UNINETT Fagspesifikasjon

# RAMMEBETINGELSER OG KRAV TIL NETTVERKS-OVERVÅKNING I CAMPUSNETT

| | |
|---|---|
| UFS nr.: | 128 |
| Status: | Godkjent |
| Dato: | 03. 10. 2011 |
| Tittel: | Rammebetingelser og krav til nettverksovervåkning i campusnett |
| Arbeidsgruppe: | Overvåkning (gc-overvakning) |
| Forfattere | Vidar Faltinsen (UNINETT), Gro-Anita Vindheim (NTNU) |
| Kategori: | Anbefaling |

# Innhold

# Sammendrag

Denne anbefalingen setter krav og gir rammebetingelser til nettverksovervåking i campusnettverk. Anbefalingen er skrevet basert på UH-sektorens kollektive, mangeårige erfaring med drift og overvåking av campusnettverk.

Områdene fault management, accounting management og performance management dekkes. Innen hvert område spesifiseres hvilke funksjoner som må eller bør være dekket av nettverksovervåkingen.

Som det fremkommer er det svært mange oppgaver som skal ivaretas. I valget mellom et komplett totalsystem som favner alle krav eller et sett med mindre verktøy som settes sammen til en total-løsning anbefales sistnevnte. Det er viktig å vektlegge god integrasjon. Man bør prioritere et sentralisert, fleksibelt alarmsystem, samt en overbyggende portal med felles håndtering av autentisering og autorisasjon.

Anbefalingen vektlegger robusthet i overvåkingen. Lokalisering av overvåkeren må være gjennomtenkt, man bør overvåke overvåkeren, vurdere grad av redundans og ha gode rutiner for reetablering dersom overvåkingssystemet skulle havarere. Av hensyn til robusthet anbefales det ikke å virtualisere overvåkingen.

Sikkerhet må prioriteres. Her anbefales bruk av SNMPv3. Dersom valgte verktøy ikke støtter dette foreslås tiltak for å gjøre sikkerheten tilfredsstillende.

# 1 Innledning

Denne anbefalingen setter krav og gir rammebetingelser til nettverksovervåking i campusnettverk. Anbefalingen er skrevet basert på UH-sektorens kollektive, mangeårige erfaring med drift og overvåking av campusnettverk. UNINETT har gjennom GigaCampus-programmet (2006-2009) [1] koordinert fagområdet med en egen arbeidsgruppe innen overvåking. UNINETT har også utplassert overvåkingsmaskiner, de såkalte verktøykassene [2] og målepålene [3], i UH-institusjonenes campusnettverk. Verktøykassene og målepålene inneholder en samling av åpen kildekodeverktøy utviklet av UNINETT/UH-sektoren [4,5,6] eller andre.

## 1.1 Behovet for nettverksovervåking

IKT-avdelingens hovedoppgave er å understøtte virksomhetens mål. For et universitet eller høgskole vil det si å understøtte de mål som settes for undervisning, forskning og formidling. IKT-avdelingen leverer et sett med tjenester (og underliggende infrastruktur) for sine ansatte og studenter. Det stilles krav til disse tjenestene og disse kravene vil variere, men en generell trend er at man blir stadig mer avhenging av at det underliggende nettverket skal virke døgnet rundt, året rundt. Det skal ikke bare virke, det skal ha tilstrekkelig kapasitet, samt god og pålitelig responstid. Det må kunne håndtere en flora av ulike tjenester, med ulike karakteristika, fra sanntidsapplikasjoner, som IP-telefoni og videokonferanse, til ekstremt trafikkintensive dataoverføringer brukt i tungregning og annen forskning.

Nettverket er i seg selv komplekst. Det består av svært mye utstyr og kabling satt sammen i system. Tar vi et av de større campusnettverkene i sektoren, NTNU, som eksempel, består campusnettverket der av om lag 25 rutere, 1100 svitsjer og 1600 basestasjoner (med 20 kontrollere). Det går gigabitrater med trafikk døgnet rundt, året rundt. Dette skal virke 24x7.

Det er viktig å bygge inn feiltoleranse i nettverket, i hvert fall i de mest sentrale delene. Ideelt sett bør man unngå at en sviktende enkeltkomponent lammer hele eller deler av nettverket. Graden av redundans i nettverket vil være en kost/nyttevurdering. Her henviser vi til UFS 114: Feiltolerant Campusnett [7]. Men uansett hvor mye feiltoleranse man bygger inn, så *vil* problemer oppstå. Komponenter vil svikte og må da erstattes. Dette krever god overvåking som gir presise alarmer i feilsituasjoner. Overvåkingsverktøy spiller også en viktig proaktiv rolle ved at driftspersonell kan få indikasjoner på problemer i en tidlig fase og da kan løse disse før det blir kritisk.

## 1.2 Avgrensing

Nettverksovervåking er nært beslektet og har klare overlapp med system- og tjenesteovervåking. System- og tjenesteovervåking spiller en tilsvarende rolle, da myntet på tjenermaskiner og tjenester. Et tredje hovedområde er klientovervåking, der fokus er på drift og vedlikehold av klientmaskinene i nettverket. Vi behandler ikke system-, tjeneste- eller klientovervåking nærmere i dette dokumentet.

Dokumentet behandler heller ikke de underliggende driftsprosessene, dvs. de aktiviteter, metoder og prosedyrer som er nødvendig for god proaktiv IKT-drift. Dette er godt beskrevet i beste praksis rammeverket ITIL [8]. Husk at nettverksovervåking ikke har noen egenverdi. Det er utelukkende et hjelpemiddel for IKT driftspersonell. Verktøyene skal understøtte IKT avdelingens aktiviteter, metoder og prosedyrer for god drift, vedlikehold og videreutvikling av IKT infrastrukturen. Selv om verktøyene *er* viktige, så husk at organisering, personalressurser, arbeidsmetodikk, metode- og rutineverk er vesentlig viktigere.

La oss presist gjøre rede for hva nettverksovervåking egentlig er. Nettverksovervåking utgjør en sentral del av nettverksadministrasjon eller 'Network Management' på engelsk. Nettverks-administrasjon defineres som aktiviteter, metoder, prosedyrer og verktøy som understøtter drift og vedlikehold av nettverket. En vanlig måte å karakterisere nettverksadministrasjon er FCAPS[1] - Fault, Configuration, Accounting, Performance and Security [9].

| Nettverksadmin-områder | Forklaring |
|---|---|
| **Fault management** | Monitorere nettverket med underliggende komponenter. Detektere feil og sende alarmer. |
| Configuration management | Holde oversikt over alle komponentene i nettverket. Understøtte konfigurasjon av nettverksutstyret. Holde et arkiv med endringslogg. |
| **Accounting management** | Holde oversikt over trafikklast og trafikktyper. Ser også på helsetilstanden til nettverksutstyret (CPU-last, minneforbruk, miljødata, m.m.). |
| **Performance management** | Ser på nettverkets yteevne, herunder forsinkelse, pakketap, ytelse, jitter (varians i forsinkelsen). |
| Security management | Kontroll på tilgangen til nettverket og komponentene i nettverket. |

Områdene 'configuration management' og 'security management' anses å være *utenfor* skopet til nettverksovervåking og blir ikke behandlet videre i dette dokumentet. Hovedfokus for nettverksovervåking er 'fault management', altså det å monitorere nettverket og generere alarmer ved feilsituasjoner. Vi definerer også 'accounting management' og 'performance management' innenfor begrepet nettverksovervåking. I kapittel 2 gir vi en nærmere beskrivelse av disse tre funksjons-områdene.

## 1.3 Underliggende teknologier

Den mest utbredte metoden for å innhente data fra nettverksutstyr baserer seg på IETF-standarden SNMP (Simple Network Management Protocol), nærmere omtalt i vedlegg A.

I 2006 ferdigstilte IETF en ny standard, NETCONF. NETCONF er tiltenkt rollen som arvtager til SNMP, men det er foreløpig uklart om dette i praksis vil skje. NETCONF er omtalt i vedlegg B.

[1] FCAPS er et rammeverk for nettadministrasjon definert av ITU-T. FCAPS ble først introdusert gjennom ISO 10040 tidlig på 1980-tallet.

# 2 Funksjonsområder og krav

Vi går her nærmere inn på funksjonsområder og krav til nettverksovervåkingen. Inndelingen gitt i kapittel 1.2 blir fulgt, der områdene fault, accounting og performance management behandles.

## 2.1 Fault management

Fault management går ut på å monitorere komponentene i nettverket, detektere feil og sende alarmer. Dette er de basalt mest viktige oppgavene innen nettverksovervåking. Når en komponent svikter ønsker du en alarm og du ønsker den med en gang.

### 2.1.1 Monitorering

Et nettverksovervåkingssystem vil bestå av en rekke underliggende monitorer. En monitor har til oppgave å regelmessig sjekke om alt er i orden og i motsatt fall sende en alarm. Når feilen er utbedret sender monitoren en friskmelding. En monitor er gjerne dedikert til en spesiell oppgave:

- *Ping monitoren* sjekker at det er liv i alt utstyr (rutere, svitsjer, trådløst utstyr, tjenere, m.m.).
- *Interface monitoren* sjekker om interface/samband er operative
- *Modulmonitoren* sjekker at alle moduler i en modulær svitsj/ruter fungerer. Overvåking av strømforsyninger og viftemoduler inngår her.
- *Terskelmonitoren* sender alarm når trafikklast, CPU-last eller tilsvarende går over en definert grense.

Ping monitoren benytter ICMP echo (ping), mens interface-, modul- og terskelmonitor bør baseres på SNMP. Alle monitorene sender alarmer til alarmsystemet.

Innenfor tjenesteovervåking har vi i tillegg *tjenestemonitoren* som sjekker at alle tjenestene er operative. En god tjenestemonitor simulerer kommunikasjonsprotokollen til tjenesten og validerer om den får fornuftig respons (det er ikke nok å sjekke at TCP/UDP-porten er oppe).

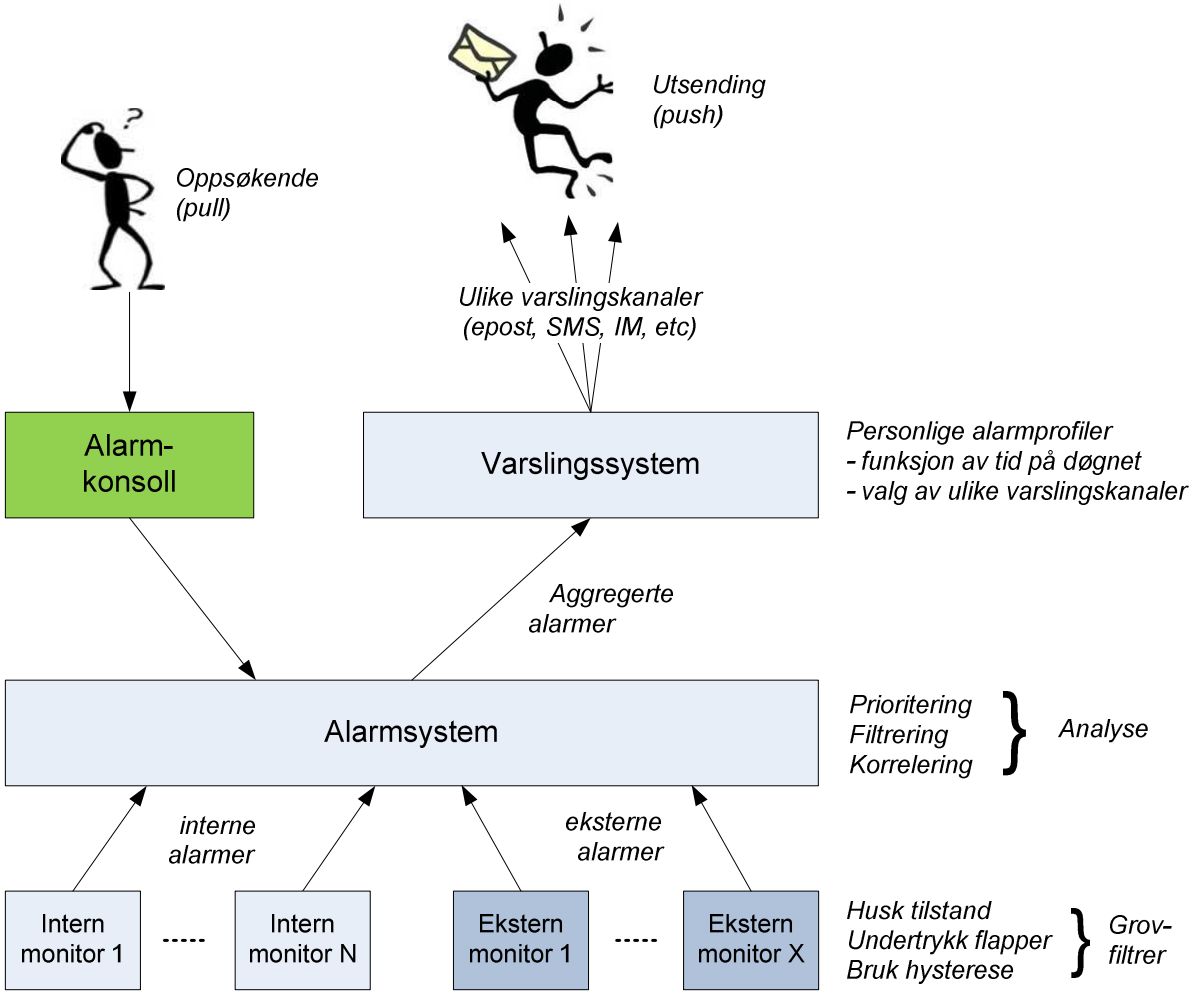
Følgende grovfiltrering bør gjøres av monitorene:

- Bygg inn robusthet i monitoreringen slik at ubetydelige forstyrrelser *ikke* blir rapportert som gjentagende nede- og oppealarmer. Dette skaper unødvendig støy. En bedre tilnærming er å rapportere slike kortvarige forstyrrelser som egne ”flapp-alarmer”.

- Hold rede på tilstand og ikke rapporter en gitt nedetilstand mer enn *en gang* til alarmsystemet. Sørg også for at den tilhørende oppemeldingen enkelt kan relateres til nedemeldingen, slik at alarmsystemet kan friskmelde hendelsen.
- I forhold til terskelalarmer, bruk hysterese med to definerte terskler, en sykmeldingsterskel og en lavere friskmeldingsterskel [2]. Et eksempel er at man ønsker alarm på 90% CPU-forbruk og friskmelder denne alarmen først når CPUen er under 80%. Dersom man også friskmeldinger på 90% kan man potensielt få en serie alarmer når lasten svinger rundt 90%.

### 2.1.2 Alarmsystem

De fleste overvåkingssystemer tilbyr enkle mekanismer for å sende en alarm når en hendelse inntreffer, for eksempel ved å kunne sende epost til prekonfigurerte epostadresser. Et fullverdig alarmsystem må ha vesentlig mer funksjonalitet enn dette. I kapittel 5 argumenterer vi for at du i din overvåkingsportefølje vil trenge *flere, ulike* verktøy. Vi anbefaler imidlertid å *samle alarm- og varslingsfunksjonaliteten til ett verktøy*, der de andre verktøyene sender sine alarmer til dette ene systemet. Figur 1 viser anbefalt arkitektur.




*Figur 1: Monitorer, alarmsystem og varslingssystem*

[2] gjelder alarmer for terskler målt på oppadgående flanke.

Verktøyet som har alarmsystemet har som regel også egne monitorer. *Intern* alarmrapportering kan realiseres på forskjellig proprietært vis, via database el.l, men for *eksterne* monitorer må en standard kommunikasjonsform benyttes. Av hensyn til fleksibilitet anbefales det at alarmmottaket støtter både SNMP trap og epost. Videre må alarmsystemet ha fleksible mekanismer for å tolke eksterne alarmer for igjen å kunne klassifisere dem. Det er i mange tilfeller urealistisk å tilpasse alarmformatet i det eksterne systemet. Det er da fordelaktig om man kan gjøre systemspesifikke tolkninger i det sentrale systemet.

Alarmsystemet har til oppgave å behandle innkommende alarmer og se de ulike alarmene i sammenheng. Denne analysefasen er viktig for å gi et best mulig alarmbilde videre til driftsoperatøren (via varslingssystemet). Følgende oppgaver ligger til alarmanalysen:

- Prioriter alarmer i ulik alvorlighetsgrad. Ved et større utfall kan det være svært mange alarmer som kommer samtidig og det er da viktig å kunne skille mindre viktige fra viktige alarmer.
- Korreler alarmer sendt fra forskjellige monitorer. Dersom de relaterer seg til samme hendelse, filtrer bort overflødige alarmer eller aggreger til en felles alarm. Dette kan være vanskelig å få til i praksis. Alarmsystemet må *ikke* gå på akkord med robusthet. Det er bedre med en alarm for mye enn en for lite.
- Korreler også alarmer i forhold til nettverkstopologi. Dette krever at alarmsystemet har kjennskap til topologien i nettverket. Ved større utfall kan alarmsystemet da avdekke hovedfeilkilde (”root cause”). Eksempelvis når en sentral ruter er nede vil overvåkeren være ute av stand til å vite om bakenforliggende komponenter faktisk er nede (med mindre det er en redundant vei). Driftsoperatør ønsker ikke hundrevis av nedemeldinger i et slikt scenario, men en tydelig melding om den sentrale komponenten som er nede. En mekanisme for å skille mellom en nedemelding og en ”ukjent status”-melding anbefales (der ukjent status får en lavere prioritet).

### 2.1.3 Alarmkonsoll

Behandlede eller aggregerte alarmer sendes videre til varslingssystemet som igjen sender alarmene til relevante driftsoperatører (mer under). Operatørene tilbys også en alarmkonsoll for å sjekke status. Et intuitivt grensesnitt er å markere tilstander med røde, gule og grønne lys. Et rødt lys gir en tydelig beskjed om at noe er galt. Dersom det er svært mange komponenter/hendelser som overvåkes, må alarmkonsollen støtte *gruppering* og en *hierarkisk* visning av alarmer i grupper og undergrupper. På den måten blir antall lys på toppnivå overkommelig og en rød eller gul alarm her kan forfølges til nivået under, eller nivået under der igjen, for å finne kilden.

### 2.1.4 Varslingssystem

Varslingssystemet har til oppgave å sende alarmer til driftsoperatørene. Typisk har ulike driftsoperatører ulike roller og/eller ansvarsområder, så muligheter for individuelt oppsett av alarmprofil er et rimelig krav. En operatør bør kunne justere i sin profil basert på alarmprioritet, utstyrstype/ kategori eller område alarmen kommer fra. Videre bør man fleksibelt kunne velge varslingskanal for ulike kategorier av alarmer. Epost og SMS må som minimum støttes, IM (instant messaging, herunder for eksempel XMPP eller IRC) kan med fordel støttes.

Driftsoperatøren bør også kunne sette opp ulike varslingsprofiler til ulike tider av døgnet. Eksempelvis ønskes kanskje ikke alarmer på kveldstid med mindre man har beredskapsvakt (dette gjelder særlig SMS). Eller kanskje ønsker man topp prioriterte alarmer også på kvelden og i helga.

Ved forsendelse av alarmer må varslingssystemet kunne køe meldinger basert på prioritet. Dette er essensielt for å sikre at de viktigste alarmene kommer frem og ikke drukner i køen av mindre viktige alarmer.

## 2.2 Accounting Management

Med accounting management forstås mekanismer for å holde oversikt over trafikklast og trafikktyper, samt helsetilstand på nettverkskomponentene.

### 2.2.1 Samle trafikktellere

For å få oversikten over trafikkvolumet i nettverket bør man regelmessig samle inn trafikktellere fra alle ruter- og svitsjeporter (med SNMP). Følgende data bør som et minimum samles inn:

- Trafikk inn/ut som bytes (octets) og antall pakker
- Feiltellere (feilpakker og forkastede pakker)
- Multicast/broadcast trafikkmengde

Vi anbefaler å samle inn data fra *samtlige* porter i nettverket. Det aller meste vil man aldri se på, men når en hendelse inntreffer er det ønskelig å kunne studere forskjellige tabeller/grafer fra ulike statistiske datasett for å kunne sannsynliggjøre hva som er årsaken til problemet. Uten datagrunnlag vil man lete i blinde og kanskje ikke komme noen vei.

Dataene må lagres i en underliggende database. Det er viktig med lav oppløsning på dataene for å kunne identifisere kortere perioder med unormalt trafikkmønster. Fem minutts oppløsning anbefales som et minimum.[3]

Man bør ta vare på data flere år tilbake for trendstudier. Her vil det være mest hensiktsmessig å aggregere eldede data slik at det totale lagringsbehovet blir overkommelig. Man må ved slik aggregering ta vare på maksimums- og minimumsverdier. Systemet må være fleksibelt ved at man selv kan definere reglene (tidsintervallene) for aggregering. Som et minimum må daglig, ukentlig og månedlig statistikk være enkelt å produsere.

Fleksible mekanismer for å søke i dataene og dernest presentere dem tabulært eller grafisk er essensielt. God responstid er en forutsetning. Følgende komponenter bør inngå:

- Topologiske nettverkskart som gir oversikt over trafikkflyten i nettverket til en konfigurerbar tid, der nåtid er default. Nettverkskartet bør være hierarkisk slik at man kan fordype seg i en enkelt del av nettverket.
- Et fleksibelt rapportsystem der man kan lage sammenstilte rapporter og oversikter. Rapportene må la seg sortere i alle kolonner.
- God grafisk visualisering med valg av ulike graftyper. Det må være enkelt å plotte ulike datakilder i samme graf, eller i grafer under hverandre, da med lik tidsakse. Dette vil gi driftsoperatøren et forbedret utgangspunkt for å studere en gitt hendelse.

Trendanalyse (Capacity Management i ITIL) kan også inngå i nettverksovervåkingssystemet, dvs. en analyse av trafikkvekst over tid som gir prognoser for trafikkvekst fremover.

### 2.2.2 Helsesjekk av nettverksutstyr

I tillegg til trafikktellere bør man samle inn informasjon fra rutere, svitsjer og annet nettverksutstyr som gir et bilde av deres ”helsetilstand”. Her inngår:

- CPU-last

[3] Ved bruk av SNMP til innsamling vil det ved 5 minutts intervall forutsettes 64 bits tellere for trafikkvolum i gigabitrater.

- Minneforbruk
- Diskforbruk (nvram, flash disk)
- Strømforbruk, herunder på PoE interface
- Temperatursensorer på utstyret (helst ved både ved vifteinntak og uttak)

Disse dataene er også viktige for å avgjøre om nettverket fungerer optimalt. Krav til lagring, aggregering og presentasjon av slike data blir som for trafikktellere.

### 2.2.3 Trafikktyper

Dette er data som gir oversikt over hva slags type trafikk det er i nettverket, herunder oversikt over hvilke IP-adresser som snakker med hvilke (både IPv4 og IPv6), i hvilket omfang (antall pakker og antall byte) og i hvilken tidsperiode. Dataene må kunne detaljeres ytterligere ned på TCP/UDP-portnivå. Å ha data for alle slike "transaksjoner" i nettverket er naturligvis ressurskrevende, så her vil man også måtte ha rutiner for å aggregere og slette data. Oppbevaringstid og tilgang til slike data må vurderes nøye og reguleres i henhold til gjeldende personvernlovgivning.

Dataene må kunne settes sammen for å gi trendoversikt i forhold til:

- Top talkers (inngående og utgående)
- Mest brukte avsender- og mottakterporter (TCP og UDP)
- Forhold mellom TCP, UDP, ICMP og annen trafikk
- Multicast trafikkmengde
- Trafikk på AS-nivå (autonomous system)

System for å samle inn slike data kan være passivt måleutstyr som ser på all forbigående trafikk på sentrale punkter i nettverket. Alternativt kan rutere eksportere slike data til nettverks-overvåkingssystemet. IETF standarden IPFIX (RFC 3917) bør da benyttes, alternativt kan det Cisco-proprietære Netflow formatet benyttes.

### 2.2.4 Maskinsporing

Maskinsporing er et eget område under accounting management. Det innebærer å samle på IP-mac bindinger fra rutere, både for IPv4 og IPv6, og brotabellsdata fra svitsjer for på den måten ha oversikt over når og hvor (hvilken svitsjeport) en gitt maskin koblet seg til nettverket.

Sammensatt gir disse dataene trend data om hvor mange maskiner som er i bruk i de ulike subnettene, samt andelen av maskiner som benytter IPv6.

Dataene er også svært nyttige ved sikkerhetsinsidenter der det foreligger en klage på en gitt IP-adresse til et gitt tidspunkt.

## 2.3 Performance Management

Det siste området, Performance management, studerer nettverkets yteevne, herunder forsinkelse, pakketap, ytelse og jitter (varians i forsinkelsen). Dette kan måles med utplasserte prober i nettverket som sender testdata mot ekkopunkter i nettverket. Ofte benyttes ping (ICMP ECHO) til dette formålet, men man kan også benytte UDP ECHO, evt. andre protokoller. Man må ta vare på følgende data:

- Rundreisetid
- Pakketap

- Jitter (varians i forsinkelsen)

En ulempe med målinger basert på rundreisetid er at man ikke vet om oppstått forsinkelse var på tur eller retur i nettverket. For å utbedre dette kan man benytte enveismålinger, men det krever at tiden er nøyaktig synkronisert for avsender og mottager (helst med GPS antenne). En annen ulempe kan være at man ikke vet på hvilket hopp i nettverket problemet oppsto. Dersom man måler mot hver ruter i nettverket vil man lettere kunne avgjøre hvor forsinkelsen inntraff. En potensiell feilkilde er at enkelte rutere vil prioritere slike henvendelser svært lavt og da gi en unaturlig dårlig responstid.

Et system der sluttbrukere kan måle sin ytelse frem til et målepunkt i nettverket er også et nyttig verktøy. Såkalte internett speedometer utfører en slik oppgave. Det anbefales en løsning som er i stand til å gi detaljer om nettverksytelsen, herunder pakketap, maksimal pakkestørrelse, avsatt bufferplass i endesystemene m.m.

Et annet viktig område er kvalitetsmålinger. Ved å se på pakkeavstand og sammenstille dette med tidsmerker i RTP-trafikk kan man si noe om nettverkets kvalitet i forhold til leveranse av tale- og videostrømmer.

# 3 Robust overvåking

For at overvåkingen skal være mest mulig robust må man vurdere en del sentrale rammebetingelser, herunder strategisk lokalisering av overvåkeren, overvåking av overvåkeren, SMS-oppsett, samt redundans i selve overvåkingen.

## 3.1 Lokalisering av overvåkeren

Planlegg nøye hvor dere lokaliserer overvåkingstjeneren i nettverket. En desentralt plassert overvåker kan med større sannsynlighet selv bli kuttet av fra resten av nettverket og således bli ute av stand til å gjøre jobben sin. Overvåkeren bør stå sentralt i nettverket i nærheten av de sentrale tjenermaskinene. Overvåkeren vil da ved de fleste scenarioer pålitelig kunne avgjøre hvilke tjenester som er operative og hvilke brukere som er nettverksmessig avskåret.

For å styrke påliteligheten ytterligere bør overvåkerne være på et subnett med redundant rute ut (via VRRP eller tilsvarende). Videre bør tjeneren ha redundant strømforsyning, der strømforsyningene får mating fra adskilte kilder, ideelt sett adskilte UPS-kilder.

## 3.2 Overvåking av overvåkeren

Under alle omstendigheter kan overvåkeren i seg selv dø, eller prosesser på overvåkeren kan stoppe opp. Det er derfor viktig å overvåke overvåkeren. Dette overses ofte, men bør prioriteres. En overvåker som slukner i all stillhet vil gi uheldige følger dersom noe uønsket skjer ellers i nettverket. En enkel overvåker av overvåkeren sjekker at den svarer på ping. En mer avansert, sjekker at alle nødvendige prosesser kjører, at filsystem ikke er fullt m.m.

Overvåkeren av overvåkeren bør være plassert på en adskilt fysisk lokasjon og den bør kunne sende SMS direkte til driftspersonell.

## 3.3 SMS som varslingskanal

Som nevnt i kapittel 2.1 bør alarmsystemet kunne sende ut alarmer på SMS, i tillegg til epost. SMS-utsending kan gjøres på flere måter, f.eks via en SMS-gateway på Internett, men det anbefales imidlertid *ikke*. Det ligger i overvåkingens natur at den skal kunne varsle når omverden er avskåret. Det mest robuste er således å ha en mobiltelefon eller tilsvarende GSM-enhet direkte tilkoblet serie/USB-port på overvåkingstjeneren. Vær oppmerksom på at GSM-dekningsgrad kan være dårlig i maskinrom som befinner seg i kjellerlokale bortgjemt i bygningsmassen. Ekstra ekstern antenne kan løse dette problemet.

## 3.4 Redundant overvåking

Som alle andre tjenere så må også overvåkingsmaskinene av og til tas ned for vedlikehold. Dette må man ha et avklart forhold til. Er det akseptabelt i et slikt planlangt vedlikeholdsvindu å klare seg uten overvåking? Hvis ikke, kan man da eksempelvis utnytte overvåkeren av overvåkeren til å i det minste gjøre en enkel tilstandsmonitoring av nettet?

Man må ta høyde for at overvåkeren kan havarere og ny maskin må etableres. Tre alternative løsningsforslag på problemet foreslås:

1. *Cold standby:* Reetabler overvåkeren så raskt som mulig.
   I dette scenarioet overvåk overvåkeren, ha en reservemaskin på lager, ha backup av overvåkingsdatabasen og andre data den samler inn (trafikkstatistikk, maskinsporingsdata, logger, IPFIX-/netflow-data m.m.). Ha videre en beredskapsvakt som opererer 24x7 og ha en god instruks for hvordan vedkommende kan reetablere overvåkeren dersom den dør.

2. *Hot standby:* Repliker overvåkeren til en overvåker du har i beredskap.
   Dette krever kontinuerlig replikering av overvåkingsdata. Det krever videre at primær overvåker overvåkes og at alarm på at noe er galt sendes beredskapsvakt som dernest manuelt med noen håndgrep kan sette i produksjon den sekundære maskinen. Et manuelt grep vil da være at den sekundære overtar IP-adressen til den primære og at man manuelt starter overvåkingsprosessene på den sekundære. Man kan muligens unngå å bytte IP-adresse, men det gir deg mange potensielle problemer, da svært mye er bundet mot IP-adressen eller DNS-navnet, herunder SNMP trap forsendelse fra nettelektronikk, syslogforsendelse. epostforsendelse av eksterne alarmer, SNMP filter i utstyr som kan overvåkes og selve webgrensesnitt til overvåkeren

3. Bruk to aktive overvåkere som er synkronisert seg i mellom.
   Dette er et mer komplekst oppsett der to maskiner overvåker og/eller er synkronisert og sømløst kan overta for hverandre. De bør plasseres på hver sin lokasjon og de bør også overvåke hverandre. Anycast[4] kan potensielt brukes for å unngå IP-adresse forvirring fra eksterne systemer, men det er mange forhold å tenke på for å få noe slikt til å virke. Vi går ikke nærmere inn på dette her.

Vår anbefaling er å gjøre dette relativt enkelt. Det er tross alt viktigere å bygge inn god redundans i de sentrale tjenestene enn i selve overvåkingen (overvåkingen har ingen egenverdi). *Alternativ 1 vil være en god nok løsning for de aller fleste.*

## 3.5 Virtuell overvåkingsmaskin

Dersom du velger å virtualisere overvåkingsmaskinene får man noen nye muligheter. En fordel vil være er at man enkelt kan reetablere overvåkingstjenesten ved havari, skjønt det fordrer at man har flere bladsystem, som i seg selv er redundante og med en redundant, virtuell arkitektur (basert på VMWare, KVM eller annet). Det fordrer også at man kan kjøre videre på samme IP-adresse på den erstattede virtuelle maskinen, jfr kap 3.4. Bygger man et tjenermiljø som ivaretar alt dette nærmer man seg en meget god løsning for tjenesteleveransene generelt, nettverksovervåking spesielt.

Man bør imidlertid vurdere robustheten i et slikt oppsett. En frittstående tjener som ikke er avhengig av noe annet enn seg selv er potensielt en mer robust overvåker enn en som er innvevd i bladsystem og virtuelle systemer. Husk at det er viktig å beholde en pålitelig overvåker også når det verste uhellet er

[4] Anycast er en mekanisme der samme IP-adresse opptrer på to eller flere maskiner i ulike deler av nettverket. Ruting til de ulike maskinene blir annonsert likeverdig fra hvert sitt hold. Andre maskiner på nettverket vil benytte den maskinen som står nærmest. Anycast gir en naturlig lastdeling, men også implisitt redundans. Tilstandsløse tjenester som DNS-resolvere er godt egnet som anycast-tjenester.

ute. Å raskt få identifisert feilkilden ved en pågående hendelse kan redusere utfallstiden for brukerne signifikant.

# 4 Sikker overvåking

*Her omtales SNMP og NETCONF. Les mer om dette i vedlegg A og B.*

Det er meget viktig å ivareta sikkerheten i forhold til dine rutere, svitsjer og annen nettelektronikk. For nettverksovervåking anbefales derfor SNMPv3 som besørger en sikker, kryptert kommunikasjon mellom overvåker og SNMP-agent. NETCONF kan på sikt være et godt alternativ, men det er foreløpig for få implementasjoner.

Realiteten er imidlertid at svært mange nettverksovervåkingssystem på markedet baserer seg på SNMPv2c. Her er sikkerheten mangelfull. Trafikken går i klartekst, inklusive SNMP "passordet" (community) som benyttes. En "man-in-the-middle" kan sniffe dette og da potensielt få tilgang til nettutstyret med SNMP. Dette er uheldig for SNMP leserettigheter og katastrofalt for SNMP skrivetilgang. Med sistnevnte har man makt til å restarte en ruter/svitsj, endre eller slette hele konfigurasjon.

Tross disse svakhetene finner vi det tilrådelig å benytte SNMPv2c til nettverksovervåking. Vi anbefaler da at følgende tiltak innføres:

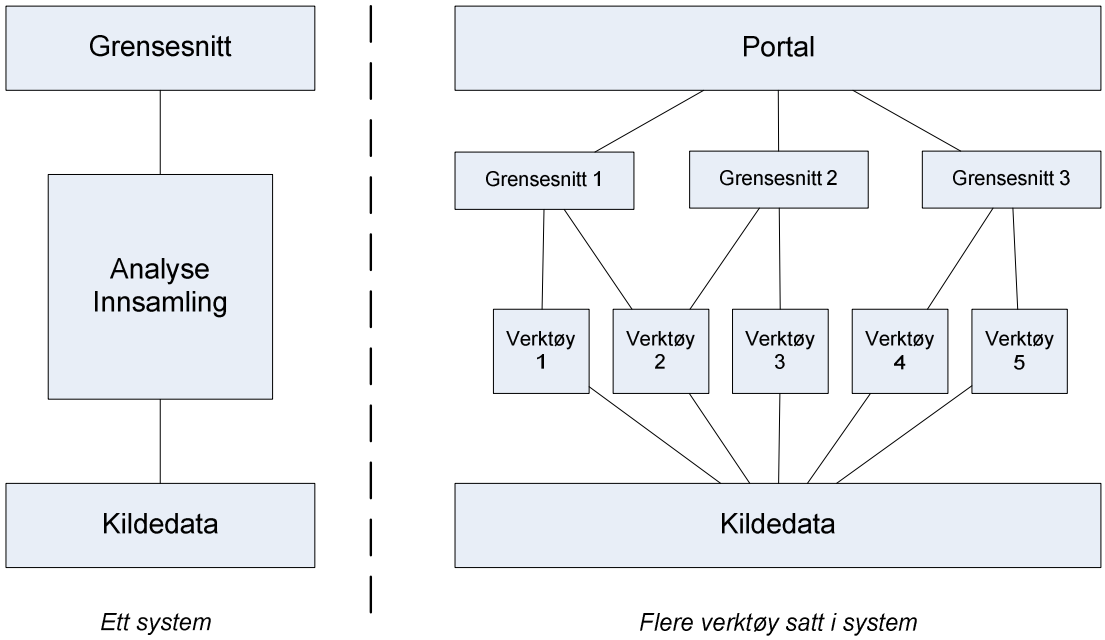
1. SNMP-tilgangen til nettverksutstyret begrenses så strengt som overhode mulig. Ingen andre maskiner enn overvåkingsmaskinene trenger å kommunisere med nettverksutstyret over SNMP.
2. Management IP-adresser til svitsjer bør være på et dedikert subnett hvor tilgangen er strengt regulert. Trådløse basestasjoner som er plassert i publikumsareal bør være på nok et dedikert subnett (ikke blandet med svitsjer).
3. Adgangen til selve overvåkingssystemet bør begrenses til autorisert personell. Dette gjelder både nettverksaksess via SSH og HTTPS og fysisk tilgang til maskinrom. For SSH og HTTPS er ikke brukerpålogging tilstrekkelig sikkerhet, begrens i tillegg hvilke subnett som får tilgang til maskinen (kun subnett hvor autorisert personell har adgang). Se forøvrig UFS122 [10] om anbefalt IKT-sikkerhetsarkitektur.
4. Vær varsom med autooppdagelse av nytt utstyr. Slik autooppdagelse vil typisk probe alle maskiner den finner på nettverket med SNMP "passordet" (community) som nettverksutstyret er konfigurert med. Det blir da *veldig* lett for tredjepart å sniffe dette passordet. Autooppdagelse på avgrensede subnett uten sluttbrukertilgang slik som dedikerte subnett for nettelektronikk er akseptabelt.

Uavhengig av hvilken SNMP versjon som brukes anbefales også følgende sikkerhetsperimeter:

5. Webgrensesnittet til overvåkingssystemet bør ha et innebygget autorisasjonssystem som gjør det mulig å på gruppenivå begrense adgangen til et utvalg av de underliggende webverktøyene.

# 5 Anskaffelse av nettverksovervåkingssystem

Gjennomgangen i kapittel 2 har vist at et nettverksovervåkingssystem må tilby et svært bredt spekter av funksjonalitet. Det ideelle ville være å anskaffe et komplett nettverksovervåkingssystem som støtter all denne funksjonaliteten. Alternativet er et sett med systemer som til sammen dekker den funksjonalitet man etterspør. Vi står ovenfor to alternative systemmodeller som vist på figur 2.




*Figur 2: To alternative systemmodeller for nettverksovervåkingsverktøy*

La oss se nærmere på fordeler og ulemper med de to alternativene før vi til slutt oppsummerer og konkluderer.

## 5.1 Ett komplett system

En åpenbar fordel med å anskaffe et komplett system er at man slipper å tenke på integrasjon mellom verktøy. Man får et enhetlig brukergrensesnitt å forholde seg til. Dersom systemet er kommersielt vil man antagelig ha mulighet for god support og man kan få pålitelige lovnader om videreutvikling av

systemet, samt jevnlige programvareoppdateringer m.m. Installasjon og oppdatering av systemet kan også være svært brukervennlig og enkelt.

Før anskaffelse av slike store systemer må man vurdere følgende forhold:

- Innkjøpspris sammenlignet med anskaffelse av et sett mindre system (der noen, sågar alle kan være åpen kildekode og gratis). Enkelte store systemer har en betydelig pris.
- Implementasjon av et stort system krever ressurser, herunder innleie av eksterne konsulenter, kursing av personell m.m. Ofte vil oppstartskostnaden være større enn innkjøpskostnaden.
- Supportavtale sammenlignet med å klare seg på egen hånd. Disse løpende kostnadene kan også være betydelige.
- Mulighet for å gjøre lokale tilpasninger/utvidelser. Dette er et viktig punkt som ikke må undervurderes. Det er nesten uten unntak nødvendig å tilpasse/justere systemet til lokale forhold. For noen systemer vil slik tilpasning koste mer enn det smaker. Det kan skyldes flere forhold, som at programmet er lite konfigurerbart, koden vanskelig tilgjengelig, systemet i seg selv komplekst, m.m. Dersom man gjør endringer i programkoden kan man også få problemer på sikt ved at man må manuelt vedlikeholde og repetere sine endringer ved fremtidige systemoppgraderinger. Husk at arbeidstimene som går med til tilpasning er en del av regnestykket. I mange tilfeller krever slike lokal tilrettelegging også innleie av konsulenter.

## 5.2 Et sett med mindre systemer

Dersom man velger et sett med mindre verktøy, er initiell investering og årlig vedlikehold lav eller potensielt gratis (åpen kildekode). Man har da også et verktøysett, gitt at verktøyene er modulære og godt dokumentert, som man med overkommelig innsats kan justere. Men dette er heller ikke gratis, man må sette av ressurser til lokal tilpasning under alle omstendigheter. Å erstatte et delsystem eller et enkeltverktøy blir uansett enklere.

En ulempe med små verktøy kan være at supportavtale ikke kan tegnes og/eller at support og programvareoppgraderinger blir upålitelige. Et lite åpent kildekodeverktøy har også større risiko for å dø ut enn et verktøy fra en større leverandør. Man må seriøst vurdere denne risikoen.

På den annen side kan et aktivt åpent kildekodeprosjekt med store brukermiljøer fungere meget godt. Systemet blir godt testet og kodebidrag kan komme fra mange hold. Det er imidlertid viktig at prosjektet har et pålitelig utviklingsteam i ryggen som kan styre og koordinere utviklingen. Skulle utvikling allikevel stoppe opp, vil man, gitt åpenheten, potensielt være i stand til å vedlikeholde og utvikle systemet selv, i egne rekker.

Velger man å satse på flere enkeltverktøy bør man under alle omstendigheter være nøktern. *For* mange systemer i drift er uheldig. Systemer satt i drift av enkeltpersoner som *ikke* brukes av hele driftsmiljøet bør definitivt *unngås*. Verktøyene i porteføljen bør heller ikke overlappe, i hvert fall ikke den funksjonaliteten man velger å bruke i de enkelte verktøy. Eksempelvis kan flere systemer ha en tjenesteovervåker eller en alarmsentral. Man bør velge ett og bare ett system for disse oppgavene.

Man bør søke å redusere kompleksitet når man kan. I valget mellom å ha et distribuert eller hierarkisk sett med monitorer som hver dekker en del av nettverket kontra å sentralisere dette til en maskin bør sistnevnte foretrekkes. Med dagens tjenerkraft burde ikke dette være noe problem. Kraftigere CPU, mer minne, bedret disk I/O er parametre man kan justere for en relativt billig penge.[5]

En åpenbar ulempe med å ha mange systemer i drift er at driftspersonellet får mange ulike grensesnitt å forholde seg til. Dersom hvert system har sin egen løsning for autentisering og autorisasjon vil dette også kreve mye i forhold til dublering av oppsett for brukernavn og passord m.m. Man bør prioritere

[5] Merk at for svært store campusnett kan det vise seg vanskelig å gjøre trafikkinnsamling av alle svitsjeporter fra en maskin. Dersom en gitt oppgave må distribueres, så velg i tilfelle en enkel og ryddig arkitektur.

god integrasjon mellom systemene. Et overbygg gjennom *en felles portal*, som vist på figur 2, med felles autentisering og autorisasjon er en god modell. Ett av de valgte verktøyene bør tilpasses til denne funksjonen.

En annen potensiell fallgruve er at man dublerer eller mangedobler innsatsen med å vedlikeholde kildedata. Eksempler på kildedata er informasjon om utstyret som skal overvåkes (med IP-adresse, SNMP community, plassering, m.m.), stedsinformasjon (navn, lokalisering av data-/komm.-rom m.m.) og organisatorisk (hvem er ansvarlig for utstyr). Man bør tilstrebe *en felles autoritativ kilde* (jfr figur 2) for slike data for på den måten å forenkle vedlikehold, samt redusere sannsynligheten for inkonsistens.[6]

## 5.3 Oppsummering og konklusjon

La oss sammenstille argumentene gitt i de forrige delkapitlene:

| | Et kommersielt NMS | Flere åpen kildekode verktøy |
|---|---|---|
| **Innkjøpspris** | | Lavest |
| **Supportavtale** | | Lavest |
| **Erstatte delsystem** | | Enklest å få til |
| **Integrasjon mellom verktøy** | Ingen problemstilling | |
| **Pålitelig forvaltning** | Kan ha fortrinn | |
| **Hyppige oppdateringer** | | Kan ha fortrinn[7] |

Nå er jo valget i realiteten mer sammensatt. De små verktøyene trenger eksempelvis ikke være åpen kildekode og man kan naturligvis velge et større kommersielt system der man søker å integrere mindre verktøy med dette. Dette blir også et spørsmål om økonomi, budsjettrammer og hvorvidt man har kultur og ønske om egenutvikling eller ikke.

Sistnevnte blir som regel bare en opsjon for de større miljøene. Slik har det også vært i Norge. UNINETT og de store universitetene har nytt godt av miljøer der kreative ildsjeler har fått boltret seg og over tid utviklet løsninger som steg for steg har forbedret driftshverdagen. For et mindre miljø er ikke dette en realistisk opsjon. Å kjøpe et komplett system blir da et forlokkende alternativ.

Men her er det viktig å lære av vår egen historie, og da vi mener UH-sektorens kollektive historie. Det finnes eksempler på institusjoner som har prøvd ut store, kommersielle NMS og blitt skuffet. Systemene har i følge produktark virket svært lovende, men ved en reell installasjon har man funnet mangler og svakheter. Å rette på dette, enten via leverandøren eller ved egen tilpasning, har ved flere tilfeller vist seg både tidkrevende og kostnadsdrivende.

Dette er selvfølgelig ikke et faktum som kan hugges i stein. Produkter utvikles hele tiden og nye kommer til. Men generelt sett bør man være skeptisk til store systemer som lover svært mye og gaper svært bredt. Å satse på flere mindre verktøy, der hvert enkelt verktøy har en mer spesialisert funksjon, har vist seg å være en mer levedyktig strategi.[8]

[6] Som et alternativ eller supplement til manuelt å fore systemet med kildedata, kan man automatisk oppdage nye komponenter i nettverket. Dette kan være en fordel, da det bedre vil sikre at en ruter eller svitsj ikke blir uteglemt fra overvåkingen. På den annen side kan det hende at komponenter man ikke ønsker å overvåke blir innlemmet automatisk. En kombinasjon av automatikk og manuell godkjenning er det beste. Vær imidlertid oppmerksom på at auto oppdaging basert på SNMPv2c kan være en sikkerhetsrisiko, jfr. kapittel 4.

[7] Store NMS kan være tungrodde og ikke gi så hyppige oppdateringer som aktive åpen kildekodeverktøy.

[8] Nettopp dette var en viktig motivasjon bak UNINETTs verktøykassetilbud [2] som ble introdusert under GigaCampus-programmet [1] i 2006. Det at UNINETT, på vegne av sektoren, påtok seg et tilretteleggings- og utviklingsansvar har gjort at også mindre høgskoler med lav risiko har kunnet velge en slik plattform. Det ligger åpenbare stordriftsbesparelser i en slik modell, noe en uavhengig konsulentrapport også har slått fast [11].

# A. SNMP

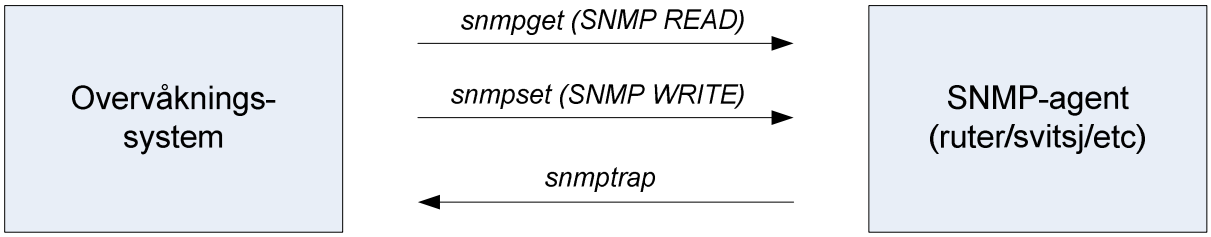
SNMP står for Simple Network Management Protocol og er opprinnelig definert som SNMPv1 i følgende RFCer:

- RFC 1155, mai 1990: Structure and Identification of Management Information for TCP/IP-Based Internets.
- RFC 1157, mai 1990: A Simple Network Management Protocol (SNMP).
- RFC 1213, mars 1991: (Version 2 of) Management Information Base (MIB) for Network Management of TCP/IP-based Internets.

Etter dette har en lang rekke RFCer kommet til, for eksempel OSPF v2 MIB (aug 91), BGP v3 MIB (okt 91) og RMON MIB (des 91).

SNMP består av tre grunnleggende kommunikasjonsformer (se figur 3):

- `snmpget`: Overvåkingsystemet poller agentene og innhenter ønsket informasjon (ønsket MIB-variabel/variabler).
- `snmpset`: Overvåkingssystemet endrer MIB-variable hos agenten for på den måten å endre konfigurasjon til agenten.
- `snmptrap`: Agenten sender selv en melding til overvåkingssystemet når en bestemt hendelse inntreffer.

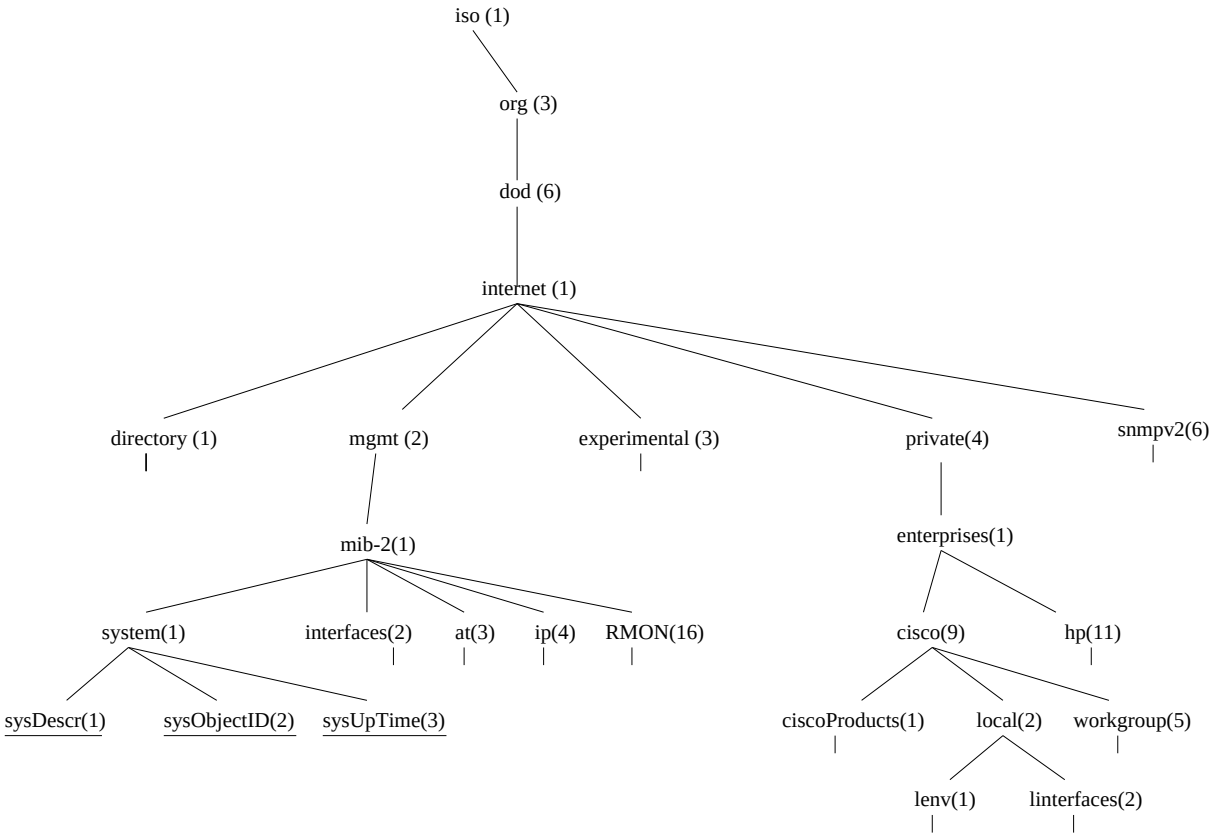



*Figur 3: SNMP kommunikasjonsformer*

UDP brukes som transportprotokoll for SNMP med de ulemper det har (UDP er forbindelsesløs, uten noen garanti for at pakken kommer frem). Det er mulig å implementere SNMP over TCP også, men dette er i praksis lite utbredt.

Informasjonen som agenten (nettverksutstyret) gjør tilgjengelig via SNMP er organisert i en trestruktur omtalt som MIB (Management Information Base). Trestrukturen og den syntaktiske oppbygningen som sådan er en standard. Videre er viktige grener av trestrukturen standardisert, herunder MIB-II. MIB-

strukturen tillater også private, proprietære subtrær, slik at de ulike leverandørene kan tilby systemspesifikk informasjon. Deler av trestrukturen er vist på figur 4.




*Figur 4: Utdrag fra MIB-strukturen (Management Information Base)*

Løvnodene i trestrukturen inneholder informasjonen som enten kan leses (`snmpget`) eller skrives (snmpset). På figuren vises tre løvnoder (`sysDescr`, `sysObjectID` og `sysUpTime`). Som et eksempel vil følgende forespørsel gi oppetiden til trd-gw1.uninett.no:

```
snmpget trd-gw1.uninett.no .1.3.6.1.2.1.1.3.0
```

SNMP-protokollen har blitt utbedret flere ganger. Sikkerheten i SNMP er omstridt og ble forsøkt utbedret i versjon 2, men IETF arbeidsgruppen ble ikke enig og man endte i januar 1996 på versjon 2c (RFC1901). Her sendes fortsatt SNMP ”passordet” (community) ukryptert over nettverket og kan potensielt sniffes (med mindre forholdsregler tas).

SNMPv2c gjorde andre forbedringer til protokollen, bl.a. ble `getbulk` introdusert, som muliggjør overføring av større mengder data i samme SNMP-forspørsel. SNMPv2c støtter også 64 bits tellere (mot de opprinnelige 32 bits tellerne) som er nødvendig for monitorering av trafikk i gigabithastighet.

Nyeste versjon av SNMP er v3 (RFC 3411-3418). Den ble endelig ble godkjent i 2004 og forelder v1 og v2c. I SNMPv3 er sikkerheten endelig utbedret. Autentiseringen foregår nå kryptert og man kan også velge å kryptere selve datatransporten.

Til tross for at SNMPv3 har vært en godkjent standard i mange år, så er det fortsatt SNMPv2c som dominerer i faktiske implementasjoner.

# SNMP-krav til nettverksutstyret

Ved anskaffelse av rutere, svitsjer og annen nettelektronikk er det viktig å vurdere utstyrets evne til å støtte ulike MIBer. Dette vil variere fra produsent til produsent. Som regel har produsenten sine egne proprietære MIBer hvor data gjøres tilgjengelig. Som et supplement er dette bra, men det er svært viktig at IETF sine standard-MIBer støttes fullt ut. Overvåkingssystemet bør i sin ende, såfremt det er mulig, basere seg på data fra standard MIBene. På den måten kan overvåkeren være en generisk SNMP-innsamler som kan overvåke utstyr fra en rekke produsenter.

Her følger en liste over minimumskrav som nettverksutstyret bør støtte:

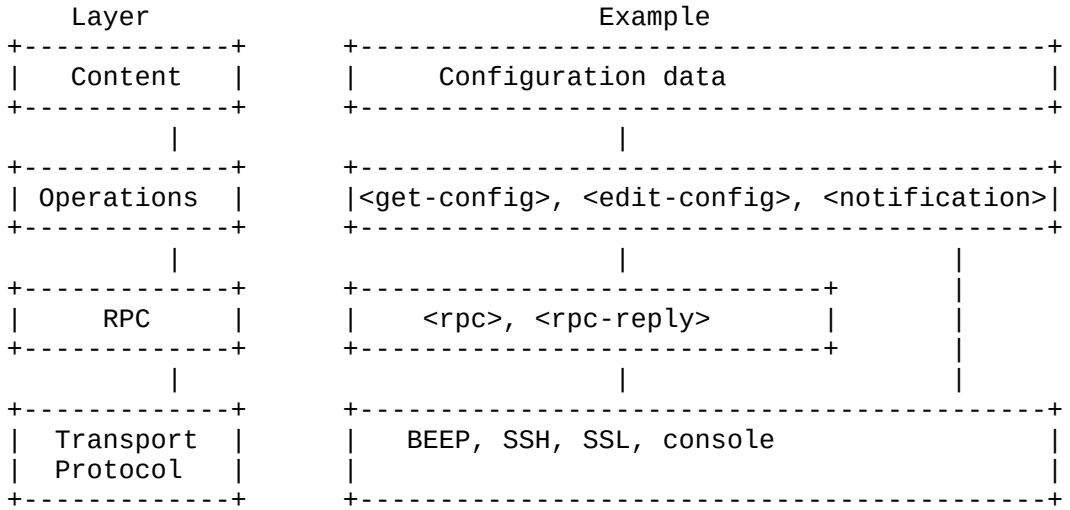
| | |
|---|---|
| RFC 3418: MIB II | (systemdata) |
| RFC 2863: IF-MIB | (interface, inklusive 64 bits trafikktellere) |
| RFC 4293: IP-MIB | (IP-interface og ARP; IPv4 og IPv6) |
| RFC 4133: ENTITY MIB | (moduler, optikk, programvareversjon, serienummer) |
| RFC 4188: BRIDGE-MIB | (brotabell for svitsjer) |
| RFC 4363: Q-BRIDGE MIB | (brotabell per vlan, vlan konfigurasjon) |
| RFC 3635: Etherlike-MIB | (dupleksdata for svitsjeporter) |
| RFC 2368: MAU-MIB | (fysisk medium for porter, for eksempel parkabel, fibertype, etc) |

# B. NETCONF

En årsak til at man har klart seg så lenge med svak sikkerhet i SNMP er at protokollen i praksis er mest brukt til monitorering. For å konfigurere utstyr er CLI mest utbredte metode. Det er flere grunner til dette, bl.a. at CLI er tekstbasert og enkel å forholde seg til. Dessuten implementerer ikke (de fleste) utstyrsleverandører full funksjonalitet gjennom SNMP.

En IETF arbeidsgruppe ble nedsatt for å se på en alternativ protokoll for å konfigurere nettverksutstyr på en sikker, skalerbar og fleksibel måte. Arbeidet munnet i desember 2006 ut i protokollen NETCONF (the Network Configuration Protocol , RFC 4741).

NETCONF tilbyr mekanismer for å installere, endre og slette konfigurasjonen til nettverksutstyr. Operasjonene er implementert over et RPC lag (Remote Procedure Call). NETCONF bruker en XML-basert (Extensible Markup Language) data enkoding. NETCONF kan kjøre ved hjelp av flere alternative transportprotokoller. Konseptuelt er NETCONF inndelt i fire lag:




XML kan være tungt å jobbe med og en egen IETF arbeidsgruppe, NETMOD, har definert et mer ”menneskevennlig” modelleringsspråk, YANG. YANG er definert i RFC 6020-6021 (oktober 2010). Det er pågående arbeid i NETMOD arbeidsgruppen der man ser på måter å styrke NETCONF sin kompabilitet med SNMP.

Det er foreløpig få implementasjoner av NETCONF og YANG.

# Referanser


**[1]** GigaCampus-programmet (2006-2009) : http://www.gigacampus.no

**[2]** UNINETTs verktøykassetilbud: https://ow.feide.no/gigacampus:verktoykasse

**[3]** UNINETTs målepåleplattform: http://forskningsnett.uninett.no/produkt/maalepale.html

**[4]** NAV (Network Administration Visualized): http://metanav.uninett.no/

**[5]** Stager: http://software.uninett.no/stager

**[6]** Programvare utviklet av UNINETT: http://software.uninett.no

**[7]** UFS 114 Feiltolerant Campusnett , https://ow.feide.no/gigacampus:ufs#nett

**[8]** ITIL (IT Infrastructure Library), http://www.itil-officialsite.com/

**[9]** FCAPS, http://en.wikipedia.org/wiki/FCAPS

**[10]** UFS122: Anbefalt IKT-sikkerhetsarkitektur i UH-sektoren
https://ow.feide.no/gigacampus:ufs#sikkerhet

**[11]** GigaCampus Lønnsomhetsbetraktning av 04.07.08
https://ow.feide.no/_media/gigacampus:gigacampus-lonnsomhet.pdf

# Definisjoner

| | |
|---|---|
| **CLI** | Command Line Interface |
| **GBIC** | **G**iga**B**it **I**nterface **C**onverter (fiberoptikk for GigaBit ethernet) |
| **HSRP** | Hot Standby Routing Protocol (Cisco proprietær) |
| **HTTPS** | HTTPS er en sikker utgave av HTTP, som er kommunikasjonsprotokollen til World Wide Web |
| **IM** | Instant Messaging |
| **IRC** | Internet Relay Chat |
| **ITIL** | Information Technology Infrastructure Library |
| **MIB** | Management Information Base |
| **NMS** | Network Management System |
| **RCS** | Revision Control System |
| **RRD** | Round Robin Database |
| **RSS** | Really Simple Syndication |
| **SFP** | Small Form-factor Pluggable (same funksjon som GBIC, men mindre formfaktor) |
| **SNMP** | Simple Network Management Protocol |
| **SSH** | Secure Shell |
| **TFTP** | Trivial File Transfer Protocol |
| **UPS** | Uninterruptible Power Supply |
| **VRRP** | Virtual Router Redundancy Protocol |

Ved spørsmål omkring denne eller andre UFSer – kontakt campus@uninett.no
Andre UFSer er tilgjengelige på www.uninett.no/ufs