklene til SNMP USM beregnes ut i fra menneskelig lesbare passord. Det er ingen krav til passordet, men det bør ikke være enkelt å gjette.

Generering av nøkler fra passord skjer på følgende måte.

1. Brukerpassordet brukes som input for å lage en 2^20 oktetter lang streng ved å repetere passordet så mange ganger som nødvendig (overflødige bit kuttes) for å lage strengen digest0.
2. En hash av denne (MD5 eller SHA-1) kjøres for å lage digest1, som er brukerens nøkkel.

Denne teknikken har følgende fordeler:

- Den sørger for en kraftig bremsing av brute-force og ordliste angrep, ved å øke tiden det tar for å beregne den ovenfor skisserte to trinns løsningen.
- Den frakobler brukernøklene fra nettverksmanagementsystemene. Ingen stasjon trenger å oppbevare verdier for brukernøkler. Istedenfor genereres nøklene fra brukerpassordet ved behov.

Blumenthal [67] lister en vurdering av motivene for tilnærmingen med NMS uavhengig passord.

1. Vedlikehold av sentrale oppbevaringssteder for hemmelige nøkler er et sterkt alternativ til å generere nøkler fra passord. Denne metoden påvirker generell pålitelighet og kan gjøre feilsøking umulig hvis oppbevaringsstedet ikke er tilgjengelig når det trengs.
2. Duplikate oppbevaringssteder forverrer den overordnede sikkerheten ved å tilby flere angrepsmål for potensielle angripere.
3. Oppbevaringsstedene (både sentraliserte og duplikate) må vedlikeholdes på en sikker lokasjon. Dette reduserer muligheten for 'forward camp establishment' ved store gjenopprettingsforsøk (firefighting).

Lokalisering av nøkler

En lokalisert nøkkel er definert som en hemmelig nøkkel delt mellom en bruker og en autoritativ SNMP motor [46]. Hensikten er at brukeren kun trenger å vedlikeholde en enkelt nøkkel (eller to nøkler hvis både autentisering og konfidensialitet er påkrevd) og derfor bare trenger å huske ett passord (eller to). De faktiske hemmeligheter delt mellom en enkelt enhet og hver autoritative SNMP motor, er forskjellig. Prosessen hvor en enkelt brukernøkkel blir konvertert til flere unike nøkler, en for hver fjerntliggende SNMP motor, refereres til som nøkkellokalisasjon. Referanse [67] forklarer motivasjonen for en slik strategi. Denne oppsummerer vi her i følgende mål for management av nøkler:

- Hvert SNMP agent system i et distribuert nettverk har en egen unik nøkkel for hver bruker som er autorisert til å operere på denne. Hvis flere brukere er autoriserte som managere, så har agenten en unik autentiseringsnøkkel og en unik krypteringsnøkkel for hver bruker. Hvis en nøkkel for en bruker blir kompromittert, vil derfor nøklene for de andre brukerne fortsatt være brukelige.
- Nøklene til en bruker på forskjellige agenter er ulike. Hvis en agent blir kompromittert, vil bare brukernøklene for den agenten være kompromittert, og ikke brukernøklene i bruk for andre agenter.
- Nettverks managemet kan utføres fra et hvilket som helst punkt i nettverket, uten hensyn til tilgjengeligheten av et forhåndskonfigurert nettverks management system (NMS). Dette tillater en bruker å utføre management funksjoner fra hvilken som helst management stasjon. Dette tilveiebringes av 'passord til nøkkel' algoritmen som er beskrevet tidligere.

Vi kan også definere følgende som ting å unngå:

1. En bruker må huske (alternativt forvalte) et stort antall nøkler, som øker med tillegging av nye agenter.
2. En fiende som får kjennskap til nøkkelen til en agent, blir i stand til å utgi seg for å være hvilken som helst annen agent for hvilken som helst bruker, eller hvilken som helst bruker for hvilken som helst annen agent.

For å håndtere de foregående målene og betraktningene, blir en enkelt brukernøkkel avbildet ved hjelp av en ikke-reversible enveis funksjon (f.eks. en sikker hash funksjon) til forskjellige lokaliserte nøkler for forskjellige autentiserte motorer (forskjellige agenter). Prosedyre er som følger:

1. Skap strengen digest2 ved å sette sammen digest1, den autoritative motorens snmpEngineID verdi, og digest1.
2. Hvis en 16-oktetts nøkkel ønskes, beregnes MD5 hashen av digest2. Hvis en 20-oktetts nøkkel ønskes, beregnes SHA-1 hashen av digest2. Resultatet er brukerens lokaliserte nøkkel.

Den beregnede lokaliserte nøkkelen kan konfigureres på en sikker måte i agentens system. På grunn av enveis egenskapene til MD5 og SHA-1, er det umulig for en motstander å finne ut av brukernøkkelen, selv om han klarer å avsløre en lokalisert nøkkel.

I motsetning til manuell nøkkelhåndtering, har SNMPv3 USM automatisk nøkkeloppdatering etter en konfigurert tid. Prosessen er definert som følger:

- Den nye nøkkelen krypteres ved å bruke den gamle nøkkelen som krypteringsnøkkel.
- En enveis hash funksjon brukes for å produsere en verdi fra den gamle nøkkelen.
- Denne verdien XOR-es med den nye nøkkelen og sendes til agenten.
- Agenten kan XOR-e det innkommende resultatet med den sammen funksjonen anvendt på den gamle nøkkelen for å produsere den nye.

Imidlertid må de passordgenererte nøklene være gitt i utgangspunktet.

5.1.2 View-Based Access Control Module (VACM)

Den view-baserte aksesskontrolmodulen (VACM) [47] opererer som nevnt på PDU prosesseringsnivået og bestemmer hvorvidt en gitt prinsipal har adgang til gitte MIB objekter for å utføre gitte funksjoner (read, write, notify). For at dette i det hele tatt skal være meningsfullt må den forespørrende entiteten bli autentisert skikkelig.

For denne avhandlingen er det ikke vesentlig å kjenne til hvordan denne fungerer, men mer at denne funksjonen eksisterer. Den er kanskje en av hovedforskjellene i forhold til de andre sikkerhetsløsningene (IPSec og TLS), siden de ikke opererer på applikasjonsnivået.

Aksessgodkjenningen i VACM logikken foregår på følgende måte:

- Først identifiseres hvilken sikkerhetsmodell og sikkerhetsnavn (hvem) som vil ha aksess.
- Så blir dette kombinert inn i en aksesstabell med følgende andre parametere: i hvilken kontekst (hvor) objektene finnes, med hvilken sikkerhetsmodell og nivå (hvordan) de skal aksesseres, og for hvilken operasjon (hvorfor) av les, skriv og varsle de skal aksesseres.
- Så sammenliknes denne aksesstabellen med hva slags type objekt (hva) og hvilken instans (hvilken) av denne den forespørrende entiteten ønsker å aksessere.
- Hvis alt dette samsvarer med tillatte operasjoner blir aksessen godkjent.

5.2 Sikkerhetsnivå

Protokollag
SNMPv3 opererer på applikasjonslaget. Den tilbyr ende-til-ende sikkerhet for hele, eller deler av SNMP meldingene. Fordeler med å operere på applikasjonslaget er spesifisering av sikkerhetstjenester uavhengig av andre lag, og muligheten for aksess kontroll. Ulempen er at løsningen kun kan brukes for den gitte applikasjonen, og at man dermed må operere med flere sikkerhetsløsninger hvis man i tillegg ønsker å sikre noe annet.

Nøkkelhåndtering
Måten nøkler håndteres på for SNMPv3 USM er ganske særegen. Brukernøkler lages fra manuelle brukerpassord som utvides og hashes. Hashingen skal gjøre det mer tidkrevende å søke seg frem til riktig brukerpassord. Men samtidig gir brukerpassord et mye mindre utfallsrom for sannsynlige nøkler. Ingen nøkler overføres over SNMP meldinger, og en bruker får en unik nøkkel med hver motor. Hvis en agent kompromitteres er resten av systemet beskyttet fordi nøkkelen som hver agent deler med en bruker er unik. Nye nøkler utveksles kryptert med den gamle. Forutsetningen her er å legge inn brukerpassordet for hver ny bruker og agent denne skal kommunisere med.

Data autentisering
SNMPv3 USM bruker en Hashed Message Authentication Code (HMAC) for dataintegritet og autentisering av dataopprinnelse (bruker) for hele SNMP meldingen. Tilgjengelige hash funksjoner er HMAC-MD5 eller HMAC-SHA-1 som henholdsvis bruker en 128-bits og en 160-bits autentiseringsnøkkel.

Avspillingsbeskyttelse
SNMPv3 USM har en egen tidslinjefunksjon for beskyttelse mot forsinkelse eller avspilling av pakker. Denne baserer seg på tidsparametere for SNMP motorene som det skjer en egen utveksling for i hver melding, og opererer i så måte uavhengig av andre systemer.

Konfidensialitet
For konfidensialitetsbeskyttelse av 'Scoped PDU' delen av meldingen, brukes symmetrisk kryptografi. Algoritmen som er tilgjengelig for SNMPv3 USM er den symmetriske block cipheren Data Encryption Standard (DES) i Cipher Block Chaining (CBC) modus med en 64-bits initialiserings vektor (IV) og 56-bits nøkkel, henholdsvis beregnet fra, og tatt direkte fra siste og første halvdel av et 128-bits nøkkelinput. Ingen andre algoritmer er for tiden spesifisert for SNMPv3 USM. Men det arbeides med et Internet Draft om innarbeiding av den symmetrisk block cipheren Advanced Encryption Standard (AES) i Cipher FeedBack (CFB) modus med 128-bits initialiserings vektor (IV) og en 128-bits nøkkel [68]. I tillegg undersøkes og vurderes det allerede nå aktivt internt i Cisco Systems å implementere denne. Dette i følge produktmanager for Cisco IOS management, Mark Basinski i Cisco Systems.

Aksesskontroll
SNMPv3 har en egen modul for aksesskontroll kalt User-based Access Control Module (VACM) som opererer på 'Protocol Data Units (PDU) i applikasjonslaget. Modulen skiller aksesskontroll helt ned på brukernivå, hvilket er gunstig i forhold til det å kunne kjøre en stram policy for adgang og operasjoner på objekter. Den har en oppbygning hvor brukere kan tilhøre en eller flere grupper, hvilket gjør løsningen meget skalerbar.

Kombinasjoner
De forskjellige kombinasjonene av sikkerhetsnivå for SNMPv3 USM er

1. Ingen autentisering eller konfidensialitet. Da fungerer brukernavnet som en identifikator på omtrent samme måte som en 'community-string'.
2. I tillegg autentisering av hele meldingen.
3. I tillegg kryptering av Scoped PDU.

Det gir for SNMPv3 USM ikke mening å tilby konfidensialitet uten autentisering.

Totalvurdering
Totalt sett er SNMPv3 USM ser i utganspunktet ut til å være bra balansert. Men det sannsynlige utfallsrommet for brukerpassordene er betraktelig mindre enn vanlige random genererte passord. I tillegg er 56-bits kryptering med Single DES i CBC modus noe svakt.

5.3 Konfigurasjon

Brukergrensesnitt
Cisco ruteren (agenten) har et hierarkisk kommandolinjebasert brukergrensesnitt. Den har også forskjellige aksessnivå for forskjellige operasjoner.

Laptopen (manageren) har også et kommandolinjebasert brukergrensesnitt for å kjøre kommandoer mot agenten. I tillegg har denne konfigurasjonsfiler hvor standard parametere kan legges inn, slik at kommandoen kan startes uten å spesifisere ofte gjentatte parametere.

Installasjon
For agentsiden (ruteren) er installasjonen en del av IOS programvaren. Vi hadde en utfordring her i form av at vi måtte installere nytt IOS software sett, for å få SNMPv3 USM til å virke med konfidensialitet (1DES). Dette til tross for at slik støtte var en del av kravspesifikasjonen til vår oversendt til leverandøren. Softwaresettet som vi måtte etterinstallerer het IPPluss/IPSec/3DES. Vanskeligst var det å overføre programmet til ruteren via Trivial File Transfer Protocol (TFTP).

For manageren kom programvaren til net-snmp som standard med Redhat distribusjonen. Men den kunne også lastes ned som et 'Red Hat Package Manager program' (RPM), som pakker ut og installerer seg selv.

Konfigurasjonsparametere
For å konfigurere en SNMPv3 agent, må det settes en hel del parametere med avhengighet til hverandre. Parametere som måtte konfigureres for å kunne kjøre operasjoner mot ruteren var: view, aksessliste, gruppe og bruker [69]. For å forenkle jobben brukte vi samme view hele tiden. Gruppene fikk forskjellige rettigheter til 'viewet' (read, write, notify) og forskjellig sikkerhetsnivå avhengig av rettighetene (ingen sikkerhet, autentisering og konfidensialitet inklusive autentisering). Hver bruker fikk sikkerhetsparametere og ble tillagt i en gruppe. Det hele tok litt tid å få kontroll på, men når det først var gjort en gang, var det lettere å gjøre på nytt. Forandringen av parametere oppdaterte seg umiddelbart i ruteren, så det var ikke nødvendig å f.eks. restarte programvaren.

Konfigurasjonsparametrene på agenten var få men effektive. Vi måtte bruke brukernavn og sikkerhetsparametere for operasjon med forskjellige brukere mot ruteren. Dette ble etterhvert veldig enkelt da vi oppdaget at vi kunne bruke en konfigurasjonsfil for gjentatte parametere.

Dokumentasjon
Dokumentasjonen for SNMPv3 for Cisco var relativt bra og dekkende, men utfordrende å finne frem til for en som er ukjent med å navigere på deres hjemmesider. I brukergrensesnittet er det mulig å få hjelp til å finne rette kommandoer ved å navigere seg gjennom subkommandoer. Antall brukere av SNMPv3 er imidlertid svært begrenset, så derfor var det vanskelig å finne andre f.eks. på diskusjonsforum eller e-postlister som kunne noe særlig om emnet. Heldigvis hjalp en supportavtale med leverandøren litt på dette. I tillegg fikk jeg igjennom Forsvaret, i stand et møte med fagpersonell innen management og sikkerhet i Cisco Systems Norge, og kom gjennom dem i kontakt med produktmanager for blant annet SNMPv3 i deres IOS programvare.

Dokumentasjonen på net-snmp pakken og SNMPv3 USM funksjonalitet i denne var imidlertid svært begrenset. Men gjennom å sammenstille forskjellig informasjon var det mulig å få en ide om hvordan det måtte gjøres. Dette er også muligens et resultat av at SNMPv3 er lite brukt.

Skalerbarhet
SNMPv3 inklusive sikkerhet er relativt greit å konfigurere i et begrenset omfang. Dersom løsningen oppnår en viss størrelse blir det mye arbeid å legge inn brukerparametere inklusive aksessrettigheter for hver ny agent. Men dette kan tas fortløpende etter hvert som antallet agenter vokser. Antall agenter som kan operere i et nett er begrenset av managerens kapasitet. Men ved bruk en distribuert tilnærming med mellomliggende managere er det mulig å konfigurere et ganske stort system.

For kommando-linje managere var det lettere å legge inn parametere, da disse kun trenger riktig brukernavn med tilhørende passord for generering av autentiserings- og konfidensialitetsnøkler. Hvordan dette fungerer for overordnede managere (network management stations) slik som for eksempel HP OpenView, har vi imidlertid ikke fått noen kjennskap til.

Vedlikehold
Det viktigste med vedlikeholdet av SNMPv3 er nøkkeloppdatering. Dette er godt håndtert med en automatisk prosedyre for oppdatering av nøklene utledet fra brukerpassordene. Oppdatering av programvaren er spesifikk for implementasjonene. For net-snmp kan programvaren lastes inn på hosten og kjøres med en egen oppdateringskommando, uten at konfigurasjonsfilene går tapt.

Oppdatering av programvaren for SNMP på ruteren betyr imidlertid utskiftning av hele operativsystemet, noe som fører til at ruteren vil være utilgjengelig en liten stund. Selve konfigurasjonen antas at enkelt kan kopieres og tilbakeføres etter oppdatering, men vi har imidlertid ikke testet dette i praksis.

5.4 Ytelse

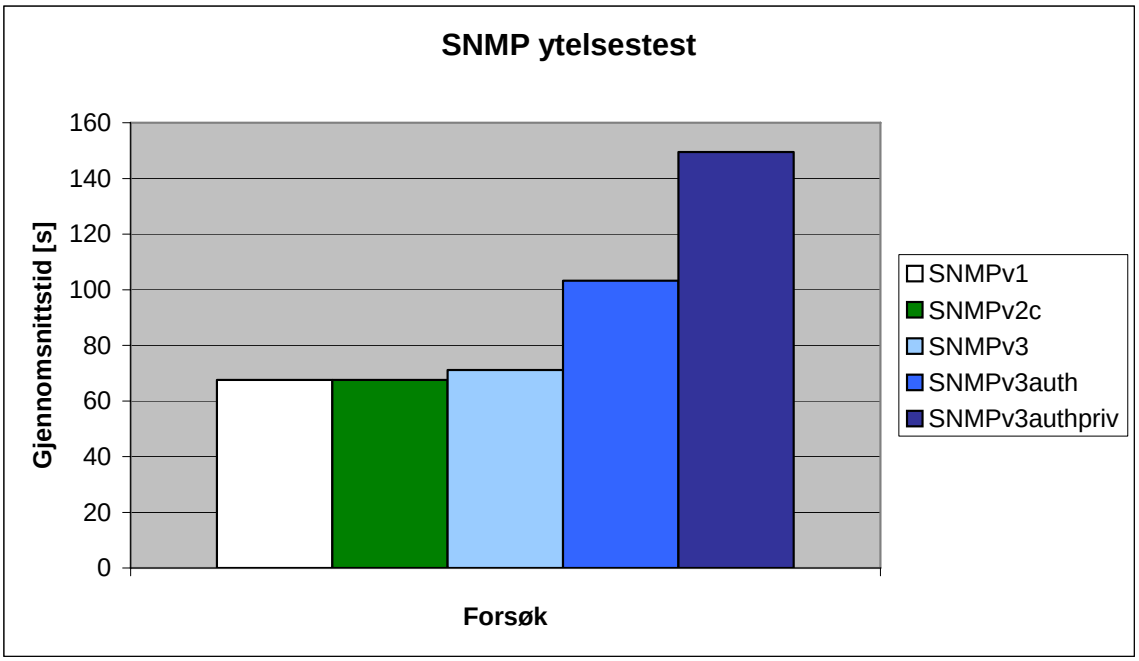
Vi har gjennomført fem forskjellige typer ytelsestester, i tre bolker for hver av SNMP versjonene. For SNMPv1 og 2c har det kun vært utført et testsett for hver versjon. Mens for SNMPv3 har det vært utført ett testsett for hvert sikkerhetsnivå. Disse er henholdsvis ingen sikkerhetsfunksjoner, dataautentisering og konfidensialitet med dataautentisering. For alle forsøkene har det blitt gjort 30 repetisjoner for å oppnå en god statistisk signifikans.

Vi repeterer at det som måles, er tiden det tar å hente ned til manageren alle objektvariable fra management informasjonsbasen med operasjonen get.

Parametere	SNMPv1	SNMPv2c	SNMPv3	SNMPv3auth	SNMPv3authpriv
Sum	2025,123322	2026,881221	2135,651627	3099,188129	4484,141967
Antall repetisjoner	30	30	30	30	30
Gjennomsnitt	67,504111	67,562707	71,188388	103,306271	149,471399
Middelverdi	67,251250	67,246838	71,236857	103,356010	149,233216
Minverdi	67,038164	67,000081	70,186751	102,049329	149,016070
Maksverdi	68,148882	68,731982	71,767064	104,822379	151,007303
Standardavvik	0,45103490	0,49995249	0,38645525	0,65369002	0,49396587
Standardfeil	0,08234733	0,09127842	0,07055675	0,11934692	0,09018542
Varians	0,20343248	0,24995249	0,14934766	0,42731064	0,24400228
Konfidensgrad(95,0%)	0,16841929	0,18668543	0,14430484	0,24409200	0,18444999

Tabell 1: Resultat fra SNMP ytelsestester

Tabell 1 viser det statistiske resultatet fra de fem forsøkene. Vi ser at standardavviket for gjennomsnittet er relativt liten. Tallmaterialet er med andre ord relativt solid.

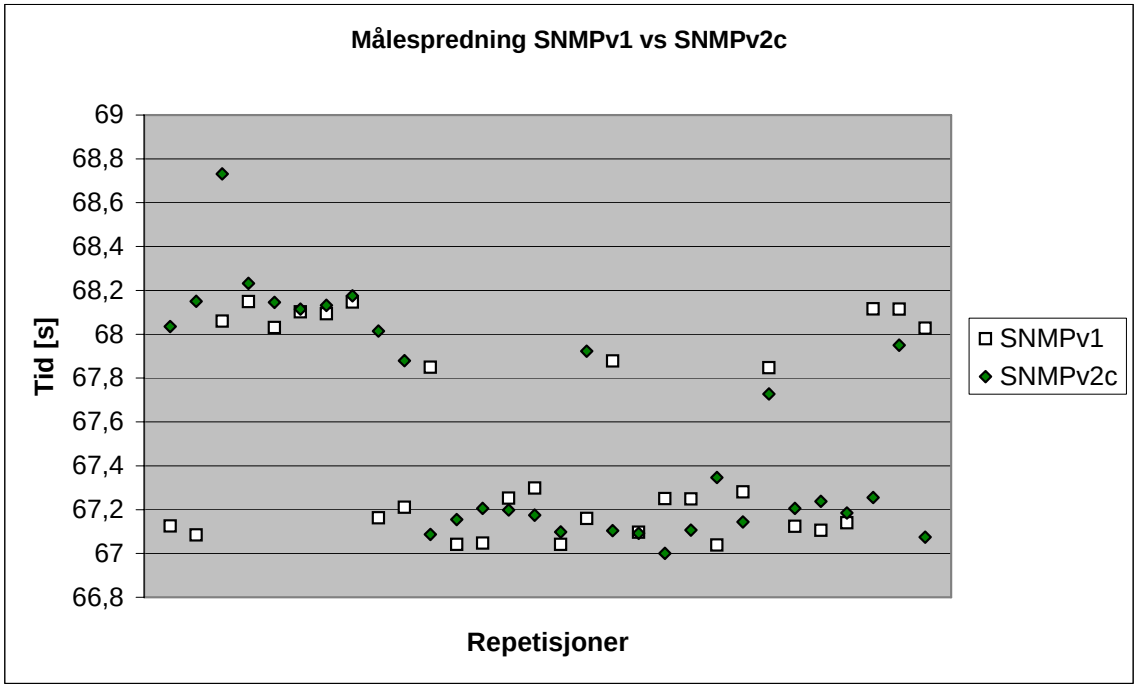


Figur 10: Resultat fra SNMP ytelsestester

Figur 10 viser en grafisk fremstilling av gjennomsnittet for hver av de fem forsøkene. Standardavviket er så lite at det ikke er synlig på grafen.

Vi ser at SNMPv1 og SNMPv2 kommer relativt likt ut, som forventet da disse har samme meldingsformat for testoperasjonen. SNMPv3 uten noen sikkerhetsfunksjoner tar litt lengre tid, også dette som forventet da denne har et litt større meldingsformat.

Tillegget for SNMPv3 med autentisering er omtrent 45%, mens tillegget for både autentisering og konfidensialitet er på hele 110%, og differansen mellom disse er da på 65% (relativt til SNMPv3 uten sikkerhetsfunksjoner).



Figur 11: Spredning av måleresultat for SNMPv1 og SNMPv2c

Vi ser av Figur 11 at forskjellen for alle måleresultatene mellom SNMPv1 og SNMPv2c er veldig liten. Spredningen er ganske lik, noe som gir dem omtrent samme konfidensintervall. Med andre ord har vi overlapp i store deler av konfidensintervallet for de to forsøkene. På bakgrunn av dette vil vi videre i denne hovedoppgaven ta utgangspunkt i at SNMPv1 og SNMPv2c er like, og kun bruke SNMPv1 av de to for videre målinger.

6 Internet Protocol Security (IPSec)

Dette kapittelet tar for seg Internet Protocol Security (IPSec). Det begynner med en presentasjon av protokollen, og tar så for seg hver av de tre evalueringsparametrene.

6.1 Protokollen

Internet Protocol Security (IPSec) finner sted på nettverkslaget. Den utfører sikkerhet for IP-datagram fra enhet til enhet, uavhengig av underliggende og overliggende lag.

Arbeidsgruppen for IPSec [70] publiserte spesifikasjonen første gang i august 1995, og en revisjon i november 1998. De viktigste dokumentene er: sikkerhetsarkitekturen [71] , et pakkeformat Authentication Header (AH) [72] med autentisering, et pakkeformat Encasulation Security Payload (ESP) [73] med kryptering og valgfri autentisering, definisjon av et rammeverk for nøkkelhåndteringen Internet Security Association and Key Management Protocol (ISAKMP) [74] og definisjon av en implementasjon i dette rammeverket Internet Key Exchange (IKE) [75].

Tjenester
Sikkerhetstjenestene for IP-datagrammene implementeres gjennom de nevnte tilleggsheaderene. Vi har tre variasjoner: Authentication Header (AH) som autentiserer datagrammet, Encapsulation Security Payload (ESP) som krypterer datagrammet, og ESP med en valgfri autentisering.

Følgende tjenester inngår i de tre løsningene:
- Forbindelsesløs integritet – ikke for ESP med kun krypto
- Autentisering av dataopprinnelse – ikke for ESP med kun krypto
- Avvisning av avspilte pakker (en form for delvis sekvensintegritet)
- Konfidensialitet (kryptering) – ikke for AH
- Konfidensialitet av trafikkflyten (en begrenset form) – ikke for AH

6.1.1 Sikkerhetsassosiasjoner (SA)

Sikkerhetsassosiasjoner er et sentralt konsept i IPSec. En sikkerhetsassosiasjon er et enveisforhold mellom en sender og mottaker hvor det ytes sikkerhetstjenester til trafikken som bæres på forbindelsen mellom dem. For å få et gjensidig forhold med toveiskommunikasjon trengs det to sikkerhetsassosiasjoner. AH og ESP kan ikke blandes i samme SA.

En sikkerhetsassosiasjon er unikt identifisert ved tre parametere:
- Security Parameters Indeks (SPI): En identifikator (bitstreng) som mottakeren bruker for å kunne prosessere pakken i rett sikkerhetsassosiasjon (SA).
- IP Destination Address: Adressen for sikkerhetsassosiasjonens endepunkt, for tiden kun unicast adresser.
- Security Protocol Identifier: Identifikator for type sikkerhetsassosiasjon (AH eller ESP)

For hvert IP-datagram identifiseres sikkerhetsassosiasjonen unikt ved destinasjonsadressen i hodet og SPI-en i den lukkede utvidede meldingsheaderen (AH eller ESP).

En database over sikkerhetsassosiasjoner for å definere parametrene assosiert for hver SA, er en del av hver IPSec implementasjon. En SA er vanligvis definert av følgende krav:

- Sequence Number Counter: en verdi for å generere sekvens nummer
- Sequence Counter Overflow: flagg identifisering ved overløp av sekvens nummer telleren.
- Anti-Replay vindu: brukes for å avgjøre om de inkommende datagram er gjentakelser.
- AH informasjon: Autentiseringsalgoritme, nøkler, nøkkellevetid, og relaterte parametre som brukes med AH.
- ESP informasjon: Kryptering og autentiserings algoritme, nøkler, initieringsverdier, nøkkel levetid, og relaterte parametere brukt av ESP.
- Levetid til sikkerhetsassosiasjonen: Hvor lenge sikkerhetsassosiasjonen kan brukes før den må byttes ut.
- IPSec Protocol Mode: Transport eller tunnel modus, dette tas opp i det kommende avsnitt, eller et wildcard.
- Path MTU: Maksimum transmisjonsenheter (MTU) før datagram må oppstykkes, og aldringsvariable.

6.1.2 Nøkkelhåndtering

Nøkkelforvaltnings delen av IPSec involverer bestemmelse og distribusjon av hemmelige nøkler. IPSec arkitekturen [71] spesifiserer støtte for to typer nøkkelforvaltning:

- Manuell: En systemadministrator konfigurerer manuelt hvert system med algoritmer, egne nøkler og med nøkler fra andre kommuniserende systemer.
- Automatisk (ISAKMP/IKE): Protokoller for automatisk nøkkel og sesjonsforvaltning som muliggjør å sette opp Sikkerhetsassosiasjoner (SA) med nøkler på forespørsel, og letter bruken av nøkler i større voksende distribuerte systemer.

ISAKMP/IKE er standardmetoden for nøkkelhåndtering i IPSec, og består av følgende elementer:

- *Internet Security Association and Key Management Protocol (ISAKMP) [74]:* ISAKMP er et protokollrammeverk som definerer formater for nyttelast, mekanismer for implementering av protokoller for nøkkelutveksling og forhandling av sikkerhetsassosiasjoner.
- *Internet Key Exchange (IKE) [75]:* IKE er en protokoll for nøkkelutveksling som er implementert innen rammeverket til ISAKMP. IKE sørger for autentisering av IPSec kommunikasjonsmotparter, forhandling av IPSec nøkler og forhandling av IPSec sikkerhetsassosiasjoner.

ISAKMP

ISAKMP dikterer ikke selv noen spesifikke algoritmer for nøkkelutveksling; ISAKMP består heller av et sett av meldingstyper som støtter bruken av en mengde algoritmer for nøkkelutveksling.

Spesifikasjonen definerer fem standard utvekslingstyper som støttes:

- Base Utveksling: Tillater nøkkelutveksling og autentifiseringmateriale å sendes sammen, noe som minimaliserer antall utvekslinger på kostnaden av ikke å sørge for identitetsbeskyttelse.
- Identitet beskyttelses utveksling: Utvider 'Base Utveksling' til å beskytte brukerens identitet.
- Bare autentiserings utveksling: Brukes for å utføre gjensidig autentisering uten nøkkel utveksling.
- Aggressiv Utveksling: Minimaliserer antall utvekslinger på kostnad av ikke å sørge for identitetsbeskyttelse.

IKE
Internet Key Exchange (IKE) er en hybrid av de to nøkkelutvekslingsprotokollene Oakley og Scheme, som definerer hvordan autentisert nøkkelmateriale utledes, hvor Scheme i tillegg definerer en metode for rask nøkkeloppdatering.

IKE algoritmen er karakterisert ved fem viktige trekk:
1. Den anvender en mekanisme kjent som cookies for å hindre clogging angrep (tilstopping).
2. Den muliggjør at de to partene forhandler en gruppe; dette spesifiserer i korte trekk de globale parametre for Diffie-Hellman nøkkelutveksling.
3. Den bruker nonces for å sikre seg mot avspillingsangrep.
4. Den muliggjør utveksling av verdier for offentlige Diffie-Hellman nøkler.
5. Den autentiserer Diffie-Hellman utvekslinger for å hindre man-in-the-middle angrep.

Tre forskjellige autentiseringsmetoder kan brukes med IKE:
- *Digitale signaturer*: Utvekslingen autentiseres ved å signere en gjensidig anskaffbar hash; hver part krypterer hashen med deres private nøkkel. Hashen er generert gjennom viktige parametere, slik som bruker-id og nonces.
- *Offentlig nøkkel kryptering*: Utvekslingen autentiseres ved krypteringsparametere slik som ID-er og nonces med senderens private nøkkel.
- *Symetrisk nøkkel kryptering*: En nøkkel avledet av en utenforstående mekanisme kan brukes for å autentisere utvekslingen med symmetrisk kryptering av utvekslingsparametrene.

Vi vil her ikke gå mer i dybden på ISAKMP og IKE.

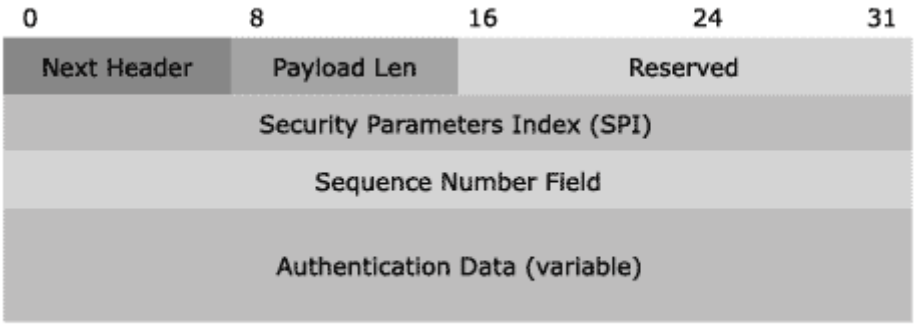
6.1.3 Transport og tunnel modus

Både AH og ESP støtter to modus: transport- og tunnelmodus. Transportmodus beskytter hovedsakelig høyere lags protokoller i form av nyttelasten til datagrammet. Dette brukes typisk for ende-til-ende kommunikasjon mellom to entiteter (f.eks. manager og agent). Tunnelmodus beskytter hele datagrammet, ved å pakke dette inn i et nytt datagram med en egen IP header etter at AH eller ESP feltet er lagt til. Hele det originale datagrammet går gjennom en tunnel fra et punkt av IP nettverket til et annet, og ingen entiteter (rutere) underveis er i stand til å undersøke det indre IP hodet.

Vi har i denne oppgaven best nytte av transportmodus, da vi skal beskytte SNMP trafikk fra ende til ende (manager til agent). Videre ønsker vi å teste både AH (autentisering) og ESP (konfidensialitet og autentisering)

6.1.4 Authentication Header (AH)

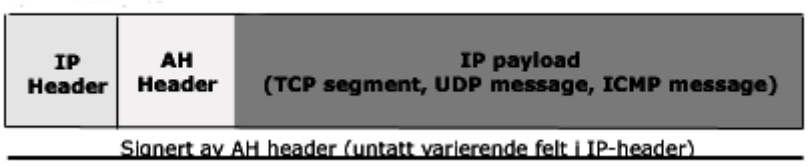
Authentication Header (AH) [72] sørger for data autentisering, data integritet, og valgfri anti replay tjenester. AH er innesluttet i de data som skal beskyttes (et fullt IP datagram). Autentiseringen er basert på bruken av meldingsautentiseringskode (MAC), med en felles hash kode (MD5 eller SHA-1) mellom de to partene.



Figur 12: Authentication Header (AH) formatet

AH hodet (Figur 12) inneholder de følgende felter:

- Neste hode – Identifiserer IP nyttelasten ved å bruke IP protokoll ID. For eksempel, verdien 6 representerer TCP.
- Lengde – Identifiserer lengden på AH hodet.
- Sikkerhetsparameterindeks (SPI) – brukt i kombinasjon med destinasjons adressen og sikkerhetsprotokollen (AH eller ESP) for å identifisere den korrekte sikkerhetsassosiasjon for kommunikasjonen. Mottakeren bruker denne verdi for å avgjøre med hvilken sikkerhetsassosiasjon pakken er identifisert.
- Sekvens nummer – Sørger for anti replay beskyttelse for pakken. Dette er et 32-bits inkrementerende tall, som ikke kan gjentas på grunn av levetiden til assosiasjonen. Dersom en pakke gjentas, avvises denne.
- Autentiseringsdata – Inneholder integritessjekk verdien (ICV), også kjent som meldings autentikasjons koden (MAC), som brukes for å verifisere både meldingsautentisering og dataintegritet. Mottakeren regner ut ICV verdien og sjekker den mot denne verdien (som blir regnet ut av avsender) for å verifisere integriteten. ICV blir regnet ut gjennom IP hodet, AH hodet og IP nyttelasten.

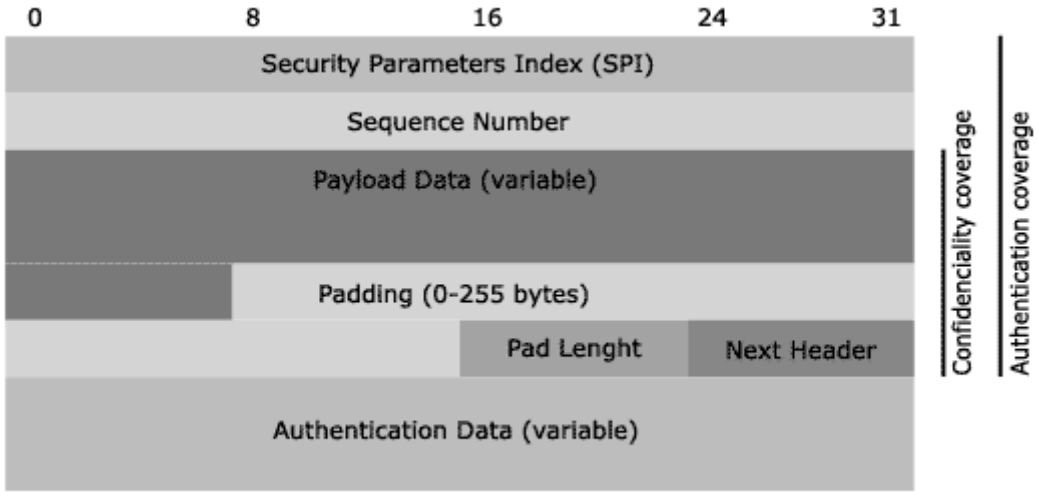


Figur 13: AH transport modus

Figur 13 illustrerer hvordan AH tillegsheaderen settes sammen i datagrammet for transport modus. Legg merke til at hele datagrammet (med unntak av varierende headerfelt som time to live) er signert

6.1.5 Encapsulating Security Payload (ESP)

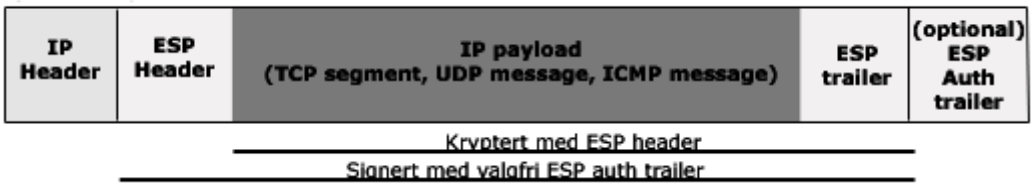
Encapsulation Security Payload (ESP) [73] besørger data konfidensialitet, data integritet, og beskyttelsestjenester, i tillegg en valgfri autentisering av dataopprinnelse, og anti replay tjenester. Krypteringen er basert på symmetrisk block cipher (eks. DES, 3DES – CBC modus, AES osv.). Autentiseringen er basert på meldingsautentiseringskode (MAC) med en felles hash kode (MD5 eller SHA-1) mellom de to partene.



Figur 14: IPSec Encapsulation Security Payload (ESP)

ESP hodet (Figur 14) inneholder følgende felt:

- Security Parameters Index (SPI): identifiserer sikkerhetsassosiasjonen.
- Sekvensnummer: ett inkrementelt tall som beskytter mot avspilling av datagram.
- Nyttelast data: Dette er et transportnivå segment (transport modus) eller IP-pakken (tunnel modus) som beskyttes ved kryptering.
- Padding: Besørge riktig blokkstørrelse for 'block cipher' kryptering, og trafikkflyt konfidensialitet i form av ukjent meldingslengde.
- Pad length: Indikerer antall utfyllingsbit umiddelbart før dette feltet.
- Next header: Identifiserer neste hode i nyttelast dataene.
- Autentiseringsdata: et felt av variabel lengde som inneholder integritets sjekk verdi (ICV) beregnet gjennom ESP pakken minus feltet for autentiseringsdata.



Figur 15: ESP transport modus

Figur 15 illustrerer hvordan ESP tillegsheadere og trailere settes sammen for datagrammet i transportmodus. Legg merke til at kun nyttelasten og ESP traileren (padding) er kryptert, og at signeringen i tillegg dekker kun ESP headeren (ikke IP Headeren).

6.2 Sikkerhetsnivå

Protokollag
IPSec opererer på nettverkslaget. Ved hjelp av en egen applikasjonsprotokoll forhandler den frem ende-til-ende sikkerhet for nyttelasten til IP datagrammet. Fordeler med denne løsningen er at den kan brukes til å sikre alt som går over IP protokollen, og derfor er veldig anvendelig for å sikre flere trafikktyper samtidig.

Nøkkelhåndtering
Forvaltning av nøkler i IPSec kan enten foregå manuelt eller automatisk. Automatisk nøkkelforvaltning foregår innen rammeverket til ISAKMP, hvor IKE er den mest aktuelle implementasjonen. Denne autentiserer motparten og setter opp en sikker kanal for å forhandle sesjonsnøkler for en sikkerhetsassosiasjon. Autentiseringen kan skje på følgende tre måter: ved hjelp av digitale signaturer, offentlig nøkkelkryptering eller symmetrisk nøkkelkryptering. Sikkerhetsassosiasjonen fornyes etter en gitt tid slik at ingen får tid til å knekke krypteringsnøkkelen. Fornyelse kan skje ved hjelp av den gamle sesjonsnøkkelen eller i en separat sikker kanal dedikert for nøkkelutveksling. Det sistnevnte kalles Perfect Forward Secrecy (PFS).

Dataautentisering
IPSec AH og ESP bruker i likhet med SNMPv3 USM Hased Message Authentication Code (HMAC) for dataintegritet og autentisering av dataopprinnelse. Forskjellen er at denne har definert tre hash algoritmer. HMAC-MD5 [76] bruker en 128s-bit nøkkel, HMAC-SHA1 [77] bruker en 160-bits nøkkel og den nye algoritmen HMAC-RIPEMD [78] bruker en 160-bits nøkkel. For transport modus autentiserer AH hele meldingen inklusive IP-header, mens ESP valgfritt autentiserer ESP headeren og payloaden.

Avspillingsbeskyttelse
Avspillingsbeskyttelsen i IPSec skjer i form av at det er lagt inn sekvensnummer i både AH og ESP headeren.

Konfidensialitet
IPSec ESP støtter et stort spekter av krypteringsalgoritmer. I vårt testscenario hadde vi imidlertid kun tilgang til triple DES. Ruteren støttet både 56-bits CBC Single DES og Triple DES, men FreeS/WAN implementasjonen støttet kun Triple DES, da utviklerne mener Single DES er så dårlig at de nekter å støttet den. Vi forsøkte oss imidlertid med en sidestrømsimplementasjon kalt SuperFreeS/WAN [79], men vi hadde problemer med å kompilere den inn i en ny kjerne. Denne ville gitt oss tilgang til en rekke flere algoritmer inklusive 1DES og nullkrypteringsalgoritmen. I tillegg må det nevnes at krypteringen i transport modus, kun dekker IP-payloaden.

Aksesskontroll
IPSec har ingen egen mekanisme for aksesskontroll da denne opererer på nettverkslaget. For vårt scenario er det imidlertid aktuelt å benytte seg av aksesskontrollen til SNMPv1. Denne bruker community-strengen til å identifisere og gi aksesstilgang. Dette er ingen sterk form for aksesskontroll, men muligheten for å differensiere mellom forskjellige communities er i hvert fall der. Skaleringen med communities er veldig grov og det er derfor vanskeligere å identifisere hvem som for eksempel utførte en operasjon (notarisasjon).

Kombinasjoner
IPSec meldinger kan transporteres i enten transport eller tunnel modus. Vi valgte å bruke transport modus da denne er skreddersydd for sikkerhet fra ende til ende, og dette er mest relevant for SNMP trafikk. Videre hadde vi kun mulighet for å bruke ESP i vår freeswan ipsec implementasjon. Vi fikk derfor ikke mulighet for å teste AH, men brukte ESP med nullkryptering istedenfor for å få til oppsett med kun autentisering.

Totalvurdering
Totalt sett fremstår IPSec en veldig god sikkerhetsløsning, særlig med tanke på tilgang til sterke kryptoalgoritmer for konfidensialitetsbeskyttelse og at den beskytter hele IP-payloaden.

6.3 Konfigurasjon

Brukergrensesnitt
Cisco 2621 ruteren bruker alltid et hierarkisk kommandolinjebasert grensesnitt, med forskjellige privilegerte tilgangsnivåer. FreeS/WAN 1.99 implementasjonen av IPSec opererer for det meste med konfigurasjonsfiler men også kommandolinje for å sette opp sikkerhetsassosiasjoner.

Installasjon
IPSec skulle også være en del av IOS pakken som ble levert med ruteren. Imidlertid var den ikke det, så vi måtte bestille på nytt fra leverandøren for å få et 'feature set' av type 'IPPluss/IPSec/3DES' Det var tungvint å laste dette nye IOS-et over på ruteren via TFTP.

Videre var installasjonen (og konfigurasjonen) av FreeS/WAN kanskje den aller største utfordringen. Freeswan kom originalt som en RPM pakke, som kunne lastes ned via prosjektsiden. Det viste seg imidlertid at standardversjonen verken støttet 1des eller nullkryptering. Derfor prøvde vi å installere Super-FreeS/WAN som er en patchet versjon av freeswan. For at denne skulle virke måtte man bygge en ny kjerne for IPSec modulen 'Klips'. Dette var imidlertid også svært krevende, og det endte til slutt med at vi fant kilde rpm-er for ferdig bygde kjerner med super-freeswan. Denne var imidlertid heller ikke rett frem å installere, men tilslutt med litt brukerhjelp gikk dette bra. Det viste seg også at det var en bug i et av de sentrale skriptene for den pakkede versjonen av super-freeswan. Vi fikk imidlertid til slutt identifisert feilen, og tilsendt en fiks fra utvikleren [80].

Konfigurasjonsparametere
For å konfigurere IPSec, så må en rekke parametere settes. På ruteren måtte følgende gjøres:

- Lage utvidet tilgangsliste for å definere hvilke trafikk som skulle krypteres.
- Lage IPSec transform sett, som sier i fra om ønskede protokoller (AH eller ESP), kryptografiske algoritmer, og operasjonsmodus (tunnel eller transport).
- Lage et 'crypto-map' som inneholder: mottaker, tilgangsliste, transform sett, og eventuelle SPI parametre for manuelt oppsett av sikkerhetsassosiasjonen.
- Tilegne 'crypto-map' til et grensesnitt.

FreeS/WAN implementasjonen måtte konfigureres via en konfigurasjonsfil. Her måtte de korresponderende parametere settes. Dette opplevdes som utfordrende, siden notasjonen er forskjellig mellom ruteren og freeswan, og tilgangen til dokumentasjon var dårlig. Sikkerhetsassosiasjonen ble enkelt satt opp ved hjelp av ISAKMP. Den kan også settes opp manuelt, men dette er noe mer komplekst.

Dokumentasjon
Cisco sin dokumentasjonen på IPSec var relativt bra. Cisco har en konfigurasjons og kommando referanseguide med hver av hovedversjon for deres IOS.

For FreeS/WAN var derimot dokumentasjonen mangelfull. Det tok undertegnede lang tid for å få de korrekte parametere for i det hele tatt å kommunisere med ruteren. Grundige studier av manualer og svar fra e-post lister var eneste måten å komme videre med problemene på. Og her fortjener freeswan skryt for en stor og hjelpsom community av utviklere og brukere.

Skalerbarhet
Skalerbarhet er et pluss med IPSec. Siden alle transportlag bæres over IP, kan den også brukes av mange andre programmer enn SNMP, uten å måtte endre noe. Nøkkelvedlikehold gjennom offentlignøkkelinfrastrukturen er også et stort pluss med tanke på større systemer. Men IPSec har også viktige utfordringer i forhold til oversetting mellom private og offentlige IP-addresser, såkalt Network Address Translation (NAT) [7], skjønt det finnes måter å håndtere dette på.

Vedlikehold
Vedlikeholdet av IPSec er veldig enkelt. Med en gang en automatisk SA er etablert, blir den også bevart med periodiske oppdateringer og endringer. Manuelle sikkerhetsassosiasjoner kan imidlertid brytes av, slik at disse må settes opp på nytt, eller gå for evig og alltid. Manuell sikkerhetsassosiasjon tilrådes derfor ikke i produksjonsnett. Å oppdatere FreeS/WAN programvare kan gjøres uten å måtte kaste konfigurasjonsfilene. IPSec må imidlertid være nede under oppdatering.

6.4 Ytelse

For IPSec har vi gjennomført seks forskjellige ytelsestester, i to bolker for SNMPv1 og SNMPv3. For begge bolkene har vi testet IPSec med Encapsulation Security Payload (ESP) i transport modus. Så har vi variert på kryptoalgoritme.

Vi var nødt til å bruke ESP med nullkryptering for å få til kun autentisering, da freeswan implementasjonen ikke støtter bruk av kun AH. I tillegg til å teste med 1des for kryptering og autentisering, testet vi også kryptering med 3des. Dette fordi 1des modulen i super-freeswan er utledet fra 3des modulen i freeswan (som nekter å implementer 1des fordi de mener 1des er for svak), og vi dermed hadde en mistanke om at disse ville være ganske like rent ytelsesmessig.

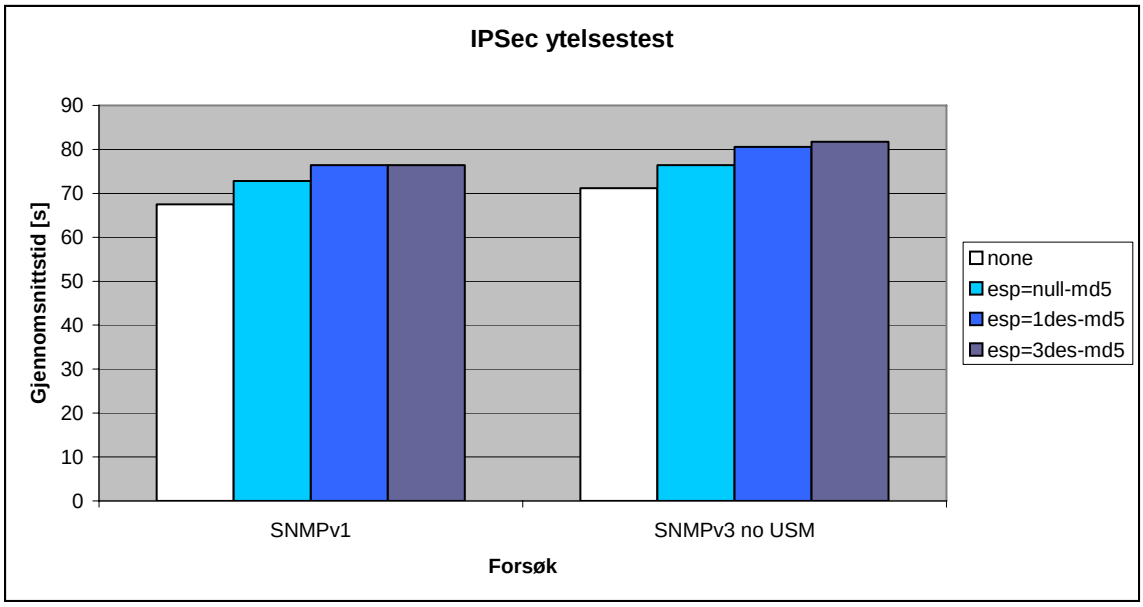
Vi har altså tre tester for hver av SNMP versjonene, en med nullkryptering, en med 1des og en med 3des, hvor alle tre bruker HMAC-MD5 til dataautentisering. For alle forsøkene har det blitt gjort 30 repetisjoner for å oppnå god statistisk varians.

	SNMPv1			SNMPv3		
Parametere/esp crypto	**null**	**1des**	**3des**	**null**	**1des**	**3des**
Sum	2184,92	2293,46	2293,55	2293,56	2417,29	2451,85
Antall repetisjoner	30	30	30	30	30	30
Gjennomsnitt	72,8308	76,4486	76,4518	76,4518	80,5762	81,7282
Middelverdi	73,0925	76,4210	76,2239	76,2240	80,3966	81,8164
Minverdi	72,0300	76,0859	76,0052	76,0052	79,0734	80,7619
Maksverdi	73,8807	76,7697	77,8123	77,8123	81,8322	82,6272
Standardavvik	0,529946	0,176508	0,459776	0,459778	0,682574	0,527685
Standardfeil	0,096754	0,032226	0,083943	0,083944	0,124620	0,096342
Varians	0,280843	0,031155	0,211394	0,211396	0,465908	0,278451
Konfidensgrad(95,0%)	0,197885	0,065909	0,171683	0,171684	0,254878	0,197041

Tabell 2: Resultat fra IPSec ytelsestester

Tabell 2 viser det statistiske resultatet fra de seks forsøkene. Vi ser at standardavviket for gjennomsnittet også her er relativt lite. Vi har med andre ord et godt tallmateriale.

I denne forbindelse er det naturlig å kommentere at for tidligere tester, mens vi fortsatt strevde med IPSec implementasjonen, opplevde vi en endring i testresultatene. Det hadde seg slik at ruteren gjorde et temposkifte for prosessering av pakkene. Vi tror dette kan ha hatt sammenheng med caching i ruterens minne, men dette er bare en antagelse. Imidlertid førte dette til at vi var svært oppmerksomme ovenfor eventuelle temposkifter for de videre testene, inklusive alle endelige tester som dette fenomenet ikke har forekommet for.



Figur 16: Resultat av IPSec ytelsestester

Figur 16 viser en grafisk fremstilling av resultatet fra ytelsestestene for IPSec. Standardavviket er så lite at det ikke er synlig på grafen.

Vi ser at SNMPv3 uten USM, som forventet bruker noen flere sekunder enn SNMPv1 på grunn av ulik størrelse på meldingsformatet. De relative forskjellene innad mellom de to versjonene ser imidlertid ut til å være ganske lik.

Forskjellen mellom ingen sikkerhetstjenester og esp med nullkryptering ligger på henholdsvis 7,4 og 7,9 %. Forskjellen opp til 1des er henholdsvis 13,3 og 9,4 %, som gir en differanse mellom 1des og nullkryptering på henholdsvis 5,4 og 5,8 %. Hoppet videre fra 1des til 3des er ekstremt lite, på henholdsvis 0,3 og 1,6 % (relativt til ingen sikkerhetstjeneste). Dette bekrefter antagelsen om at 1des implementasjonen er ganske lik til 3des, da den er bygget ut fra denne.

Altså er den relative tidskostnaden for å legge til bedre sikkerhetstjenester liten for alle sikkerhetsnivåer. I det hele tatt ser IPSec ut til å være en meget tidsmessig effektiv løsning for å implementer sikkerhetstjenester.

7 Transport Layer Security (TLS)

Dette kapittelet tar for seg Transport Layer Security (TLS). Det begynner med en kort presentasjon av løsningen, og tar så for seg hver av de tre evalueringsparametrene.

7.1 Protokoll

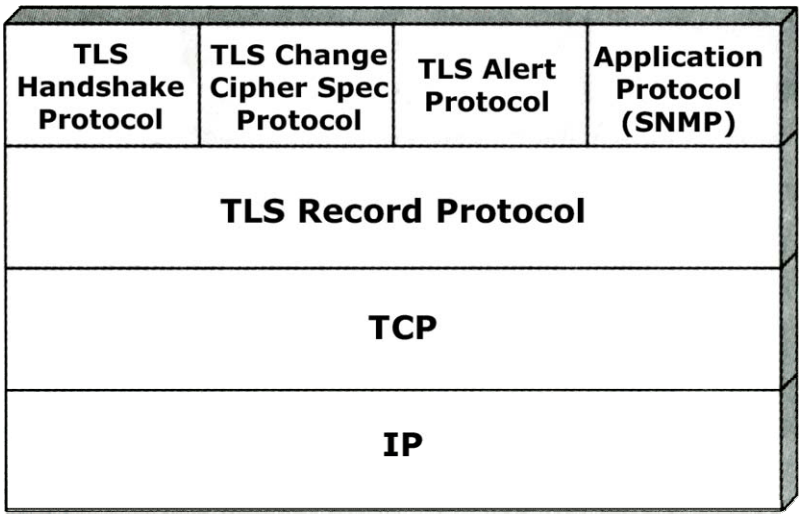
Transport Layer Security (TLS) finner sted på transportlaget. Den utfører sikkerhet for TCP-forbindelser fra ende til ende. TLS er avhengig av et forbindelsesorientert transportlag. Alle applikasjoner som går over TCP kan derfor bruke denne løsningen.

Arbeidsgruppen for TLS [81] ble etablert i 1996 for å standardisere en sikkerhetsprotokoll for transportlaget. Gruppen tok utganspunkt i Netscape sin Secure Socket Layer (SSL) versjon 3.0, og publiserte i 1999 spesifikasjonen for versjon 1.0 av TLS protokollen [82].

Tjenester
Hensikten med TLS er å beskytte klient/tjener kommunikasjon mot:
- Avlytting
- Modifikasjon
- Fabrikasjon

Organisering av TLS
Følgende steg inngår for å kunne ta i bruk TLS klient/tjener kommunikasjonen.
1. Handshake og forhandling av kryptoalgoritmer
2. Autentisering av partene
3. Nøkkelrelatert informasjonsutveksling
4. Utveksling av applikasjonsdata

Punktene som utgjør TLS kan deles opp i to protokoller (Figur 17) som sammen tilbyr forbindelsessikkerhet.
- TLS Handshake Protocol (steg 1 til 3)
- TLS Record Protocol (steg 4)



Figur 17: Visualisering av TLS protokollstakken

7.1.1 Handshake Protocol

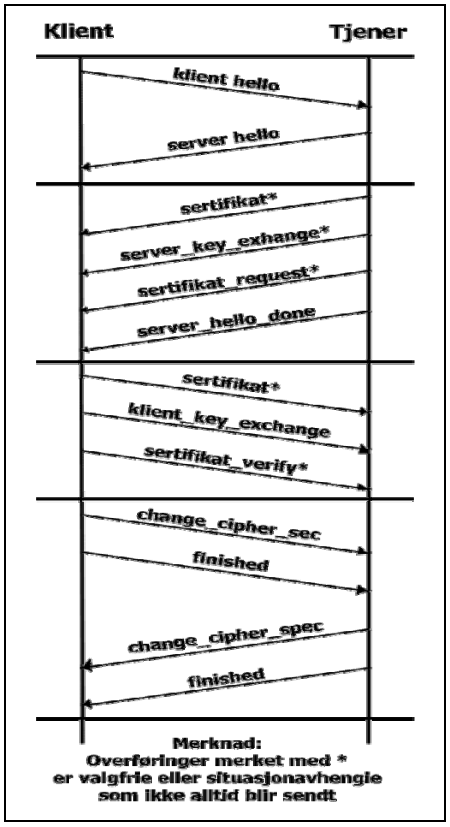
Handshake Protokollen er ansvarlig for autentisering og nøkkelutveksling for å etablere eller gjenopprette sikre sesjoner. Ved etablering av en sikker sesjon, gjør protokollen følgende:

- Forhandler krytpoalgoritme
- Autentiserer serveren og eventuelt klienten
- Utveksler informasjon om sesjonsnøkler

Forhandling av krytpoalgoritme
Klienten og tjeneren tar kontakt og velger kryptoalgoritme som brukes til meldingsutvekslingen.

Autentisering
I TLS beviser tjeneren identiteten sin for klienten. Klienten kan også behøve å bevise sin identitet ovenfor serveren. Denne autentiseringen baserer seg på Public Key Infrastructure (PKI) som tilveiebringer offentlige nøkler på en sikker måte. De eksakte autentiseringsmetodene bestemmes under forhandling av kryptoalgoritme.

Nøkkelutveksling
Klienten og tjeneren utveksler tilfeldige tall og et spesielt tall kalt Pre-Master Secret. Disse tallene kombineres med ytterligere data, slik at klienten og tjeneren kan lage en felles hemmelighet, kalt Master Secret. Denne brukes av klienten og tjeneren til å generere 'Write MAC Secret', som er sesjonsnøkkelen for hash algoritmen, og 'Write Key' som er sesjonsnøkkelen for kryptering.



Figur 18: TLS handshake tidslinje

Etablering av en sikker sesjon med TLS
TLS Handshake Protokollen involverer følgende steg (Figur 18)

1. Klienten sender en 'klient hello' melding til serveren, sammen med klientens randomverdi og støttede kryptoalgoritmer.
2. Serveren svarer ved å sende en 'server hello' melding til klienten, sammen med serverens randomverdi.
3. Serveren sender sitt sertifikat til klienten for autentisering, og kan eventuelt be om et sertifikat fra klienten
4. Hvis serveren ba om et sertifikat, sender klienten dette
5. Klienten lager en random 'pre-Master Secret' og krypterer den med tjenerens offentlige nøkkel.
6. Serveren mottar 'pre-Master Secret'. Serveren og klienten genererer hver sin Master Secret og sesjonsnøkler basert på 'pre-Master Secret'.
7. Klienten sender en 'change cipher spec' notifikasjon til serveren for å indikere at klienten begynner å bruke den nye sesjonsnøkkelen til å hashe og kryptere meldingene. Klienten sender også en 'client finish' melding.
8. Serveren mottar 'change cipher spec' og bytter sin 'Record Layer Security' tilstand til symetrisk kryptering og bruker den nye sesjonsnøklene. Serveren sender en 'server finished' melding til klienten.
9. Klienten og serveren kan nå utvesle applikasjonsdata over den sikre kanalen de har etablert. Alle meldinger fra klienten til tjeneren og motsatt krypteres med sesjonsnøkkelen.

Gjenopprettelse av en sikker sesjon med TLS

1. Klienten sender en 'Client hello' melding og bruker sesjonsidentifikasjonen til sesjonen som skal gjenopprettes.
2. Serveren kontrollerer sitt buffer for tilsvarende sesjonsidentifikator.
    a. Hvis denne finnes er serveren i stand til å gjenoppta sesjonen og sender en 'Server Hello' melding med den samme sesjonsidentifikatoren.
    b. Hvis denne ikke finnes, genererer serveren en ny sesjonsidentifikator, og klienten og tjeneren gjennomfører en full handshake
3. Klienten og tjeneren må utveksle 'Change cipher spec' medlinger og sende 'Client finished' og 'Server finished' meldinger
4. Klienten og tjeneren kan nå gjenoppta utveksling av applikasjondata over en sikker kanal.

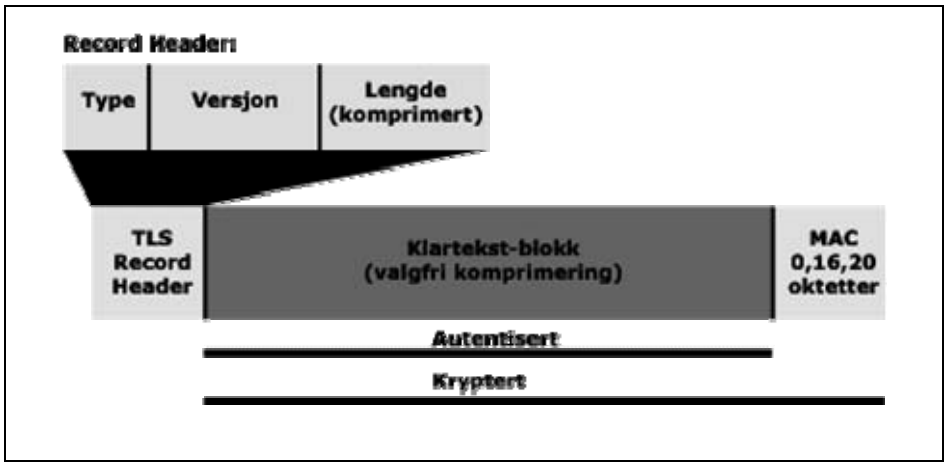
åx

7.1.2 TLS Record Protocol

'TLS'Record Protocol' sørger for å sende og verifisere mottatte applikasjonsdata med riktig sesjonsparametere.

Record Protokollen sikrer applikasjonsdata ved å bruke de nøkler som Handshake protokollen genererte, og de algoritmer det ble enighet om. Record Protokollen er ansvarlig for å sikre applikasjonsdata og verifisere integriteten og opphavet for disse. Den administrerer følgende:

- Deler utgående meldinger i håndterbare blokker, og sette sammen igjen innkomne meldinger.
- Komprimere utgående blokker og dekomprimere innkommende blikker (valgfritt)
- Legger til Message Authentication Code (MAC) for utgående meldinger, og verifiserer innkommende meldinger med MAC.
- Kryptere utgående meldinger (symmetrisk kryptografi) og dekryptere innkommende meldinger (kryptering kan skrus av)



Figur 19: TLS Record

Når Record Protokollen er ferdig, blir de utgående krypterte dataene (Figur 19) videresendt til TCP laget for transport, og de innkomne sammensatte dataene videresendt til applikasjonslaget.

7.2 Sikkerhetsnivå

Protokollag
TLS opererer på transportlaget (egentlig sesjonslaget i OSI referansemodellen, men TCP/IP operere ikke med betegnelsene sesjons- og presentasjonslag). Den sørger for ende-til-ende sikkerhet for trafikk som går over TCP, eller eventuelt andre forbindelsesorienterte transportlag. Sikkerhetsmekanismene dekker hele TCP nyttelasten.

Nøkkelhåndtering
Nøkkelforvaltningen i TLS skjer ved hjelp av en egen handshake protokoll. I denne har man kun mulighet for å autentisere kommunikasjonsmotparten ved hjelp av digitale sertifikater eller Diffie-Helman algoritmen. Sertifikatene må utveksles manuelt på forhånd eller bekreftes ved hjelp av en Public Key Infrastructure (PKI). Den sistnevnte kan gjøre hele nøkkelhåndteringen veldig skalerbar. Selve nøkkelutvekslingen beskyttes med asymmetrisk krytpering.

Dataautentisering
Samme metode som for de andre løsningene (Hashed Message Authentication Code) brukes for dataintegritet og autentisering av data. Tilgjengelige algoritmer er HMAC-MD5 og HMA-SHA1.

Avspillingsbeskyttelse
TLS har ingen egen beskyttelse mot avspilling av meldinger. Den opererer imidlertid over TCP transportprotokollen som sørger for sekvenshåndtering. Denne bør forkaste pakker som har sekvensnummer uten for vinduet sitt, men det kan hende at den istedenfor endrer vindusstørrelse da den søker å få alle pakker igjennom.

Konfidensialitet
TLS Record Protokollen støtter en hel del symmetriske krypteringsalgoritmer. Noen av disse er: Single DES, Triple DES, RC4 og AES. Hele TCP payloaden konfidensialitetsbeskyttes.

Aksesskontroll
TLS har ingen egen mekanisme for aksesskontroll da denne opererer på transportlaget. For vårt scenario er det imidlertid aktuelt å benytte seg av aksesskontrollen til SNMPv1. Denne bruker community-strengen til å identifisere og gi aksesstilgang. Dette er ingen sterk form for aksesskontroll, men muligheten for å differensiere mellom forskjellige communities er i hvert fall der. Skaleringen med communities er veldig grov og det er derfor vanskeligere å identifisere hvem som for eksempel utførte en operasjon (notarisasjon).

Kombinasjoner
Det som er litt spesielt med TLS er at den legger på data autentisering av meldingen før den krypteres. Vi har to kombinasjoner når det gjelder autentisering og kryptering. Vi kan ha begge deler, eller legge på en 'null krypterings algoritme' og benytter dermed bare autentiseringsfunksjonen.

Totalvurdering
Transport Layer Security virker som en kurant sikkerhetsprotokoll, med unntak av at den ikke har noen avspillingsbeskyttelse.

7.3 Konfigurasjon

TLS har ikke blitt konfigurert på grunn av manglende støtte for TLS i relevant kjernenettverksutstyr (rutere). Vi velger likevel å si litt om dette på et generelt grunnlag. Dette vil imidlertid ikke inngå i den endelige vurderingen i kapittel 9.2.

Brukergrensesnitt
På manageren (laptop) er SSL en del av samme pakke som SSH kommer med. Brukergrensenittet er her kommandolinjebasert.

Installasjon
SSL/TLS er veldig lett å installere på en linuxboks, dersom man har tilgang til en rpm pakke. For ruteren var umulig å få aktivert noen TLS støtte, da dette for tiden ikke er implementert i dedikert kjernenettverksutstyr.

Konfigurasjonsparametere.
Teoretisk sett likner løsningene med IPSec og TLS en del på hverandre. For eksempel har begge mulighet for autentisering gjennom en Public Key Infrastructure (PKI), og oppsett av en sikker kanal med asymmetrisk kryptering, og forhandling av sikkerhetsparametere for sesjoner gjennom denne. På et veldig tynt grunnlag antar jeg derfor at antall konfigurasjonsparametere og deres avhengighet er omtrent det samme som for IPSec.

Dokumentasjon
Vi har ikke lett noe videre etter dokumentasjon på TLS og hvordan installere og ta i bruk denne, og har derfor heller ingen forutsetning for å gi noe fornuftige data til dette punktet.

Skalerbarhet
Skalerbarheten til TLS er teoretisk sett veldig god. Det er særlig muligheten for offentlige sertifikater i kombinasjon med PKI som gjør denne løsningen skalerbar.

Vedlikehold
Det er heller ikke så lett å si noe om hvordan det er å drive vedlikehold av en TLS implementasjon. Rent teoretisk kan jeg kanskje påstå at nøkkeloppdatering burde foregå greit automatisk. ’

Oppdatering av programvare skjer enkelt for manageren (laptop), mens for ruteren må dette lastes som en del av ett nytt IOS. Men Cisco sine IOS har uansett ikke støtte for TLS.

7.4 Ytelse

Ytelsestestene i dette kapittelet baserer seg på resultatene fra 'Implementaton and Performance Analysis of SNMP on a TLS/TCP Base [6].

Dette paperet beskriver ytelsestester over UDP, TCP og TLS/TCP for både SNMPv1 og SNMPv3 USM. For løsningene med TLS eller SNMPv3 USM, har ulike sikkerhetsnivåer vært testet. Disse innebefatter dataautentisering med HMAC-MD5 og datakonfidensialitet med 1DES-CBC modus.

	none	auth	authpriv
UDP/SNMPv1	472		
TCP/SNMPv1	523		
UDP/SNMPv3 USM	665	1632	2735
TCP/SNMPv3 USM	881	1990	3634
TCP/TLS/SNMPv3	976	989	1124
TCP/TLS/SNMPv1	774	805	840

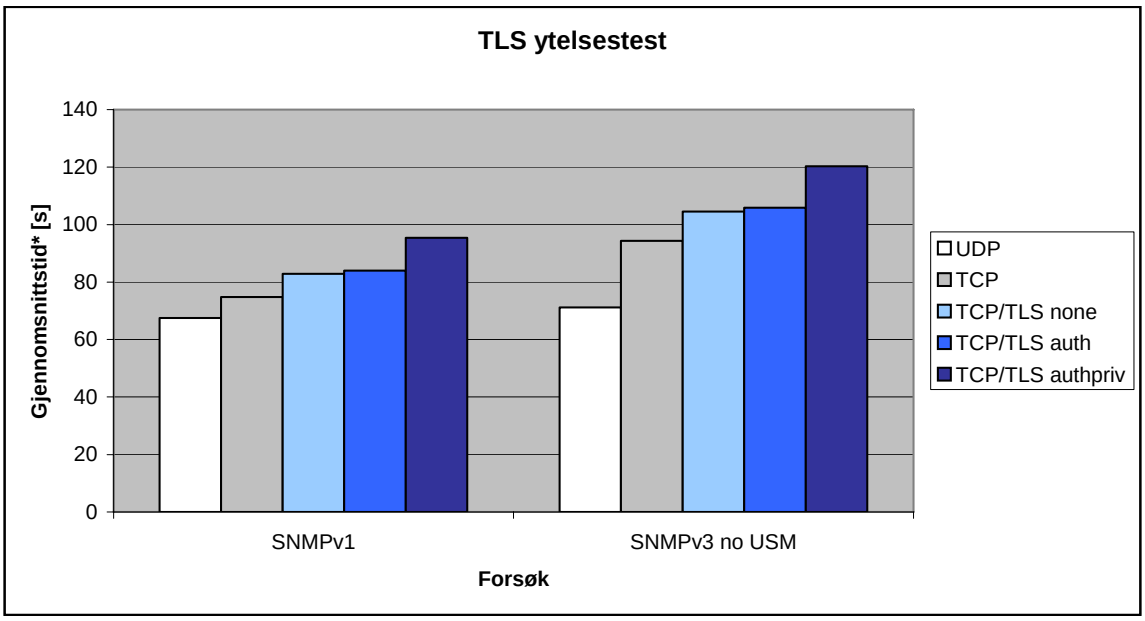
Tabell 3: Resultat fra TLS ytelsestester [6]

Tabell 3 viser ytelsesresultater for tester relevant til vår egen test med get operasjonen. For å ikke foregripe begivenhetenes gang, vil vi her ikke utdype sammenstillingen med SNMPv3 USM, men heller komme tilbake til denne i kapittel 9.3. Vi har imidlertid regnet om de relevante resultatene ved hjelp av samsvarende referanser fra våre egne tall, for å kunne sette opp et sannsynlig og sammenlignbart utfall for vårt testscenario.

	UDP	TCP	TCP/TLS none	TCP/TLS auth	TCP/TLS authpriv
SNMPv1	67,5041	74,7980	82,8636	83,9673	95,4290
SNMPv3 no USM	71,1884	94,3112	104,4810	105,8727	120,3244

Tabell 4: Omregnede resultater for TLS ytelsestest

Tabell 4 viser hvordan et tenkt utfall med TLS ville sett ut for vårt testscenario. Hovedreferansene er SNMPv1 og SNMPv3 USM over UDP. For å få en oversikt over overheaden for selve TLS protokollen, har vi også lagt inn tall for SNMP over bare TCP.



Figur 20: Omregnede resultat for TLS ytelsestest

Figur 20 viser en grafisk fermstilling av de beregnede gjennomsnittsverdiene.

Da disse verdiene er beregnet relativt ut i fra samme faktorer, gir det liten mening å sammenlikne SNMPv1 og SNMPv3 her.

Generelt sett kan vi si at TLS overheaden til kun TCP i vårt tilfelle er på omtrent 10,8%. Videre utgjør dataautentiseringen kun et tillegg på 1,3% når TLS record protokollen først er på plass. For både dataautentisering og konfidensialitet legges det til omtrent 15,2 %, og differansen mellom de to sikkerhetsnivåene er på cirka 13,8 % (relativt til TLS uten aktive sikkerhetsmekanismer).

Tallene må her tas med en stor klype salt, da de er betinget at omregningsfaktorene stemmer. Målingene er jo gjort under litt andre forutsetninger, som for eksempel annen maskinvare, og med færre repetisjoner. Men jeg vil likevel påstå at bildet som dette gir av ytelsen i store trekk stemmer.

8 Andre løsninger

Dette kapittelet tar for seg et par alternative løsninger til SNMP basert management og sikring av disse. Vi gjør her ingen evaluering i henhold til de tre parametrene, men undersøker hvorvidt det finnes støtte for disse løsningene i sentralt nettverksutstyr.

8.1 Secure Shell (SSH)

Secure Shell (SSH) tilbyr sikker 'remote login' og andre sikre nettverkstjenester over et usikkert nettverk, ved å sette opp en kryptert tunnel mellom en klient og tjener. Andre protokoller som telnet, ftp og rlogin har ikke sikret sine forbindelser, og sender f.eks. brukerpassordet i klartekst.

IETF har nedsatt en arbeidsgruppe [83] for å standardisere Secure Shell versjon 2. Arbeidsgruppen definerer en arkitektur som deler protokollen inn i tre hovedelementer:

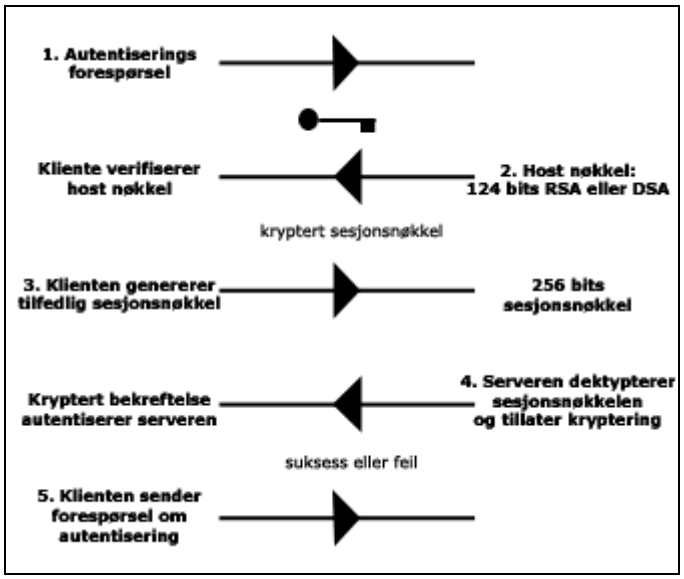
- *Transport Layer Protocol,* som håndterer autentisering av server, konfidensialitet, og integritet med 'perfect forward secrecy'. I tillegg kan den også støtte komprimering. Protokollen vil typisk kjøre over en TCP/IP forbindelse, men kan også kjøre over en hvilken som helst annen pålitelig datastrøm.
- *User Authentication Protocol,* som autentiserer brukeren på klientsiden ovenfor serveren. Denne protokollen kjører over Transport Layer Protocol.
- *Connection Protocol*, som multiplekser den krypterte tunnelen i flere logiske kanaler. Denne kjører over User Authenticaton Protocol.

Vi vil kun ta en kort titt på den viktigste.

8.1.1 Transport Layer Protocol

Transport Layer Protocol tar seg av behandlingen av data ut til og inn fra TCP-laget. Den autentiserer serveren, krypterer og komprimerer dataene, og tillegger/verifiserer MAC for dataintegritet og autentisering av dataopprinnelse.

Figur 21 illustrerer hvordan en klient autentiserer en tjener (server) i SSHv2.



Figur 21: SSH server autentisering

1. Klienten oppretter en forbindelse med serveren og ber om autentisering.
2. Serveren identifiserer seg selv med sin offentlige nøkkel, som er en 1024 bits RSA eller DSA nøkkel (offentlig nøkkel algoritmer – passordautentisering kan også brukes). Klienten slår opp i sin lokale database for å kontrollere om den offentlige nøkkelen er kjent. Ved førstegangsregistrering lagres denne her. Hvis nøkkelen forandres for en host, alarmeres klienten. På denne måten unngås 'public key substitution'.
3. Klienten genererer så ett 256-bits tilfeldig tall og velger en krypterings algoritme (f.eks. 3DES). Tallet blir så kodet med enten RSA eler DSA. En ren RSA eller DSA autentisering stoler aldri på noe annet enn en privat nøkkel. Den kodede nøkkelen sendes til serveren. Host nøkkelen sikrer autentiseringen av den spesifikke serveren.
4. Serveren dekoder RSA eller DSA krypteringen og rekonstruerer sesjonsnøkkelen. Videre sender serveren en bekreftelser til klienten med den kodede sesjonsnøkkelen. Dette autentiserer også serveren. Resten av sesjonen krypteres med symmetrisk krypto.
5. Klienten kan så sende en forespørsel om autentisering seg selv. Serveren svarer med suksess eller feil.

For den videre symmetriske krypteringen brukes den spesifiserte kryptoalgoritmen. Noen av de tilgjengelige algoritmene er: DES, 3DES Blowfish, Twofish, Arcfour, CAST128-CBC, 128 og 256 bit AES. For dataintegritet og autentisering av avsender brukes MAC med en av hash algoritmene MD5 og SHA1. Sesjonsnøklene for alle disse algoritmene kan f.eks. skiftes hver time.

SSH er en fleksibel protokoll. Ved å bruke muligheten med 'port forwarder' i TCP/IP, kan enhver applikasjon med et statisk tildelt portnummer krypteres, inklusive for SNMP (men det utdyper vi ikke her).

8.1.2 Støtte i relevant utstyr

SSHv1 har lenge vært de-facto standarden for krypterte terminalforbindelser og sikker filoverføring. Med definisjonen av SSHv2 som IETF standard, vil SSH styrke sin posisjon ytterligere. Det finnes i dag en rekke SSH klienter/tjenere for ulike operativsystemer.

Vår oppgave var å finne ut i hvilken grad det finnes, eller er mulig å implementere, støtte for SSH i eksisterende nettverksenheter. Vi har undersøkt dette for det utstyret vi brukte i vårt testoppsett. Her støttet ruteren SSHv1 [56]. Cisco Norge kunne opplyse oss om at alle deres IOS-er med kryptografisk støtte, også støtter SSHv1. Videre fortalte de at støtte for SSHv2 i all ny tilsvarende programvare er ventet i løpet av sommeren 2003.

8.2 Hypertext Transfer Protocol Secure (HTTPS)

Det er ventet at videre steg i praktisk implementasjon av nettverksmanagement vil være web-basert [10]. Med definisjonen av språkene SGML og XML som er supersett av HTML, blir det lettere og lettere å spesifisere objektdata for web basert management. Overføring av dette vil foregå over HTTP protokollen [84].

8.2.1 HTTP over TLS

Den mest vanlige måten å sikre HTTP trafikk på er å bruke TLS. Hvordan dette gjøres har TLS arbeidsgruppen spesifisert i [85]. Vi vil ikke gå nærmere inn på dette her, da kapittel 7 allerede har beskrevet og evaluert Transport Layer Security (TLS), dog for SNMP trafikk sikkerhet.

8.2.2 Støtte i relevant utstyr

Pr. i dag støtter flere nettverksenheter web-basert management (i det minste overvåkning av objektverdier). Dette inkluderer ruteren som inngår i vårt testoppsett.

Hvorvidt det er mulig å sikre HTTP basert management trafikk ved hjelp av SSL er imidlertid ikke verifisert for testoppsettet.

Det er imidlertid sannsynlig å anta at dette etter hvert vil bli en meget vanlig måte å drive management av nettverk på, og dermed også sikring ved hjelp av denne metoden.

9 Diskusjon

I dette kapittelet diskuterer vi resultatene for de tre løsningene i kapittel 5, 6 og 7, for hver av evalueringsparametrene spesifisert i kapittel 4.2. Til sist gjøres en totalvurdering med parametrene i betraktning.

9.1 Sikkerhetsnivå

Vi vil her sammenlikne sikkerhetsnivået for de tre løsningene i henhold til kriteriene i kapittel 4.2.1. Vi tar for oss de enkelte underpunktene først, og trekker etterpå en generell slutning.

Protokollag
Protokollaget løsningene opererer på er en av hovedforskjellene mellom dem. Alle løsningene tilbyr ende-til-ende sikkerhet for management trafikken, men dekker forskjellige utsnitt av meldingene med sine sikkerhetstjenester. Dette er relevant med tanke på hva som kan avsløres av avlytting av kommunikasjonen.

Internet Protocol Security (IPSec) er den løsningen som med tanke på protokollag gir det beste sikkerhetsnivået. Dette fordi den beskytter størst del av den overførte meldingen, og er overlegen når det gjelder skalerbarhet. Dernest burde Transport Layer Security (TLS) og så SNMPv3 Security komme, men vi velger å rangere disse to motsatt. Dette på grunn av SNMPv3 VACM som gir mulighet for aksesskontroll, og at TLS er avhengig å kjøre over TCP.

Så når det gjelder protokollag anbefaler vi å bruke nettverkslaget, med mindre en applikasjonsspesifikk løsning med mulighet for aksesskontroll er sterkt ønskelig.

Nøkkelhåndtering
All nøkkelhåndtering foregår egentlig på applikasjonslaget. SNMPv3 USM inneholder egne mekanismer for nøkkelhåndtering. IPSec bruker Internet Security Asossiation and Key Management Protocol, og TLS bruker en Handshake Protocol.

SNMPv3 USM har den ulempen at nøklene må legges inn på forhånd, mens de to andre kan benytte seg av offentlig nøkkelinfrastruktur. Det sistnevnte gjør systemet mer skalerbart og lettere å vedlikeholde. Den største trussel mot et slikt system er tilgjengeligheten av 'Certificate Authorities' som identifiserer partene i forkant av forhandling av sesjonsnøkler. Dette kan bedres ved å ha to redundante 'Certificate Authorities'. Ved bortfall av disse kan etablerte forbindelser likevel fortsette kommunikasjonen, da CA-ene kun benyttes for autentisering ved oppsett av nye forbindelser. De entiteter som har lokal kopi av hverandres offentlige nøkler kan også opprette ny kommunikasjon. Forhandling av sesjonsnøklene foregår for både IPSec og TLS med asymmetrisk kryptering. Asymmetrisk kryptering innebærer en tyngre matematisk beregning enn for symmetrisk kryptering. Dette gjør det vanskeligere finne nøkkelen for asymmetrisk kryptering i forhold til symmetrisk kryptering, dersom nøkkellengden ellers er den samme.

Hvordan nøklene lages har også betydning for hvor lette disse er å gjette. For SNMPv3 USM er det mye lettere å gjette (brute-force eller ordliste angrep) nøklene da disse lages fra et mye mindre utfallsrom (brukerpassord) enn helt tilfeldig genererte nøkler. Dette er kanskje den største svakheten med SNMPv3 USM.

Alle løsningene har gode metoder for nøkkeloppdatering. Likevel er det en liten forskjell i at IPSec har mulighet for å derivere nye helt uavhengige sesjonsnøkler (Perfect Forward Secrecy), mens de andre løsningene utleder sesjonsnøklene fra de forutgående. Det sistnevnte er en ulempe, fordi straks en nøkkel er kompromittert er det mye lettere å nøste opp verdien av de påfølgende sesjonsnøklene.

Dataautentisering
Dataintegritet og autentisering av dataopprinnelse foregår for alle løsningene ved hjelp av en Hashed Message Authentication Code (HMAC). De tilgjengelige hash algoritmene er HMAC-MD5 som bruker en 128-bits nøkkel, HMAC-SHA1 som bruker en 160-bits nøkkel og HMAC-RIPEMD (kun for IPSec) som også bruker en 160-bits nøkkel.

Styrken på de forskjellige algoritmene er blant annet gitt av nøkkellengden. I tillegg er HMAC-RIPEMD litt raskere enn SHA1 [86]. Alle hash funksjoner har en svakhet i form av at man kan få samme MAC fra flere meldinger (dette kalles collision) [87], men dette har ingen konsekvenser når de brukes i en enveisfunksjon som HMAC [88 89].

Sikkerhetsnivået på dataautentiseringen er den samme for de feltene av meldingen som beskyttes i hver løsning. Forskjellen ligger da i hvilke deler av meldingene som dekkes, og her gir lavest mulig lag best dekning.

For vårt testoppsett valgte vi å bruke HMAC-MD5, da dette ble brukt i tidligere tester som vi ønsket å sammenlikne med.

Avspillingsbeskyttelse
SNMPv3 USM og IPSec har egne funksjoner for beskyttelse mot avspilling. TLS har ikke dette og stiller i så måte svakt på dette punktet, selv om den har flytkontroll på TCP laget. De to løsningene for SNMPv3 USM og IPSec er henholdsvis tidsstempel og sekvensnummer. Begge disse løsningene er tellere og har ingen signifikante forskjeller i forhold til hverandre.

Datakonfidensialitet
Datakonfidensialitet foregår for alle løsningene ved hjelp av symmetrisk kryptering. Vi vil her ikke gå algoritmene veldig nøye i sømmene, men bruke nøkkellengden som den viktigste indikatoren.

SNMPv3 USM har i motsetning til IPSec og TLS, få algoritmer å velge i mellom. SNMPv3 USM har foreløpig kun støtte for Single DES som kun har 56-bits nøkkel, men mye tyder på at den etter hvert også vil få støtte for AES. Single DES begynner å bli noe svak, da det etter hvert har blitt mulig å finne den 56-bits korte nøkkelen [90, 91].

Vanlige algoritmer som søttes av IPSec eller TLS er: CAST-128, RC2, RC4-128, RC5-128, IDEA-128, BLOWFISH-128, DES-56, 3DES-192, AES-128 og AES-256 (kun IPSec foreløpig). I tillegg støtter de også NULL-kryptering, som gir mulighet for å kjøre kun autentisering med de respektive protokoller.

Pr i dag har altså IPSec og TLS de beste konfidensialitetsløsningene. Det er ikke noen store forskjeller disse i mellom, bare at de opererer på forskjellige lag og dekker ulik størrelse av meldingen. SNMPv3 USM vil forhåentligvis etter hvert kunne måle seg med dem når den får støtte for AES.

Aksesskontroll
Egen aksesskontroll er kanskje den enkeltmekanismen som peker mest i SNMPv3 sin favør. Ingen av de andre løsningene har mulighet for å kjøre noen aksesskontroll da de rett og slett befinner seg på feil lag. Disse løsningene er derfor prisgitt SNMPv1 sin heller noe dårlige aksesskontroll. Det er stor forskjell på differensieringsnivå på communities og brukere. Så hvis det er et mål å kjøre en stram aksesskontroll er SNMPv3 absolutt løsningen. Hvis ikke, går SNMPv1 aksesskontroll an, forutsatt at community-strengen konfidensialitetsbeskyttes av andre mekanismer.

Kombinasjoner
Ulike kombinasjoner av sikkerhetsnivå for de forskjellige løsningene er mange. Vi søkte kun å utforske tre nivåer for hver løsning. Dette var ingen sikkerhetstjenester, dataautentisering og kryptering med dataautentisering. For SNMPv3 USM var nettopp disse sikkerhetsnivåene definert (og derfor også utgangspunktet for vurderingen). For både TLS og IPSec ESP, var begge de to siste nivåene realiserbare. Men for IPSec ESP var det ikke mulig å legge inn tilleggsheadere uten noen sikkerhetstjenester (første sikkerhetsnivå), kun nullkryptering med dataautentisering. TLS kunne imidlertid ha både nullkryptering og samtidig droppe valgfri autentisering.

Totalvurdering
Som en totalvurdering for sikkerhetsnivå vil jeg påstå at Internet Protocol Security (IPSec) er den beste løsningen. Dette fordi den har tilfredstillende løsninger på alle kriterieområdene. Den dekker størst del av meldingsfelt med konfidensialitet og integritet. Videre har den sikre og skalerbare metoder for autentisering, nøkkelgenerering og nøkkelutveksling. Til sist har den sterke kryptoalgoritmer både for dataautentisering og –konfidensialitet.

De to andre løsningene måler seg med IPSec på mange områder, men har noen mangler som svekker den totale sikkerheten ved løsningene. Transport Layer Security (TLS) mangler avspillingsbeskyttelse, og SNMPv3 User-based Security Module (USM) velger i praksis nøkler fra et mye mindre utfallsrom enn tilfeldig genererte nøkler ville gjort, og har foreløpig en svak krypteringsbeskyttelse. Men som en styrke har den en mulighet for en stram aksesskontroll ved hjelp av VACM. Av de to løsningene, vil jeg påstå at mangel på avspillingsbeskyttelse er den mest alvorlige mangelen. Dette fordi dette gi aksess ved avspilling av autentiseringssekvens, som er en mye lettere og mer sannsynlig operasjon enn for eksempel brute forcing av brukerpassord eller forsøk på kryptoanalyse.

9.2 Konfigurasjon

Vi vil her sammenlikne enkelheten av konfigurasjonen for de tre løsningene i henhold til kriteriene i kapittel 4.2.2. Vi tar for oss de enkelte underpunktene først, og trekker etterpå en generell slutning.

Brukergrensesnitt
Brukergrensenittet for både SNMPv3 og IPSec er ganske likt med tilsvarende forskjeller mellom manager (laptop) og agent (ruter). Dette fordi brukergrensesnittet hovedsakelig er styrt av enheten og sekundert av implementasjonen som kjøres.

Ruteren brukte altså et kommandolinjebasert brukergrensesnitt, mens laptoppen i tillegg bruker konfigurasjonsfiler. En liten forskjell mellom IPSec og SNMP i manageren, var at det for IPSec måtte gjøres konfigurering i kommandolinje for å sette opp og ned sikkerhetsassosiasjoner definert i konfigurasjonsfilene, mens SNMP sendte med brukerparametere og passord (som også kan stå i konfigurasjonsfiler) direkte med utførselskommandoene.

Installasjon
Installasjonen av SNMP, IPSec og TLS/SSL i manageren hadde mange fellestrekk. Alle kunne installeres fra rpm-er, men SNMP og SSL var allerede installert som standard med operativsystemet. IPSec måtte legges inn separat, og dette var en stor og krevende jobb. Dette var særlig fordi Linux-kernelen måtte bygges på nytt for å få nullkryptering og 1des. I tillegg måtte en programfeil i IPSec implementasjonen identifiseres og fikses. Sånn sett var nok SNMP enklest å installere, og IPSec mer komplekst. TLS ble ikke forsøkt installert for systemet.

Konfigurasjonsparametere
Hovedforskjellen her er at SNMP parametrene hovedsakelig konfigureres på agenten, mens IPSec parametrene konfigureres fult ut i begge ender for å kunne sette opp en ISAKMP SA.

Min erfaring er at det til å begynne med var vanskeligst å forstå sammenhengen mellom view, aksessliste, gruppe og bruker for SNMPv3, men dette kan jo ha sammenheng med manglende forkunnskaper om SNMPv3 i forhold til IPSec. Så noe av kritikken om at SNMPv3 er kompleks å konfigurere kan muligens komme av at den er relativt ny og ukjent for mange. Videre var det lett å legge inn brukernavn og passord i kommandolinje eller konfigurasjonsfil.

Det tok også tid å få oversikt over IPSec parametrene, men her hadde jeg som sagt også mer forkunnskaper. Det at IPSec først må forhandle frem en sikkerhetsassosiasjon eller sette alle parametere manuelt, gjør nok at denne egentlig er mest kompleks. Her var manuelt oppsett mest utfordrende, da alle avhengigheter må settes helt samsvarende, istedenfor at en protokoll forhandler frem en felles enighet. Oppbygningen på ruteren med crypto maps og transform-sets var en liten terskel å komme over, men etter hvert kom dette på plass. Dette var mer oversiktlig på laptoppen hvor all konfigurasjonen var samlet i en fil.

Dokumentasjon
Begge løsningene var tidkrevende å sette seg inn i og krevde både teoribakgrunn og brukerdokumentasjon.

IPSec var den løsningen som etter mitt skjønn var best dokumentert. Denne hadde flest relevante dokumenter. Dette ha nok sammenheng med at IPSec sikrer alt som går over IP og dermed brukes i mye større omfang. Cisco Systems hadde imidlertid dekkende dokumentasjon av begge løsninger. FreeS/WAN hadde litt dokumentasjon, i forhold til NET-SNMP som hadde nesten ingen ting om SNMPv3.

Skalerbarhet
Når det gjelder skalerbarhet, er IPSec med automatisk nøkkelhåndtering og 'Public Key Infrastruktur' (PKI) det beste alternativet. Dette fordi man da automatisk kan forhandle en sikkerhetsassosiasjon med entiteter som en ikke har hatt noe kommunikasjon med før. Denne fordelen gjelder sannsynligvis også for TLS. Generelt sett er IPSec skalerbart også fordi den kan brukes som sikkerhetsløsning for alt annet som kjører over IP protokollen. SNMPv3 USM er mer begrenset i form av at den er avhengig at brukeren på forhånd legger inn sitt brukernavn og passord på de motorene den skal kommunisere med.

Vedlikehold
For både IPSec, TLS og SNMPv3 USM, foregår vedlikeholdet av ferske nøkkelsett automatisk. IPSec og TLS håndterer dette ved hjelp av forhandling gjennom egne sikre assosiasjoner, mens SNMPv3 USM forhandler ny nøkler ut i fra den gamle sesjonsnøkkelen i sammen med den pågående kommunikasjonen.

Når det gjelder oppdatering av programvaren varierer dette mest med enhetene, ikke løsning. Ruteren krever lasting og omstart av hele operativsystemet, mens laptoppen kan oppdatere pakkene omtrent mens de kjører, og samtidig beholde konfigurasjonsfilene. Men dette gjelder ikke for super-freeswan som må bli lagt inn i kernelen.

Totalvurdering
Den desidert mest kompleks løsningen å installere var IPSec super-freeswan 1.99.7.1foo, som måtte inn i Linux-kjernen og i tillegg hadde en kodefeil i et viktig script.

Men dersom vi ser bort i fra installering, var SNMPv3 mest vanskelig å konfigurere. Utfordringen lå hovedsakelig i å bli kjent med kommandolinje i ruteren, og forstå oppbygningen av view, brukere, grupper, og aksessrettigheter. I tillegg var det også generelt sett utfordrende å finne kommandoer (dokumentasjon) for net-snmp.

Etter installasjonen, var IPSec mer rett frem å konfigurere. Det tok imidlertid også her litt tid å bli kjent i kommandolinje på ruteren, og generelt sett strukturen med aksess-lister, transform-sets, crypto maps og isakmp policy.

TLS var ikke med i denne vurderingen.

9.3 Ytelse

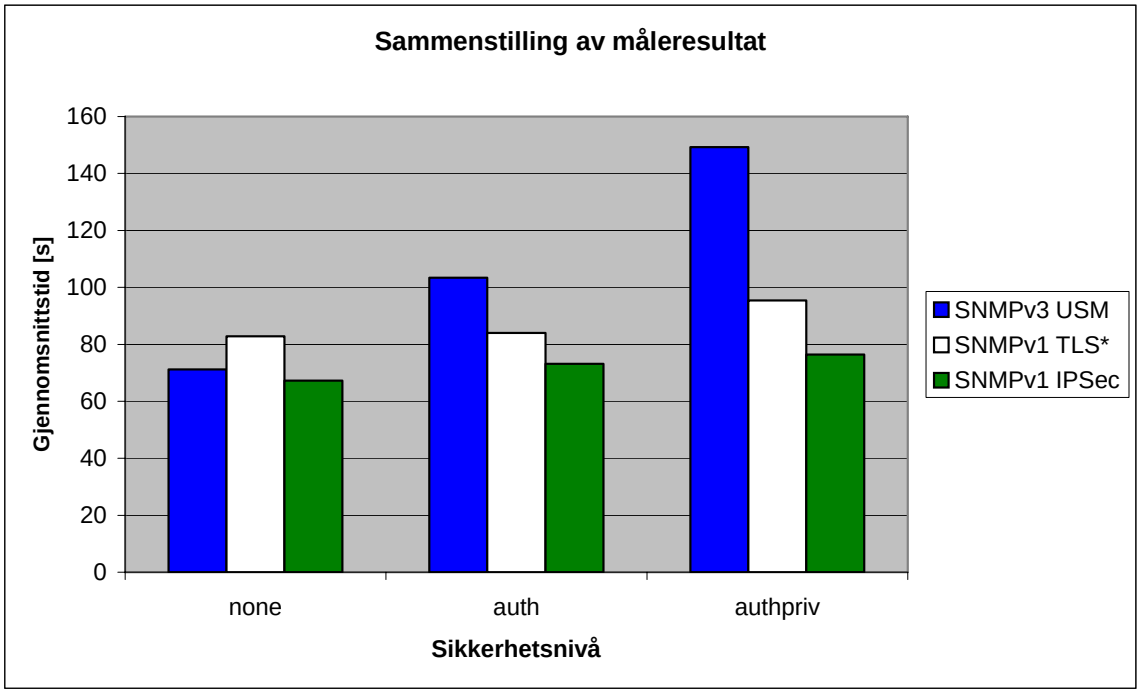
Vi vil her sammenlikne ytelsestestene for de tre løsningene i henhold til definisjonen i kapittel 4.2.3. Målet her er ganske enkelt å vurdere hvilken løsning som er raskest for de forskjellige sikkerhetsnivåene og total.

	none	auth	authpriv
SNMPv3 USM	71,23686	103,356	149,2332
SNMPv1 TLS*	82,8636	83,9673	95,4290
SNMPv1 IPSec	67,25125	73,09246	76,42104

Tabell 5: Sammenstilling av resultater fra ytelsestestene

Tabell 5 viser tallmaterialet for 'Figur 22: Sammenstilling av resultater fra ytelsestestene'. For andre løsninger enn SNMPv3 USM, valgte vi å sammenlikne med SNMPv1, da denne er kjappest.

Det bør noteres at SNMPv1 IPSec sikkerhetsnivå 'none' er gjennomsnittstiden for en vanlig SNMPv1 pakke uten IPSec. Videre må det nevnes at SNMPv1 TLS* rekken er estimert omtrentlig ut i fra tidligere kjente testresultater.



Figur 22: Sammenstilling av resultater fra ytelsestestene

Figur 22 viser en sammenstilling av måleresultatene for de tre løsningene for tre sikkerhetsnivå. Vi ser at SNMPv3 USM er den desidert tregeste løsningen. For intet sikkerhetsnivå er den omtrent bare 5% senere enn SNMPv1, mens med autentisering, er den 41% tregere. Når vi i tillegg legger til konfidensialitet, blir den hele 95% tregere enn uten sikkerhetsløsninger. Videre er SNMPv1 TLS* raskere enn SNMPv3 USM, med unntak av for ingen sikkerhetsnivå. Det sistnevnte skyldes at TLS uansett nivå må prosesseres gjennom samme system, bare uten at kryptering og autentisering blir satt i kraft. Forskjellen fra SNMPv1 IPSec er en økning på henholdsvis 23,2% 14,9% og 24,9% for de tre sikkerhetsnivåene til SNMPv1 TLS

Vi kan altså konkludere denne evalueringsparameteren med at IPSec er den desidert raskeste løsningen, så kommer SNMPv1 TLS ikke langt etter, og til sist SNMPv3 USM et godt stykke bak. Dette gjelder for alle sikkerhetsnivåer, unntatt når vi ikke har noen sikkerhetsløsning hvor ren SNMPv1 er den raskeste løsningen og SNMPv3 kommer rett etter.

9.4 Totalvurdering

Vi har i de forutgående underkapitlene funnet frem til hvilken løsning som er best innefor hvert av de tre områdene vi bruker i evalueringen. Vi vil nå gjøre en sammenstilling av disse resultatene for å komme med en totalvurdering av løsningene. Men først en liten oppsummering av resultatene fra de tre løsningene:

Sikkerhetsnivå:

- IPSec gir det beste sikkerhetsnivået da den har kvalitativt sterke sikkerhetsmekanismer for alle relevante tjenester.
- SNMPv3 USM gir det nest beste sikkerhetsnivået. Hovedstyrken ved denne er at den har mulighet for en stram aksesskontroll, som i en del tilfeller vil være overskyggende. Hovedsvakheten med løsningen er at den hadde et lite faktisk utfallsrom ved bestemming av brukerpassord, og foreløpig en svak konfidensialitetsalgoritme.
- TLS hadde det tredje beste sikkerhetsnivået av de tre. Løsningen fikk laveste plassering hovedsakelig på grunn av dårlig avspillingsbeskyttelse.

Konfigurasjon:

- SNMPv3 var totalt sett den letteste løsningen å installere og konfigurere. Selve konfigurasjonen var litt mer omfattende enn for IPSec. Dette fordi den hadde større avhengigheter mellom view, brukere, grupper og aksessrettigheter, og at dette innebar en litt ny tenkemåte iverfall for forfatteren. I tillegg var net-snmp dårligere dokumentert en freeswan-ipsec.
- IPSec ble vurdert som vanskeligst å konfigurere på grunn av en kjempekrøkkete installasjonsprosess og programmeringsfeil i scripts i den brukte implementasjonen. Men dersom man ser bort fra selve installasjonen, var selve konfigureringen enklere enn for SNMPv3.
- TLS var ikke med i vurderingen.

Ytelse:

- IPSec er den raskeste løsningen for alle de definerte sikkerhetsnivåene.
- TLS er den nest raskeste løsningen med unntak av for laveste sikkerhetsnivå.
- SNMPv3 USM er den tregeste løsningen med unntak av for laveste sikkerhetsnivå.

Dersom vi må prioritere de tre evalueringsparameterne opp mot hverandre, vil følgende oppsett være naturlig:

1. Sikkerhetsnivå har førsteprioritet, da dette er selve hensikten med oppgaven.
2. Ytelse har andreprioritet, da dette er vesentlig for hvordan løsningen fungerer i praksis. En management løsning er helt avhengig av å kunne arbeide i høyt tempo, og holde administrator løpende oppdatert på alle ønskede områder.
3. Hvor enkelt det er å konfigurere løsningen er også viktig, men dette berører i første rekke kun administratoren(e) som tar seg av konfigureringen. Inntill et vist nivå hvor det ikke lenger er forsvarlig eller mulig å bruke tid på konfigurering, er sikkerhetsnivå og ytelse mer vesentlig.

Med utgangspunkt i prioriteringen av evalueringsparemeterene, og deres resultater innad, kan vi trekke følgende overordnede slutninger:

1. IPSec gir de generelt beste resultatene på evalueringsområdene for denne hovedoppgaven. Den er både sikrest, raskest. Selve konfigurasjonen er også egentlig enklest, men den har fått minus på grunn av en vrien installasjonsprosess.
2. SNMPv3 USM er den nest beste løsningen. Denne er enklest på konfigurering til tross for at den bygger på litt uvante konsepter. Den har det nest best sikkerhetsnivå og en unik mulighet for aksesskontroll (VACM). Ytelsen er dessverre den dårligste av løsningene og den kanskje største ulempen med SNMPv3.
3. TLS er den tredje beste løsningen, da denne har tredje beste sikkerhetsnivå, nest beste ytelse og ikke er med i vurderingen av konfigurasjonen.

10 Konklusjon

Internet Protocol Security (IPSec) ser ut til å være den mest effektive løsningen for å sikre IP-basert managementtrafikk. Noen av argumentene for dette er god ytelse, gode og skalerbare nøkkelhåndteringsmekanismer, bra avspillingsbeskyttelse, og sterk konfidensialitet og dataautentisering.

Dersom det er essensielt med stram aksesskontroll, vil vi alternativt anbefale å bruke SNMPv3. SNMPv3 har gjennom sin View-based Access Control Module (VACM) i mye større grad mulighet for å differensiere aksesskontroll, enn for eksempel SNMPv1 sin community-baserte tilnærming. Ytelsen vil imidlertid gå betraktelig ned på grunn av at SNMPv3 User-based Security Module (USM) er mye tregere enn IPSec.

Det er et mål å få flest mulig organisasjoner til å gå bort fra å bruke usikret SNMP for managementoperasjoner. Vårt råd om å bruke Internet Protocol Security (IPSec) for å sikre managementtrafikk, muliggjør sikker bruk av kritiske managementoperasjoner for kontrollering av forvaltede ressurser uten å måtte oppgradere til SNMPv3. Vi har vurdert IPSec som en bedre løsning enn SNMPv3. Dette er gode nyheter for veldig mange managere som av forskjellige årsaker ikke ønsker å oppgradere til SNMPv3. Forhåpentligvis vil våre betraktninger bidra til en liten økning i den generelle sikkerheten for IP-basert management trafikk.

Det vil videre kunne være interessant å studere flere implementasjoner av SNMP og IPSec, og i tillegg kjøre tester mot nettverksenheter (arbeidsstasjoner) som kan prosessere SNMP over TLS. Særlig vil det være interessant å undersøke hvordan ytelsen for IPSec (og eventuelt TLS) blir ved uthenting av enkeltvariable fra flere entiteter, og hvilke konsekvenser dette vil få for overhead med tanke oppsetting av nye ISAKMP sikkerhetsassosiasjoner og reforhandling av sesjonsnøkler.

Fremtidige IP-baserte managementsystemer vil på mellomlang sikt mest sannsynlig være web-basert og XML over HTTPS er den mest åpenbare tilnærming. De svakheter som vi har sett på for TLS vil imidlertid være like relevante enten denne beskytter HTTP eller SNMP trafikk.

Litteraturliste

Litteraturliste

[1] W. Stallings *'SNMP, SNMPv2, SNMPv3, and RMON 1 and 2'* - 3rd ed. Addison-Wesley, 1999.

[2] International Standardization Organization (ISO), 'Information technology - Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture'. ISO/IEC 7498-2:1989

[3] W. Stallings. SNMP and SNMPv2: The infrastructure for network management. IEEE Communications Magazine, 36(3):37-43, mars 1998.
http://telecom.tlab.ch/~studer/abta/V7b-03122001.pdf

[4] W. Stallings. SNMPv3: A Security Enhancement for SNMP. IEEE Communications Surveys, Fjerde kvartal 1998 Vol. 1 No. 1
http://www.comsoc.org/livepubs/surveys/public/4q98issue/stallings.html

[5] R. Windvik, *'IFIP/IEEE International Symposium on Integrated Network Management, Seattle, mai 2001*' Reiserapport 04338, Forsvarets Forskningsinstitutt, september 2001.

[6] X. Du, M. Rozenblit and M.A. Shayman, *'Implementation and Performance Analysis of SNMP on a TLS/TCP Base'* IFIP/IEEE International Symposium on Integrated Network Management, Seattle, Washington, mai 2001.
http://www.ee.umd.edu/~shayman/papers.d/snmp.pdf

[7] B. Sivasubramanian, M.K. Sundareshan, *'Management of end-to-end Security in Collaborative IP Network Environments'* IFIP/IEEE International Symposium on Integrated Network Management, Seattle, Washington, mai 2001.

[8] International Standardization Organization (ISO), 'Information technology - Open Systems Interconnection – Basic Reference Model – Part 4: Management Framework'. ISO/IEC 7498-4:1989

[9] P. Spilling 'Fra ARPANET til internett – En utvikling sett med norske øyne', Forskningsnotat TF N 32/95, Telenor FoU, august 1995
http://www.isoc-no.no/isoc-no/social/arpa-no.html

[10] R. Boutaba and J. Xiao '*Network Management: State of the Art*' IFIP World Computer Congress 2002, Montreal 2002
http://www.ifip.tu-graz.ac.at/TC6/events/WCC/WCC2002/papers/Boutaba.pdf

[11] R. H. Glitho and S. Hayes 'Telecommunications Management Network: visions vs. reality' IEEE Communications Magazine, mars 1995 (side 47-52)
http://www.comsoc.org/livepubs/surveys/public/2q99issue/Glitho.pdf

[12] J. Thompson 'Web-based Enterprise Management Architecture', IEEE Communications Magazine, March 1998.

[13] A. Bieszczad et al 'Mobile Agents for Network Management' IEEE Communications Surveys, Fjerde kvartal 1998 Vol.1 No 1 (side 2-9)
http://www.comsoc.org/livepubs/surveys/public/4q98issue/pdf/Bieszczad.pdf

[14] M. Baldi, G. Picco 'Evaluating the Tradeoffs of Mobile Code Design Paradigms in Network management Applications', 1998

[15] M. Wooldridge, N. R. Jennings 'Intelligent Agents: Theory and Practice' The Knowledge Engineering Review. Vol. 10, No.2, 1995.

[16] K. Psounis 'Active Networks: Applications, Security, Safety, and Architectures' IEEE Communiccations Survays, fjerde kvartal 1999 (side 2-16)
http://www.comsoc.org/livepubs/surveys/public/1q99issue/pdf/Psounis.pdf

[17] R. Boutaba, and A. Polyrakis 'Pro*jecting Advanced Enterprise Network and Service Management to Active Networks*' IEEE Network Magazine Vol.16, No.1 (p28-33) januar 2002.

[18] J Davin et al. '*A Simple Gateway Monitoring Protocol*' IETF Network Working Group RFC 1028, november 1987
http://www.ietf.org/rfc/rfc1028.txt

[19] 'Simple Network Management Protocol (snmp) Charter', IETF Network Working Group, avsluttet november 1991
http://www.ietf.org/html.charters/OLD/snmp-charter.html

[20] M. Rose and K. McCloghrie *'Structure and Identification of Management Information for TCP/IP-based Internets*', IETF Network Working Group RFC 1155, mai 1990
http://www.ietf.org/rfc/rfc1155.txt

[21] M. Rose and K. McCloghrie '*Concise MIB definitions'*, IETF Network Working Group RFC 1156, mai 1990
http://www.ietf.org/rfc/rfc1212.txt

[22] J. Case et al. *'A Simple Network Management Protocol (SNMP)*', IETF Network Working Group RFC 1157, mai 1990
http://www.ietf.org/rfc/rfc1157.txt

[23] M.Rose 'Manage*ment Information Base for Network Management of TCP/IP-based internets: MIB-II*' IETF Network Working Group RFC 1158, mai 1991
http://www.ietf.org/rfc/rfc1158.txt

[24] Remote Network Monitoring (rmonmib) Charter, IETF Network Working Group, Active
http://www.ietf.org/html.charters/rmonmib-charter.html

[25] S. Waldbusser ' Remote Network Monitoring Management Information Base' IETF Network Working Group RFC 1271, november 1991
http://www.ietf.org/rfc/rfc1271.txt

[26] Secure SNMP (snmpsec) Charter, IETF Network Working Group, Concluded mai 1993
http://www.ietf.org/html.charters/OLD/snmpsec-charter.html

[27] J. Davin et al. 'SNMP Administrative Model' IETF Network Working Group RFC1351, July 1992.
http://www.ietf.org/rfc/rfc1351.txt

[28] J. Galvin et al. 'SNMP Security Protocols' IETF Network Working Group RFC1352, July 1992.
http://www.ietf.org/rfc/rfc1352.txt

[29] K. McCloghrie et al. ' Definitions of Managed Objects for Administration of SNMP Parties' IETF Network Working Group RFC 1353, juli 1992
http://www.ietf.org/rfc/rfc1353.txt

[30] SNMP Version 2 (snmpv2) Charter, IETF Network Working Group, avsluttet desember 1995
http://www.ietf.org/html.charters/OLD/snmpv2-charter.html

[31] J. Case et al. *'Introduction to Community-based SNMPv2*' IETF Network Working Group RFC 1901, januar 1996
http://www.ietf.org/rfc/rfc1901.txt

[32] J. Case et al. 'Struc*ture of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)*' IETF Network Working Group RFC 1902, januar 1996
http://www.ietf.org/rfc/rfc1902.txt

[33] J. Case et al. *'Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)*' IETF Network Working Group RFC 1903, januar 1996
http://www.ietf.org/rfc/rfc1903.txt

[34] J. Case et al. *'Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)*' IETF Network Working Group RFC 1904, januar 1996
http://www.ietf.org/rfc/rfc1904.txt

[35] J. Case et al. *'Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)*' IETF Network Working Group RFC 1905, januar 1996
http://www.ietf.org/rfc/rfc1905.txt

[36] J. Case et al. *'Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)'* IETF Network Working Group RFC 1906, januar 1996
http://www.ietf.org/rfc/rfc1906.txt

[37] J. Case et al. *'Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)*' IETF Network Working Group RFC 1907, januar 1996
http://www.ietf.org/rfc/rfc1907.txt

[38] J. Case et al. *'Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework*' IETF Network Working Group RFC 1908, januar 1996
http://www.ietf.org/rfc/rfc1908.txt

[39] K. McCloghrie 'An Administrative Infrastructure for SNMPv2' IETF Netwrok Working Group RFC 1909, februar 1996
http://www.ietf.org/rfc/rfc1909.txt

[40] G. Waters ' User-based Security Model For SNMPv2' IETF Network Working Group RFC 1910, februar 1996
http://www.ietf.org/rfc/rfc1910.txt

[41] SNMP Version 3 (snmpv3) Charter, IETF Network Working Group, Active
http://www.ietf.org/html.charters/snmpv3-charter.html

[42] J. Case et al. *'Introduction to Version 3 of the Internet-standard Network Management Framework*' IETF Network Working Group RFC 2570, april 1999
http://www.ietf.org/rfc/rfc2570.txt

[43] D. Harrington et al. '*An Architecture for Describing SNMP Management Frameworks*', IETF Network Working Group RFC 2271, januar 1998
http://www.ietf.org/rfc/rfc2271.txt

[44] J. Case et al. '*Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)*', IETF Network Working Group RFC 2272, januar 1998
http://www.ietf.org/rfc/rfc2272.txt

[45] D. Levi et al. '*SNMP Applications'*, IETF Network Working Group RFC 2273, januar 1998
http://www.ietf.org/rfc/rfc2273.txt

[46] U. Blumenthal et al. '*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*, IETF Network Working Group RFC 2274, januar 1998
http://www.ietf.org/rfc/rfc2274.txt

[47] B. Winjen '*View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*' IETF Network Working Group RFC 2275, januar 1998
http://www.ietf.org/rfc/rfc2275.txt

[48] The NET-SNMP Project Home Page
www.net-snmp.com

[49] The FreeS/WAN Project Home Page
www.freeswan.org

[50] The Super-FreeS/WAN Home Page
http://www.freeswan.ca/code/super-freeswan/

[51] Tuomo Soinis Super-FreeW/WAN Source RPM's
http://tis.foobar.fi/software/?freeswan

[52] The OpenSSL Project Home Page
http://www.openssl.org/

[53] The OpenSSH Project Home Page
www.openssh.com

[54] Cisco Systems, Inc.
http://www.cisco.com/

[55] 'Cisco 2600 Series Multiservice Platforms' Cisco Product Guide, Cisco Systems
http://www.cisco.com/en/US/products/hw/routers/ps259/index.html

[56] 'Cisco IOS Software Release 12.2' Cisco Product Guide, Cisco Systems
http://www.cisco.com/en/US/products/sw/iosswrel/ps1835/index.html

[57]TAC '*Securing Simple Network Management Protocol'* Cisco Systems, 14 februar 2003
http://www.cisco.com/warp/public/477/SNMP/snmpsecurity-20370.pdf

[58] '*Setup SNMPv3 with IOS 12.0'*, Cisco IOS Software Release 12.0 T - Feature Guide, Cisco Systems
http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t3/snmp3.pdf

[59] TAC 'An Introduction to IP Security (IPSec) Encryption' Cisco Systems 26 februar 2003
http://www.cisco.com/warp/public/105/IPSECpart1.pdf

[60] TAC 'Configuring and Troubleshooting Cisco Network-Layer Encryption: IPSec and ISAKMP' Cisco Systems 07 mars 2003
http://www.cisco.com/warp/public/707/16.pdf

[61]TAC 'IP Security Troubleshooting - Understanding and Using debug Commands' Cisco Systems, 13 januar 2003
http://www.cisco.com/warp/public/707/ipsec_debug.pdf

[62]TAC 'Configuring Secure Shell on Cisco IOS Routers' Cisco Systems, 13 februar 2003
http://www.cisco.com/warp/public/707/ssh.pdf

[63] J. E. Freund 'Mathematical Statistics' 5th ed., Prentice Hall International Editions, 1992

[64] H. Krawczyk et al. 'HMAC: Keyed-Hashing for Message Authentication' IETF Network Working Group Request for Comment 2104, februar 1997
http://www.ietf.org/rfc/rfc2104.txt

[65] R. Rivest 'The MD5 Message-Digest Algorithm' IETF Network Working Group Request for Comment 1321, april 1992
http://www.ietf.org/rfc/rfc1321.txt

[66] 'Secure Hash standard' Federal Information Processing Standards Publication 180-1, april 1995.
http://www.itl.nist.gov/fipspubs/fip180-1.htm

[67] U. Blumenthal et al. 'Key Derivation for Network Management Applications' IEEE Network mai/juni 1997.

[68] U. Blumenthal et al. ' The AES Cipher Algorithm in the SNMP User-based Security Model', Internet Draft, draft-blumenthal-aes-usm-05.txt, februar 2003 (utgår august 2003)
http://www.ietf.org/internet-drafts/draft-blumenthal-aes-usm-05.txt

[69] 'Cisco IOS Configuration Fundamentals Configuration Guide Release 12.2' pp 331-350, desember 2002.
http://www.cisco.com/application/pdf/en/us/guest/products/ps4032/c1069/ccmigration_09186a008011dfe5.pdf

[70] IP Security Protocol (ipsec) Charter, IETF Network Working Group, Active
http://www.ietf.org/html.charters/ipsec-charter.html

[71] S. Kent and R. Atkinson 'Security Architecture for the Internet Protocol' IETF Networking Working Group Request for Comment 2401, november 1998
http://www.ietf.org/rfc/rfc2401.txt

[72] S. Kent and R. Atkinson 'IP Authentication Header' IETF Network Communication Group Request for Comment 2402, november 1998
http://www.ietf.org/rfc/rfc2402.txt

[73] S. Kent and R. Atkinson 'IP Encapsulating Security Payload (ESP)' IETF Network Communication Group Request for Comment 2406, november 1998
http://www.ietf.org/rfc/rfc2406.txt

[74] D. Maughan et al. 'Internet Security Association and Key Management Protocol (ISAKMP)' IETF Network Communication Group Request for Comment 2408, november 1998
http://www.ietf.org/rfc/rfc2408.txt

[75] D. Harkins and D. Carrel 'The Internet Key Exchange (IKE)' IETF Network Working Group Request for Comment 2409, november 1998
http://www.ietf.org/rfc/rfc2409.txt

[76] C. Madson and R. Glenn 'The Use of HMAC-MD5-96 within ESP and AH' IETF Network Working Group Request for Comments 2403, november 1998
http://www.ietf.org/rfc/rfc2403.txt

[77] C. Madson and R. Glenn 'The Use of HMAC-SHA-1-96 within ESP and AH' IETF Network Working Group Request for Comments 2404, november 1998
http://www.ietf.org/rfc/rfc2404.txt

[78] A. Keromytis and N. Provos 'The Use of HMAC-RIPEMD-160-96 within ESP and AH', IETF Network Working Group Request for Comments 2857, juni 2000
http://www.ietf.org/rfc/rfc2857.txt

[79] The Super FreeS/WAN Homepage
http://www.freeswan.ca/code/super-freeswan/

[80] O. Valsgård, '[Users] (Problem solved) Encapsulation of UDP/SNMP traffic' Posting at the Super-FreeS/WAN User list, juni 2003
http://lists.freeswan.ca/pipermail/sfs-users/2003-June/004080.html

[81] Transport Layer Security (tls) Charter, IETF Network Working Group, Active
http://www.ietf.org/html.charters/tls-charter.html

[82] T Dierks and C. Allen 'The TLS Protocol Version 1.0' IETF Network Working Group Request For Comment 2246, januar 1999
http://www.ietf.org/rfc/rfc2246.txt

[83] Secure Shell (secsh) Charter, IETF Network Working Group, Active
http://www.ietf.org/html.charters/secsh-charter.html

[84] HyperText Transfer Protocol (http) Charter, IETF Network Working Group, Concluded oktober 2000
http://www.ietf.org/html.charters/OLD/http-charter.html

[85] E. Rescorla *'HTTP over TLS'* IETF Network Working Group Request for Comments 2818, mai 2000
http://www.ietf.org/rfc/rfc2818.txt

[86] A. Bosselaers 'The hash function RIPEMD-160' Department Electrical Engineering (ESAT), Katholieke Universiteit Leuven, Netherlands
http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html

[87] H. Dobbertin 'The Status of MD5 After a Recent Attack' CryptoBytes, 2(2), 1996, pp. 1-6

[88] M. Bellare, R. Canetti, and H. Krawczyk 'The HMAC construction' CryptoBytes, 2(1), 1996, pp.12-15

[89] M. Bellare, R. Canetti, and H. Krawczyk 'Keying hash functions for message authentication' Advances in Cryptology – Crypto '96, Lecture Notes in Computer Science, Springer Verlag, 1996, pp. 1-15.

[90] 'Distributed.Net and EFF DES Cracker put the final nail into the Data Encryption Standard's coffin' Pressemelding Electronic Frontier Foundation 19 januar 1999
http://www.eff.org/Privacy/Crypto_misc/DESCracker/HTML/19990119_deschallenge3.html

[91] 'Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design' Electronic Frontier Foundation, 1998

Vedlegg

A Konfigurasjon av laptop (manager)

I denne delen presenterer vi konfigurasjonsfilene for laptopen.

A.1 Konfigurasjon av SNMP

Parametrene i konfigurasjonsfilen for ipsec ble variert alt etter som hvilke versjoner og sikkerhetsnivå vi skulle teste med. Parametrene ligger imidlertid klar og må bare fjernes kommentar for. Disse parametrene kunne også overstyres i kommandolinje

```
# /etc/snmp/snmp.conf – NET-SNMP configuration file
# file created by 2003.04.07 by orv

defVersion 3
# -v 1 | 2c | 3

#defCommunity  public
# -c public | protected | private

defSecurityName testuser3
# -u testuser3 | testuser4 | testuser5

defContext ""
# -n <contextname>

defSecurityLevel noAuthNoPriv
# -l noAuthNoPriv | authNoPriv | authPriv

#defAuthType MD5
# -a MD5 | SHA

#defAuthPassphrase authpswd
# -A <passphrase>

#defPrivType DES
# -x DES

#defPrivPassphrase privpswd
# -X <passphrase>
```

Tabell 6: Konfigurasjonsfil for snmp på laptop

Konfigurasjonsfilen gjør at følgende to kommandoer betyr det samme:

```
[root@jinx /]# snmpget –v 1 –c public 192.168.1.3 system.sysUpTime.0
[root@jinx /]# snmpget 192.168.1.3 system.sysUpTime.0
```

Tabell 7: Eksempel på snmp kommando

A.2 Konfigurasjon av IPSec

```
# /etc/ipsec.conf - FreeS/WAN IPsec configuration file
# basic configuration
config setup
        interfaces="ipsec0=eth0"
        klipsdebug=none
        plutodebug=none
        plutoload=%search
        plutostart=%search
        uniqueids=yes

# defaults for subsequent connection descriptions
conn %default
        disablearrivalcheck=no

# main connections descriptions
conn snmp
        type=transport
        #also=manual-keying
        also=auto-keying
        also=adresses
        #leftprotoport=17/161
        #rightprotoport=17
        auto=add

# common parameters
conn adresses
        left=192.168.1.3
        # leftsubnet=192.168.1.13/0
        # leftnexthop=%direct
        right=192.168.1.13
        # rightsubnet=192.168.1.0/0
        # rightnexthop=%direct

conn auto-keying
        #IKE
        keyexchange=ike #default
        authby=secret
        keyingtries=1
        ike=3des-md5-modp1024
        ikelifetime=8h
        #IPSEC
        #auth=esp                # default
        #esp=null-md5
        #esp=des-md5
        esp=3des-md5
        keylife=24h
        pfs=yes
        pfsgroup=modp1024

conn manual-keying
        #spibase=0x100
        # ESP
        spi=0x100       # inbound
        leftespspi=0x101 # outbound
        #esp=null-md5
        #esp=des-md5
        espenckey=0xfedcba9876543210
        esp=3des-md5
        espenckey=0xfedcba9876543210fedcba9876543210fedcba9876543210
        espauthkey=0x0123456789abcdef0123456789abcdef
        espreplay_window=64
        # AH
        #spi=0x102       # inbound
        #leftahspi=0x103   # outbound
        #ah=hmac-md5-96
        #ahkey=0x0123456789abcdef0123456789abcdef
        #ahreplay_window=64
```

Tabell 8: Konfigurasjonsfil for IPSec på laptop

B Konfigurasjon av ruter (agent)

I denne delen presenterer vi konfigurasjonen av ruteren. Denne kan inneholde småfeil, men gir en generell ide om hvordan konfigurasjonen kan se ut.

```
2600# show startup-config
Using 2746 out of 29688 bytes
!
version 12.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname 2600
!
enable secret 5 $1$PyIw$F/YsD6q8dDLelxhp.1S1U/
enable password kiliman
!
ip subnet-zero
!
!
!
!
crypto isakmp policy 10
 encr 3des
 hash md5
 authentication pre-share
 group 2
 lifetime 84600
 crypto isakmp key 0123456789abcdef address 192.168.1.13
!
!
crypto ipsec transform-set mytset13 esp-3des esp-md5-hmac
 mode transport
crypto ipsec transform-set mytset11 esp-des esp-md5-hmac
 mode transport
crypto ipsec transform-set mytset10 esp-null esp-md5-hmac
 mode transport
crypto ipsec transform-set mytset20 esp-null esp-sha-hmac
 mode transport
crypto ipsec transform-set mytset31 ah-md5-hmac esp-des
 mode transport
crypto ipsec transform-set mytset34 ah-md5-hmac
 mode transport
!
crypto map mycmap 10 ipsec-isakmp
 set peer 192.168.1.13
 set transform-set mytset13 mytset11 mytset10 mytset20 mytset31 mytset34
 set pfs group2
 match address 100
crypto map mycmap 20 ipsec-manual
 set peer 192.168.1.13
 set session-key inbound esp 257 cipher fedcba9876543210fedcba9876543210fedcba9876543210 authenticator
0123456789abcdef0123456789abcdef
 set session-key outbound esp 256 cipher fedcba9876543210fedcba9876543210fedcba9876543210 authenticator
0123456789abcdef0123456789abcdef
 set session-key inbound ah 259 0123456789abcdef0123456789abcdef
 set session-key outbound ah 258 0123456789abcdef0123456789abcdef
 set transform-set mytset11
 match address 100
!
!
!
voice call carrier capacity active
!
!
!
!
!
!
!
!
```

```
!
mta receive maximum-recipients 0
!
!
!
!
interface FastEthernet0/0
 ip address 192.168.1.3 255.255.255.0
 no ip mroute-cache
 speed auto
 full-duplex
 no cdp enable
!
interface FastEthernet0/1
 no ip address
 no ip mroute-cache
 shutdown
 duplex auto
 speed auto
 no cdp enable
!
ip classless
no ip http server
!
!
access-list 1 permit 192.168.1.13
access-list 100 permit ip host 192.168.1.3 host 192.168.1.13
access-list 101 permit tcp host 192.168.1.3 eq telnet host 192.168.1.13
access-list 101 permit tcp host 192.168.1.3 eq www host 192.168.1.13
access-list 101 permit udp host 192.168.1.3 eq snmp host 192.168.1.13
access-list 101 permit udp host 192.168.1.3 host 192.168.1.13 eq snmptrap
no cdp run
!
snmp-server user testuser3 public v3 access 1
snmp-server user testuser4 protected v3 access 1
snmp-server user testuser5 private v3 access 1
snmp-server group public v3 noauth access 1
snmp-server group protected v3 auth write v1default access 1
snmp-server group private v3 priv write v1defalut notify v1default access 1
snmp-server community public RO
snmp-server location FFIE-3c-358-lab30
snmp-server enable traps tty
call rsvp-sync
!
!
mgcp profile default
!
dial-peer cor custom
!
!
!
!
!
line con 0
line aux 0
line vty 0 4
 password kiliman
 login
!
!
end

2600#
```

Tabell 9: Konfigurasjon for ruteren