FFI RAPPORT

RETNINGSLINJER FOR DESIGN AV MILITÆRE TAKTISKE IP-NETT

SORTEBERG Ingvild, KURE Øivind

FFI/RAPPORT-2005/02452

RETNINGSLINJER FOR DESIGN AV MILITÆRE TAKTISKE IP-NETT

SORTEBERG Ingvild, KURE Øivind

FFI/RAPPORT-2005/02452

FORSVARETS FORSKNINGSINSTITUTT
Norwegian Defence Research Establishment
Postboks 25, 2027 Kjeller, Norge

FORSVARETS FORSKNINGSINSTITUTT (FFI)
Norwegian Defence Research Establishment

P O BOX 25
NO-2027 KJELLER, NORWAY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

REPORT DOCUMENTATION PAGE

1) PUBL/REPORT NUMBER	2) SECURITY CLASSIFICATION	3) NUMBER OF PAGES
FFI/RAPPORT- 2005/02452	UNCLASSIFIED	32
1a) PROJECT REFERENCE	**2a) DECLASSIFICATION/DOWNGRADING SCHEDULE**	
FFI II/869/110	-	

4) TITLE

RETNINGSLINJER FOR DESIGN AV MILITÆRE TAKTISKE IP-NETT

DESIGN GUIDELINES FOR MILITARY TACTICAL IP-NETWORKS

5) NAMES OF AUTHOR(S) IN FULL (surname first)

SORTEBERG Ingvild, KURE Øivind

6) DISTRIBUTION STATEMENT

Approved for public release. Distribution unlimited. (Offentlig tilgjengelig)

7) INDEXING TERMS

IN ENGLISH:	IN NORWEGIAN:
a) Tactical IP based network	a) Taktisk IP-nett
b) Delay and jitter estimations	b) Forsinkelse- og jitterberegninger
c) QoS-Architecture	c) Tjenestekvalitetsarkitektur
d) Routing	d) Ruting
e) Network Time Protocol	e) Network Time Protocol

THESAURUS REFERENCE:

8) ABSTRACT

The objective of this report is to illustrate the type of performance expected in military tactical IP based networks and that it can be expected that most applications can be supported. However the performance heavily depends on parameters like transmission technology, network topology, QoS architecture, dimensioning, router architecture and implementation. Therefore the question is not if existing applications can migrate, but how the network must be design to support the requirements of these critical applications.

The amount of invested money and the network design will both determine if an IP network is capable of meeting requirements from new and existing applications.

9) DATE	AUTHORIZED BY	POSITION
	This page only	
2005-09-16	Vidar S. Andersen	Director

ISBN 82-464-0960-3

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

INNHOLD

		Side
1	INNLEDNING	7
2	MÅLSETNINGEN	7
3	PROBLEMSTILLINGEN	8
4	FORSINKELSESKOMPONENTER	9
4.1	Forsinkelseskomponenter gitt av ruterarkitektur.	9
4.2	Forsinkelseskomponenter som avhenger av belastning	10
4.2.1	Ruterbelastning	11
4.2.2	Ruterytelse	11
5	QOS ARKITEKTUR	11
6	METODE	13
7	SCENARIER	13
7.1	Topologi	13
7.2	Trafikkmatrise	14
8	DEDIKERT NETT MED 256 KBIT/S LINKER	15
8.1	Ingen QoS-mekanismer	15
8.2	Med QoS-mekanismer	16
8.3	Konklusjoner for dedikert nett	18
9	TAKTISK OMRÅDENETT MED 8 MBIT/S LINKER	19
9.1	Endring i topologi	19
9.2	Uten QoS-mekanismer	19
9.3	Med QoS-mekanismer	21
9.3.1	Trafikkmatrise	22
9.3.2	Medium trafikkbelastning	22
9.3.3	Høy belastning med omfordeling mellom klassene	22
9.3.4	Høy belastning uten mulighet for omfordeling	23
9.4	Konklusjon for et bredbånds taktisk nett	24
10	DYNAMISK RUTING	25
10.1	Referansetopologier og feilscenario	25
10.2	Resultater	25
10.3	Konklusjoner	27
11	TIDSSYNKRONISERING OG ESTIMERING AV FORSINKELSE I IP-NETT	27

11.1 Tidssynkronisering 27
11.1.1 Real-time Transfer Protocol (RTP) 28
11.1.2 Network Time Protocol (NTP) 28
11.2 Utfordringer 28
11.3 Konklusjoner 29
12 OPPSUMMERING 30
Litteratur 32

RETNINGSLINJER FOR DESIGN AV MILITÆRE TAKTISKE IP-NETT

1 INNLEDNING

Denne rapporten er utarbeidet som en del av Delprosjekt 1 (Migrasjon) i NbF Grid-prosjektet og TAMITA-oppdraget.

Forsvaret står ovenfor en omstilling fra vertikale kommunikasjonssystemer, der det er en tett kobling mellom applikasjon og kommunikasjonsteknologi, til IP-baserte nett. I den sammenheng er det ønsket å vurdere utfordringene en står ovenfor. Det opprinnelige målet med arbeidet har vært å identifisere en eller flere kritiske applikasjoner som per i dag benytter TADKOM-nettet og se om et typisk taktisk IP-basert nett kan tilby en tjenestekvalitet som tilfredsstiller kravene fra disse kritiske applikasjonene.

Formelt sett er dette ikke det korrekte målet. Problemet er mer sammensatt og består av tre hovedutfordringer, 1) hvordan signalerer applikasjonene sine krav, 2) hvilke metoder benytter nettet for å sikre tjenestekvalitetskravene gjennom for eksempel å reservere og allokere ressurser, og 3) innenfor hvilke grenser kan nettet gi en akseptabel ytelse og tjenestekvalitet til applikasjonene. I tillegg må en migrasjonsrapport identifisere spesielle krav til funksjonalitet som eksisterende viktige applikasjoner er basert på og inneholde skisser til mekanismer som kan tilby ekvivalente funksjoner i et IP-basert nett. For de to første spørsmålene henviser vi til en tidligere rapport (11) som kort oppsummerer hvilke alternative QoS-arkitekturer som tilbys innen IP-familen (IntServ og DiffServ). Begge arkitekturene har en tilstrekkelig semantikk til å møte de krav som stilles.

Når det gjelder selve ytelsen er ikke svaret like entydig. IP-protokollfamilien brukes i nett som tilbyr fra noen få bits/s til distribuerte systemer med gigabit/s kapasitet. En av forutsetningene for internettprotokollene har vært at det ikke skal stilles minimale funksjonelle krav til den underliggende transmisjonsteknologien, (1). Hvilke egenskaper et IP-nett tilbyr er derfor en funksjon av de underliggende teknologiske elementene. Linklagsteknologi, topologi, dimensjonering, trafikkpåtrykk, tilordning av ressurser til applikasjonene, ruterarkitektur, og ruterimplementasjon er alle faktorer som vil påvirke ytelsen til nettet. De fleste av disse variablene inngår som en del av et nettverksdesign. Kort oppsummert vil således investeringsrammer og selve nettdesignet være avgjørende for hvorvidt et IP-nett kan møte kravene fra eksisterende og nye applikasjoner. En migrasjonsplan må derfor inneholde designmål for det nye nettverket applikasjonene skal migrere til.

2 MÅLSETNINGEN

Målet med rapporten er å illustrere innen hvilke ytelsesrammer typiske takstiske IP-nett kan ligge. IP-nett er egnet som bærer for de fleste applikasjoner, men ytelsen til et IP-nett avhenger av underliggende teknologi, nett-topologi, QoS-arkitektur, dimensjonering, ruterarkitektur,

ruterimplementasjon, trafikkbelastning og trafikkmatrise. Spørsmålet om hvordan og når applikasjoner skal migrere er derfor ikke et spørsmål om IP-teknologien kan håndtere denne typen applikasjoner, men et spørsmål om hvordan nettet må designes for å håndtere applikasjonene.

Det er gitt at med pakkesvitsjing vil det alltid være et potensial for jitter eller variasjoner i forsinkelsen til pakkene som ankommer. Det kan potensielt finnes et lite sett med applikasjoner som stiller så ekstreme krav til jitter og ende-til-ende forsinkelse at en pakkesvitsjet teknologi ikke kan oppfylle kravene, men å identifisere om dette er tilfelle ligger utenfor mandatet til denne rapporten.

Utgangspunktet for forutsetingene og anbefalingene i denne rapporten er en luftvernapplikasjon som stiller strenge krav til forsinkelse og jitter. Rapporten gir ingen entydig konklusjon om luftvernapplikasjonen skal migrere over i det tenkte nettverket, siden det er en beslutning som bestemmes av operative krav fra også andre applikasjoner og anvendelser og de investeringsmulighetene Forsvaret får i løpet av de neste årene. Beslutningen om når og hvordan applikasjoner skal migrere til nye nettverk er derfor i hovedsak en funksjon av mulighetene til å investere i ny teknologi og oppgradere eksisterende applikasjoner. Forhåpentligvis er rapporten med på å gi retningslinjer for hvordan slike nett må bygges opp og innenfor hvilke rammer et nytt IP-basert taktisk nett må designes for å tilfredsstille de operative krav som vi forventer vil komme.

Rapporten oppsummerer resultatene av et sett av simuleringer og diskuterer disse. Den gir ikke et endelig svar på spørsmålet om et taktisk IP-nett dekker behovene. Derimot har målet vært å gi tilstrekkelig informasjon til at personer med god kunnskap om dagens og fremtidige applikasjoner skal kunne vurdere om kvaliteten er god nok og om hvilke designmessige krav og betingelser som må stilles både til nettet og til en eventuell migrasjon av eksisterende applikasjoner og utvikling av nye.

3 PROBLEMSTILLINGEN

På bakgrunn av diskusjoner internt i prosjektet og med representanter fra Forsvaret ble luftvernartilleri identifisert som en av de mest kritiske applikasjonene med hensyn på krav til forutsigbar tjenestekvalitet. Luftvernet påvirkes både av overføringens ende-til-ende forsinkelse og variasjonene i forsinkelse (jitter). Forsinkelsen i overføring av radarinformasjon oversettes direkte til en usikkerhet rundt målets geografiske posisjon. Denne komponenten må ses i forhold til den forsinkelse som ligger i selve radaren. Variasjonen i forsinkelse gir en usikkerhet knyttet til prediksjon av målets posisjon fremover i tid.

TADKOM har adressert denne problemstillingen ved å ha tidssynkronisering og tidsstempling av alle datapakker, slik at forsinkelse og jitter er gitt ved pakkemottak. Luftvernapplikasjonen kan derfor direkte vurdere usikkerheten som ligger i målposisjonen.

Med utgangspunkt i trafikkmatrisen for luftvernapplikasjonen simuleres pakkeforsinkelse og jitter for to ulike nettdesign. Det ene design er basert på medium båndbredderadioer (<300 kbit/s) i et separat luftvernnett, tilsvarende det en ser skisser av i enkelte europeiske land. Det andre nettdesignet er basert på områdenett med høyhastighets radiolinjer (8 Mbit/s). I tillegg til forskjellene i båndbredde har luftvernapplikasjonen også forskjellig prioritet i de to nettene. Dette er et design som en vil forvente i et NBF-nett hvor samme infrastruktur skal betjene ulike anvendelser.

Luftvernapplikasjonen er valgt som utgangspunkt for å beregne et estimat av forsinkelse og dermed også jitter. En analyse av best egnede protokoller og mekanismer for å gi tilsvarende funksjonalitet i IP-nett blir derfor en del av problemstillingen.

4 FORSINKELSESKOMPONENTER

I dette kapitlet beskriver vi kvalitativt de ulike komponentene som bidrar til forsinkelse og jitter i et IP-nett. Med pakkesvitsjing blir de ulike link- og ruterressursene fordelt stokastisk mellom ulike kommunikasjonsflyter. Alle faktorer som påvirker hvordan ressursene fordeles påvirker derfor forsinkelse og jitter. Grovt sett kan vi dele jitter- og forsinkelseskomponentene inn i to grupper; 1) de trafikkmessige komponentene som skyldes ressursdeling og køing i buffere, og 2) de komponentene som skyldes selve ruterarkitekturen. Den første gruppen vil fange opp de komponenter som kan påvirkes gjennom dimensjonering, mens den andre gruppen dekker faktorer som er låst når rutertypen er valgt. De to gruppene er ikke klart avgrenset, for eksempel vil kapasiteten til en ruter påvirkes av filtrering , QoS-arkitektur og pakkelengde, samtidig som den øvre ytelsesgrensen er gitt for den enkelte ruter.

Vi har valgt å fokusere på tre hovedgrupper; ruterteknologi, QoS-arkitektur og nettarkitektur. I tillegg vil radio- og mediumaksess-teknologien og geografisk utbredelse påvirke forsinkelse og jitter. De er det ikke tatt hensyn til i analysen. Variasjonene blant tilgjengelig teknologier er enorme, og dette må derfor vurderes når eventuelle investeringsbeslutninger skal tas.

4.1 Forsinkelseskomponenter gitt av ruterarkitektur.

Det finnes et bredt spekter av ruterarkitekturer, fra rutere basert på en vanlig PC med standard operativsystem til spesialiserte hardwareløsninger. Ruterens interne arkitektur vil påvirke hvilken kapasitet/ytelse den har i form av hvor mange pakker per sekund den kan behandle og videresende. Grovt sett kan vi anta at ytelsen er en funksjon av prisen.

Vanligvis foregår ruting som en software-prosess, og ytelsen blir en direkte funksjon av kapasiteten til den underliggende CPU. Det benyttes ulike typer av caching for å øke ytelsen på denne typen av rutere [2], men CPU-kapasitet vil være den begrensende ressursen. Pakkestørrelsen vil også påvirke ytelsen til ruteren siden ruteprosesseringen i vesentlig grad er uavhengig av pakkestørrelse. Hvis trafikken inneholder en stor andel små pakker reduseres ruterens kapasitet beregnet i båndbredde (bit/s).

De ulike tjenestekvalitetsfunksjonene som aktiveres på ruteren vil også påvirke kapasiteten. Historiske tall viser at med mange QoS-funksjoner aktivert, kan man oppleve opptil en halvering av ytelsen for noen modeller. Et tilsvarende argument kan formuleres for aktivering av ulike managementfunksjoner.

Kraftigere rutere, som i stor grad benyttes i kjernenettet, har en annen arkitektur hvor hvert linjekort har en prosessor som håndterer de fleste funksjonene tilknyttet selve pakkeprosesseringen og videresendingen. Disse ruterne har en høyere ytelse og er mindre sårbare for overbelastning siden den totale prosesseringskapasiteten er større og kan bygges ut etter hvert som nye linjekort settes inn.

Noen ruterleverandører har optimalisert arkitekturen for full utnyttelse av linjehastigheten. Dette gjør det nødvendig med et ”speed matching” buffer mellom selve ruterprosessering og utklokkingen av pakken. Ved høy belastning vil det være noen få pakker i et slikt utjevningsbuffer, noe som vil bidra til jitter og økt forsinkelse. Gevinsten med bufring er å sikre at overføringskapasiteten blir fullt utnyttet. Det finnes flere publiserte målinger som viser effekten av slike buffere for ulike arkitekturer og belastninger. Behovet for et slik buffer er størst ved høye båndbredder og bufferstørrelsen kan konfigureres. I taktiske nett med moderat båndbredde er behovet for et slikt buffer mindre og det bør derfor være lite.

Multicast, eller gruppekommunikasjon, er forventet å inngå som en del av NBF-nett. Prosesseringen av multicast- og unicast-pakker er forskjellig. Multicast-pakker har en mer kompleks prosesseringsflyt og historisk har dette ført til dårligere ytelse. I tillegg til en mer kompleks prosessering, skyldes den lavere ytelsen at ruterleverandørene i mindre grad har fokusert på å optimalisere funksjoner som ikke er så etterspurt. Etter hvert som multicast anvendes i større og større grad for mediadistribusjon kan vi regne med at prosesseringen blir optimalisert og ytelsen forbedret. Vi kan også forvente at deler av multicast-prosesseringen flyttes over i dedikert hardware. Det er derfor vanskelig å estimere kapasitet, jitter og forsinkelse for multicast. Imidlertid bør kapasiteten på håndtering av multicast, både i båndbredde og i antall deltagere og trafikkstrømmer, inngå i vurderingskriteriene ved valg av rutere siden det potensielt kan være store forskjeller mellom leverandører og ulike rutermodeller.

4.2 Forsinkelseskomponenter som avhenger av belastning

Hovedmotivasjonen for pakkesvitsjing er multipleksing av videresendingsressurser mellom ulike kilder med variabel trafikk. Når trafikkpåtrykket i korte perioder er større enn kapasiteten skal pakkene mellomlagres. Antallet pakker som kan mellomlagres, dvs størrelsen på bufferet, vil påvirke ende-til-ende forsinkelsen og variasjonene i forsinkelsen. Forsinkelse og jitter er derfor funksjoner av trafikkmønstre og gjennomsnittlig belastning.

En ruter har muligheten til fordele forsinkelse og variasjon ulikt på de forskjellige trafikkstrømmene. QoS-arkitekturen som bestemmer skeduleringen av pakker og algoritmen for

å kaste pakker når bufferne begynner å fylles opp avgjør hvordan forsinkelse, jitter og pakketap fordeles på de ulike trafikkstrømmene. Det finnes mange ulike skeduleringsprinsipper. Et alternativ kan være prioritet med pre-emption, hvor forsinkelsen kun er en funksjon av antall pakker på samme eller høyere prioritetsnivå. Andre alternativer er round robin skedulering, hvor ressursene fordels jevnt mellom de ulike strømmene, og FIFO (First In-First Out), hvor pakkene sendes ut etter hvert som de ankommer. Det er derfor relativt mange måter å utnytte forskjellige mekanismer for å fordele forsinkelsen mellom de ulike strømmene. QoS-arkitekturen vil således bestemme hvordan forsinkelsen skal fordeles.

4.2.1 Ruterbelastning

Ende-til-ende forsinkelsen påvirkes av belastningen på den enkelte ruter. Ved lav belastning vil ruteren rekke å behandle pakkene etter hvert som de ankommer. Ved høy belastning kan derimot pakker oppleve å bli bufret, mens de venter på ledig prosessorkapasitet eller på å bli klokket ut.

Køtiden vil normalt ha to elementer, den tiden pakken må vente før den skeduleres for sending og den tiden pakken må vente i ruterens utklokkingsbuffer (Tx-buffer).

4.2.2 Ruterytelse

Rutere må håndtere mange oppgaver i parallell, pakkeprosessering, management og ruteoppdateringer. Avhengig av ruterens arkitektur kan ruterens pakkeprosesseringsrate reduseres som følge av at ruteren må gjøre andre oppgaver, f eks oppdatere rutetabeller. For enkelte rutere varierer også ytelsen med pakkestørrelsen, dvs at ruterens ytelse i bit/s reduseres ved bruk av korte pakker.

Eksakt hvordan ytelsen påvirkes avhenger av ruterens hardware- og softwaredesign. Noen funksjoner krever ekstra prosessering eller er ikke i stand til å utnytte spesialiserte hardwareløsninger. Eksempler på dette kan være:

- Køing
- Droppfunksjoner som taildrop og WRED (Beskrevet i kapittel 5)
- Sikkerhetsfunksjoner som kryptering dersom dette gjøres i ruteren
- Komprimering
- Avansert filtrering
- Taksering eller detaljert innsamling av trafikkstatistikk

For å få en oversikt over hvilken ytelse man kan forvente vil det være nødvendig å teste ruterne i et oppsett som tilsvarer det oppsettet man ønsker å benytte i det virkelige nettet.

5 QOS ARKITEKTUR

IETF har standardisert to ulike QoS-arkitetkturer, IntServ (7)og DiffServ (3). Disse skiller seg gjennom at IntServ kan gi garantier for kvaliteten til hver enkelt trafikkstrøm gjennom at

ressurser reserveres per strøm. DiffServ derimot gir ingen sikre garantier, men tilbyr mekanismer som øker sannsynligheten for suksess for pakker som gis en høyere prioritet enn andre. Tre tjenesteklasser er definert, EF (Expedite Forwarding), AF (Assured Forwarding) og BE (Best Effort) (5), (6),.

EF-klassen gir den laveste forsinkelsen og det minste jitteret samtidig som sannsynlighet for pakketap er lav. For å sikre lav forsinkelse og lite jitter implementeres EF-klassen gjerne som en prioritetskø hvor trafikken alltid prioriteres så lenge den ikke overskrider en definert andel av den totale båndbredden. Prioritetskøen bør også være liten, noe som gjør EF-klassen mindre egnet for trafikk med store variasjoner i senderaten siden det kan føre til pakketap på grunn av overbelastning av bufferne.

AF-klassen består egentlig av fire subklasser med forskjellige kvalitetsnivåer. AF-klassen gir noe høyere forsinkelse og jitter enn EF-klassen, men tilbyr i tillegg ulike nivåer av droppsannsynlighet innen en klasse. Dette brukes for å sikre nettet mot trafikk-kilder med stor variasjon i belastningen og mot store endringer i trafikkstrømmene gjennom at ruterne kan merke pakker med høy, medium og lav droppsannsynlighet.

BE-klassen gir ingen garantier for forsinkelse eller pakketap og bør derfor brukes av applikasjoner som kan håndtere store variasjoner i tjenestekvaliteten, f eks TCP-baserte applikasjoner.

Implementeringen av AF- og BE-klassene skjer gjerne gjennom enten bruk av algoritmer som vekter trafikken ulikt basert på DiffServ-merkingen (4). Typiske algoritmer er WFQ (Weighted Fair Queueing) og WRR (Weighted Round Robin). Disse baserer seg på at de ulike klassene gis en vekt som illustrerer andelen av ressurser som tildeles denne klassen over et definert tidsintervall. Gjennom å vekte klassene på ulike måter kan man kontrollere fordelingen av båndbredden i nettet.

Det er også ulike strategier for hvordan pakker skal kastes dersom det oppstår en overbelastning. Den enkleste metoden er ”taildrop”, der pakkene kastes etter hvert som de ankommer er overfylt buffer. Siden enkelte protokoller, f eks TCP, responderer på tap ved å sette ned senderaten, har andre strategier blitt utviklet for å sikre en bedre utnyttelse av ressursene når trafikkbelastningen varierer. Typiske algoritmer er RED (Random Early Discard) og WRED (Weighted Random Early Discard). Disse algoritmene utnytter egenskapen til de applikasjonene som senker senderaten når de oppdager at pakker ikke kommer frem. RED og WRED kaster derfor pakker før bufferne er overfylte. Dette skal hindre en enda større overbelastning og samtidig kan kastingen begrenses til et fåtall trafikkstrømmer. Ruterne som benytter RED eller WRED signalerer dermed en mulig overbelastning og trafikkpåtrykket kan reduseres før nettet overbelastes helt samtidig som at pakketapet konsentreres til et mindre antall trafikkstrømmer enn hva som kunne vært tilfelle ved bruk av tilfeldig kasting eller taildrop. For applikasjoner som benytter seg av andre transportprotokoller enn TCP er ikke effekten den samme. Her er man avhengig av at applikasjonene selv responderer på pakketapet og her vil det finnes ulike

strategier og man kan derfor ikke gjøre noen forutsetninger.

6 METODE

Hovedtesen for rapporten er at det er mulig å designe et taktisk nett som har en akseptabel forsinkelse og et lite nok jitter til at det kan støtte luftvernsapplikasjoner. Samtidig er det åpenbart at det er mulig å designe et IP-nett som har en helt uakseptabel ytelse. Hensikten med rapporten er å illustrere et mulig utfallsrom for et IP-basert taktisk nett ved hjelp av simuleringer.

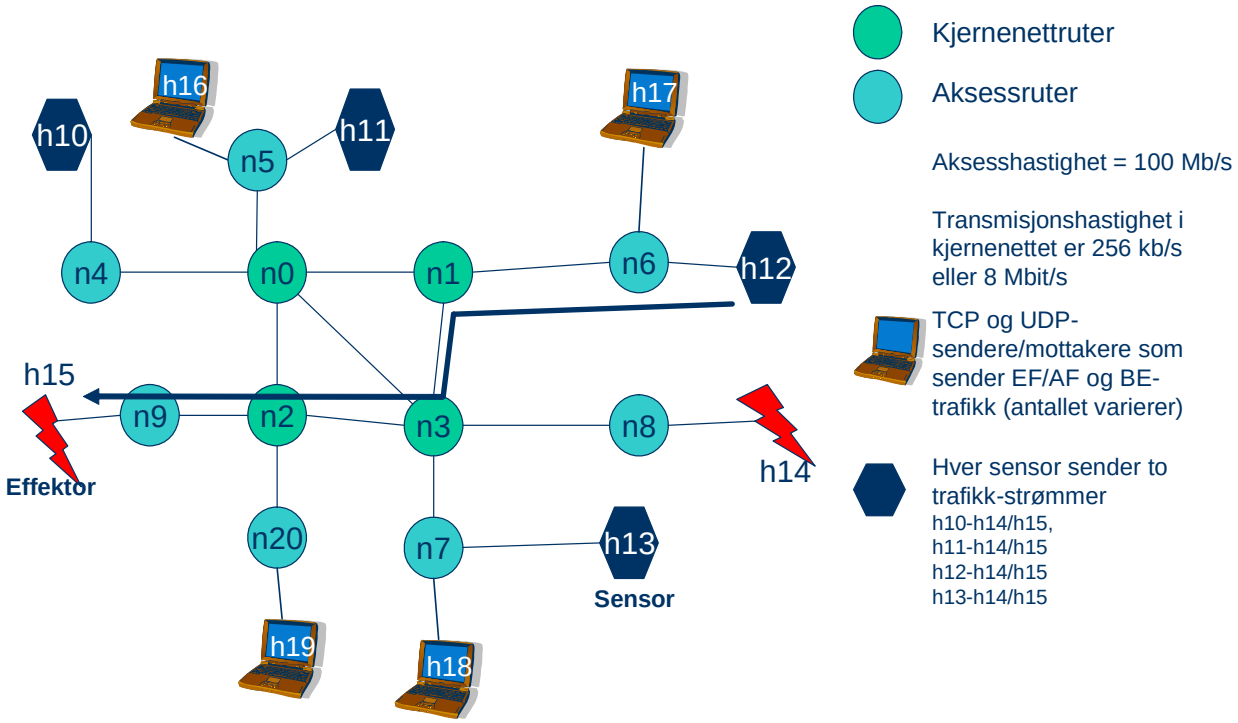
Forsinkelse og jitter er en funksjon av topologi, ruterimplementasjon, trafikkbelastning og QoS-arkitektur. For å bevare oversikten benyttes kun to ulike topologier, et maskenett og et kjedenett. Båndbredden, trafikkbelastningen og QoS-arkitekturen endres. I nett-topologien har vi ikke tatt hensyn til redundans, og all trafikk rutes langs korteste vei. Dersom MPLS eller andre lastbalanseringsmekanismer blir brukt vil dette ha innvirkning på trafikkbelastningen og dermed også resultatene. Ruterimplementasjonen er ikke med som en faktor i simuleringen. Simuleringsverktøyet som er benyttet er J-SIM v1.3.

7 SCENARIER

I simuleringene har vi forutsatt to ulike radionett, et dedikert radionett med lav båndbredde (256 kbit/s linker) og et mer generisk taktisk områdenett med høy båndbredde (8 Mbit/s linker). Denne siste typen nett vil brukes av mange ulike typer applikasjoner, og det vil derfor være vanskeligere å sikre ressurser kun til en applikasjon.

7.1 Topologi

Basistopologien består av terminaler som er tilknyttet aksessrutere gjennom en høyhastighets LAN-forbindelse. Aksessruterne er igjen tilknyttet et kjernenett. Kjernenettet består av fire ruter og trafikken går normalt over opptil to hopp i kjernenettet, se Figur 7.1. Linkene mellom kjernenettruterne og aksessruterne samt aksessnettverket er forutsatt å være 100 Mbit/s Ethernet, slik at all køing vil skje på de taktisk kjernenettruterne.



Figur 7.1 Referansetopologi

7.2 Trafikkmatrise

I simuleringen sender fire sensorer (h10-h13) til to effektornoder (h14 og h15). Hver sensor sender 100 byte pakker hvert 50 ms, som tilsvarer en 16 kbit/s strøm. Transportprotokollen som brukes er en upålitelig protokoll som verken tilbyr retransmisjon eller ratekontroll. Det anbefales ikke å bruke en pålitelig transportprotokoll som TCP. Ende-til-ende retransmisjon vil ikke være hensiktsmessig dersom tiden det tar overstiger tiden mellom hver gang sensoren genererer en ny pakke med nye måldata. En annen årsak er at pakketap vil i mange tilfeller skyldes overbelastning og en retransmisjon kan derfor føre til enda større overbelastning som igjen kan resultere i økt pakketap og forsinkelse.

Det er ikke gjort noe forsøk på å relatere denne modellen opp imot hva som faktisk gjøres i en luftvernapplikasjon, siden segmenteringen av sensorinformasjonen over i IP-pakker gjøres av applikasjonen. Det er således et designvalg ved en migrasjon av applikasjonen. Modellen med små pakker som sendes ofte, gir høy overhead men mindre jitter enn en modell med store pakker som sendes sjeldnere. Resultatene representerer derfor et designvalg optimalisert for å redusere jitter. Hvis ikke annet er oppgitt måles forsinkelse og jitter mellom nodene h12 og h15.

I tillegg til sensortrafikken eksisterer det en del andre trafikk-kilder (h16-h19 i figuren). Antallet trafikk-kilder varierer noe for de ulike scenariene og trafikk-karakteristikken til disse er derfor beskrevet nærmere i presentasjonen av resultatene.

I dette eksemplet sender hver sensor den samme informasjonen til to effektorer, I andre tilfeller kan man se for seg at antallet effektorer som skal motta informasjon er høyere og da kan det

være interessant å vurdere bruk av multicast for å redusere belastningen i nettet. Det vil åpne for å frigjøre ressurser til andre anvendelser over samme infrastruktur, men kan også føre til at man mister noe av kontrollen over trafikkflyten og dermed får mindre kontroll med ressurstildelingen.

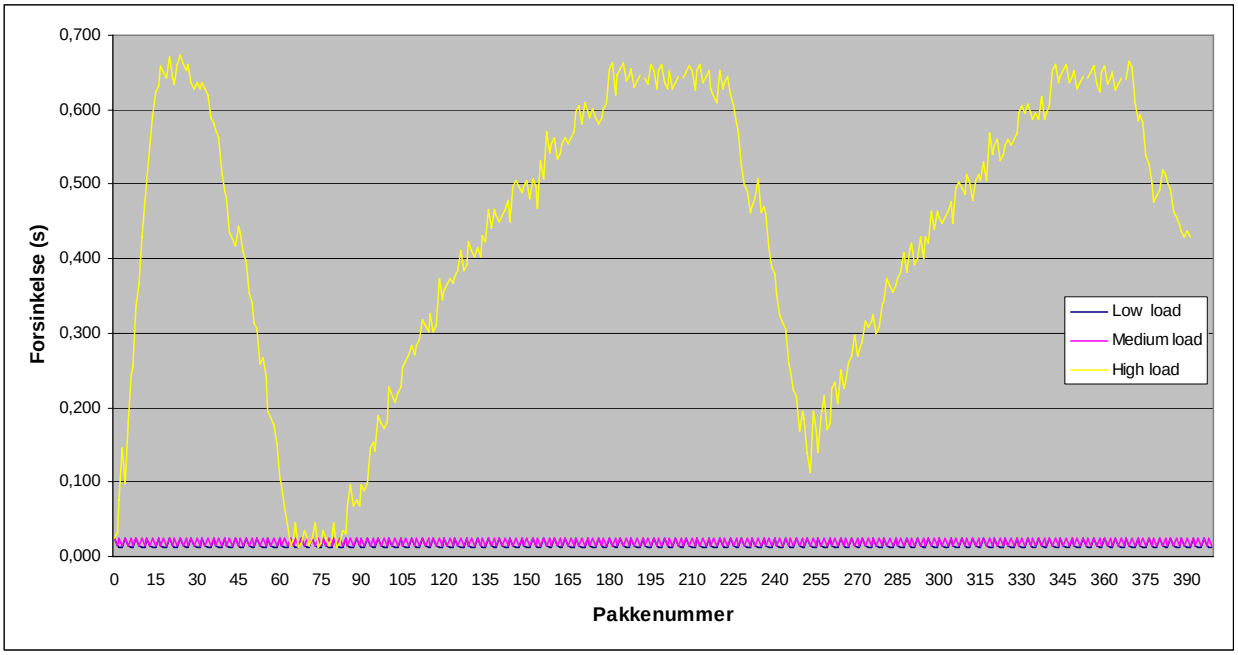
8 DEDIKERT NETT MED 256 KBIT/S LINKER

I dette scenariet har vi forutsatt at det etableres et egen nett for å støtte luftvernsystemene og at denne trafikken dermed har høyest prioritet. Eventuell resterende kapasitet kan utnyttes av andre applikasjoner. Simuleringene følger en mulig migrasjonssvei.

8.1 Ingen QoS-mekanismer

I dette alternativet er ingen tjenestekvalitetsmekanismer implementert, og det ingen prioritering av ulike pakketyper. Figur 8.1 viser forsinkelse per pakke ved ulike trafikkbelastninger. Ved lav (30% last på høyest belastet link) og medium (50% belastning på høyest belastet link) belastning består trafikken av sensorinformasjon + UDP-strømmer. Ved medium belastning er gjennomsnittlig forsinkelse 20 ms med jitter på 9 ms. Resultatene illustrerer at så lenge en har full kontroll over trafikk-kildene i et IP-nett kan en oppnå verdier for forsinkelse og jitter som ligger innenfor rimelige krav.

I simuleringen av høy belastning ble sensortrafikken og UDP-strømmene fra medium belastning beholdt, men vi la til en TCP-strøm. TCP-strømmen ble konfigurert med maksimalt mottakervindu. Denne TCP-strømmen vil øke belastningen gradvis inntil den taper pakker. Belastningen vil da bli halvert eller enda kraftigere redusert, for så å bygge seg opp igjen. I et virkelig nett vil det være mange TCP-strømmer, slik at utslagene blir mindre. I tillegg kan også mottakerbufferet begrenses for å redusere den maksimale belastningen. Dette kan imidlertid redusere utnyttelsen av ledig kapasitet.



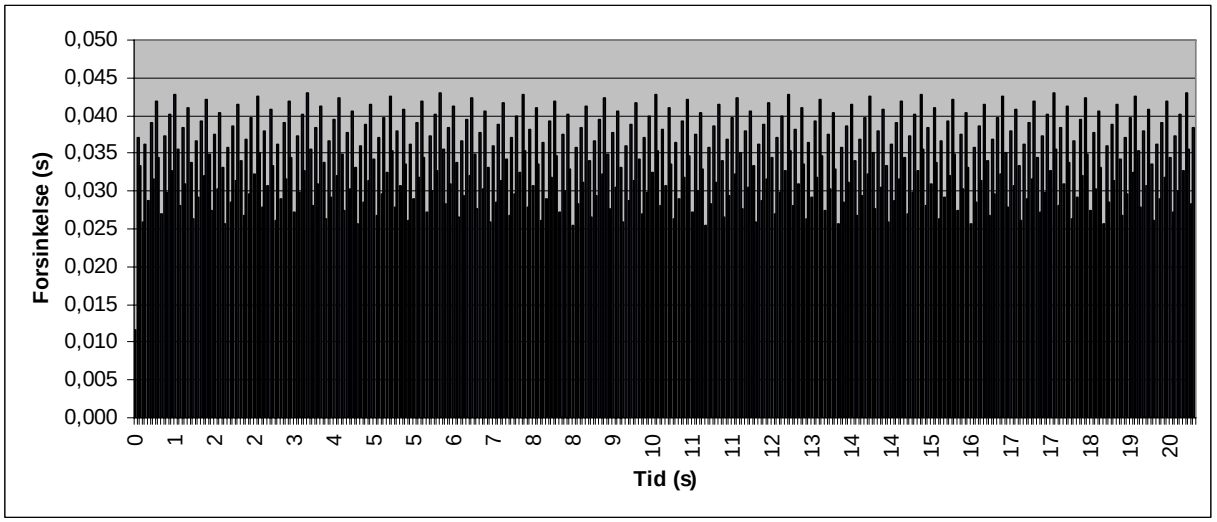
Figur 8.1 Forsinkelse for ulike trafikkbelastninger uten QoS

Trafikkmønsteret er derfor ikke helt reelt, men resultatene illustrerer to ting, 1) det velkjente faktum at i tungt belastede nett vil køtiden bli høy, og 2) i det øyeblikket en ikke lenger har full kontroll over trafikkildene så stiger risikoen for økt forsinkelse. TCP representerer mellom 60% og 70% av all Internet- trafikk (avhengig av hvor og hvordan denne måles). TCP brukes i overføring av web-trafikk og katalog- og databaseapplikasjoner. Vi forventer at den derfor vil representere en viktig del av trafikkmønstert også i dedikerte nett som ikke begrenses kun til sensorinformasjon. TCP-strømmer vil alltid søke å utnytte tilgjengelig båndbredde.

Utnyttelsesgraden til TCP-strømmene er en funksjon av bufferstørrelsen i ruterne og i endesystemene. Resultatene som er vist i Figur 8.1 representerer i så måte et ytterpunkt. Like fullt er det et faktum at dersom nettet blir gjort tilgjengelig for andre applikasjoner kan ende-til-ende forsinkelse og jitter kun garanteres dersom arkitekturen gjør det mulig å separere trafikkstrømmene gjennom bruk av ulike QoS-mekanismer.

8.2 Med QoS-mekanismer

I de neste trafikkscenariene er det brukt en standard DiffServ QoS-arkitektur. Den er konfigurert med en prioritetsklasse (EF) og en best effort-klasse (BE). Sensortrafikken blir eksklusivt merket som prioritetstrafikk, mens all annen trafikk blir merket som best effort. Igjen brukes et ekstremt scenario hvor bakgrunnstrafikken er to TCP-strømmer som periodisk vil belaste ruterne inntil bufferne er fulle og pakker tapes. For prioritetstrafikken er gjennomsnittlig forsinkelse 34 ms og med jitter på 31 ms. Resultatene som vises i Figur 8.2 viser klart at en prioritetsklasse dedikert til sensortrafikken gir en bra separasjon mellom de ulike trafikktypene og at det er mulig å kontrollere forsinkelsen og dermed også jitteret for sensortrafikken.

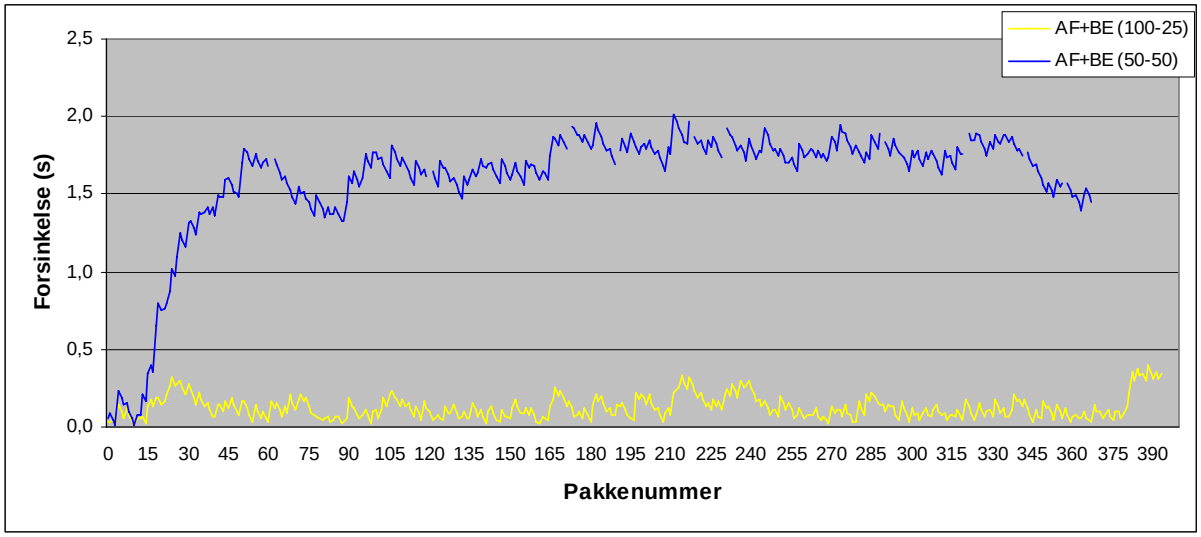


Figur 8.2 Forsinkelse for sensortrafikk rutet som prioritetstrafikk

Forsinkelsen er høyere enn den var ved et medium belastet nett. Dette skyldes en køkomponent skapt av utklokking av en BE-klasse pakke. Prioritetstrafikken vil alltid bli prioritert og skal klokkes ut først. Imidlertid, dersom en pakke med lavere prioritet allerede ligger i utklokkingsbufferet blir den ikke avbrutt, men klokkes ut på normal måte. Prioritetspakker kan i verste fall havne bak en BE-pakke som nettopp er begynt å klokkes ut i hver ruter. Metodene for å redusere denne køkomponenten er enten å bruke lokal fragmentering på linken eller å redusere pakkestørrelsen for best effort-trafikken. I simuleringene er ingen av disse metodene implementert slik at resultatene er å betrakte som konservative. Den gjennomsnittlige forsinkelsen øker med 14 ms, noe som grovt sett tilsvarer utklokking av en TCP pakke, dvs i snitt gjenstår halve TCP pakken når en høyprioritetspakke ankommer hver av kjernenettruterne.

DiffServ med en prioritetsklasse er bare en av mange mulige QoS-arkitekturer. Normalt vil det være et ønske om å kunne ha flere ulike prioritetsnivåer. Standard kommersielle rutere tilbyr normalt bare en prioritetsklasse, siden dette er hva som er definert i DiffServ-standarden. Flere prioritetsnivåer må enten tilbys gjennom spesialtilpasset utstyr, eller gjennom å utnytte Assured Forwarding-tjenesten (AF) i DiffServ. Det er definert fire ulike AF-klasser, hver med tre innbyrdes forskjellige sannsynligheter for pakkekasting. AF-tjenesten tilbyr således en større mulighet for å differensiere mellom ulike trafikktyper.

Utfordringen ved bruk av AF-klassen er å sikre korrekt konfigurasjon. I simuleringene som er gjennomført har vi benyttet en vektet round robin (WRR) skeduleringsalgoritme kombinert med en RED (Random Early Discard) mekanisme for å støtte intelligent bufferhåndtering ved overbelastning. Vektingen mellom BE- og AF-klassene ble variert. Selve trafikkmønsteret besto av sensortrafikken som var merket som AF og variable UDP-kilder som ble merket som BE. Figur 8.3 viser at ulik vekting mellom AF- og BE-klassene gir svært forskjellige resultater med hensyn på ende-til-ende forsinkelse og jitter. Ende-til-ende forsinkelsen øker fra gjennomsnittlig 0,13 s til 1,6 s når vektingen mellom de to klassene endres fra 100-25 til 50-50. Jitteret blir henholdsvis 0,38 s og 2 s for de to tilfellene.



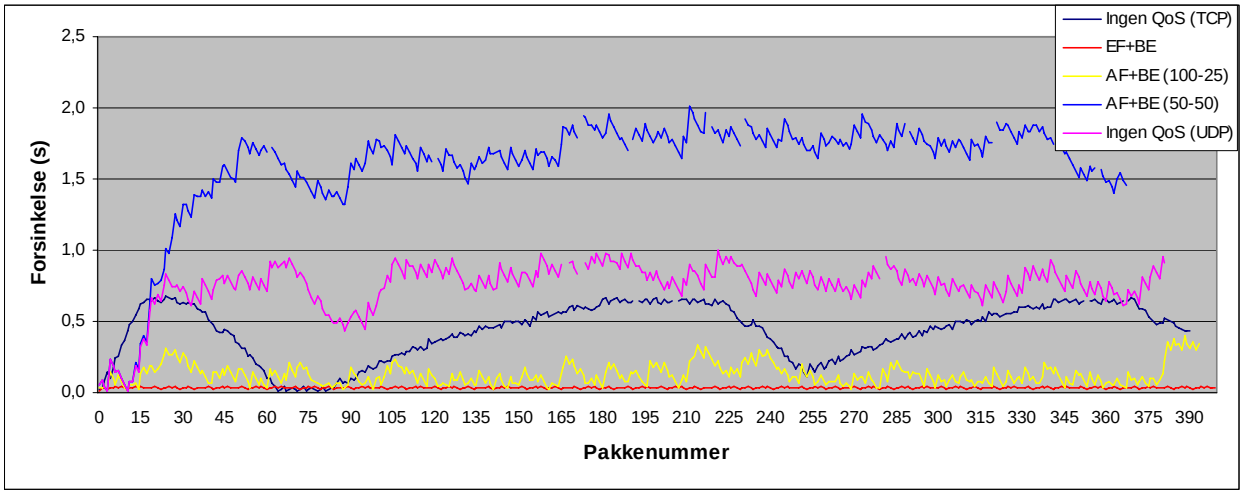
Figur 8.3 Bruk av AF- og BE-klasser med ulike vekting

8.3 Konklusjoner for dedikert nett

I eksemplet med et dedikert nett vil forsinkelse og jitter variere fra noen få titalls millisekunder til flere sekunder avhengig av forutsetningene som gjøres. Hvilket alternativ som er operasjonelt akseptabelt må avgjøres av Forsvaret. Vi kan imidlertid trekke noen konklusjoner. Dersom vi sammenligner med eksisterende TDM-baserte nettverk vil forsinkelsen for den sensortrafikken som modelleres være større enn 25 ms i et tilsvarende TADKOM nett med 64 kbit/s kanaler. Forsinkelsen i et tett kontrollert IP-nett eller et IP-nett med korrekt tilpasset QoS-arkitektur er innen samme størrelsesorden. Det er derfor mulig å designe et IP-nett hvor forsinkelsen vil være tilsvarende det den er i dag. Et IP-nett vil gi noe jitter, noe eksisterende linjesvitsjede nett ikke har. Hvorvidt jitter i størrelsesorden 10 til 30 ms er akseptabelt eller ikke er et operasjonelt spørsmål.

Simuleringene indikerer at med god kontroll over trafikkbelastningen trenger man ikke QoS-mekanismer. Vi anbefaler ikke en slik tilnærming. En av hovedideene med NbF er at nettene og applikasjonene skal løsrives. Dermed kan man nesten per definisjon forutsette at man ikke lenger har full kontroll med trafikkbelastningen. Konsekvensene av bare å addere en ukontrollert TCP-strøm er godt illustrert. En QoS-arkitektur må derfor anses som en nødvendighet.

Simuleringene illustrerer også at det eksisterer ulike alternative arkitekturer som kan resultere i forholdsvis like stor forsinkelse og jitter for sensortrafikken. Valget mellom den ene eller den andre løsningene blir et spørsmål om hvilke andre typer trafikk eller applikasjoner som også vil benytte nettet.



Figur 8.4 Sammenligning av ulike QoS-arkitekturer for et dedikert sensornett

Resultatene som er vist i Figur 8.4 illustrerer også godt at relativt små variasjoner i parametersettingene kan være avgjørende på om resultatene blir gode eller dårlige. De viser at bruk av en kombinasjon av AF og BE med en 50-50 vekting gir en dårligere tjenestekvalitet for sensortrafikken enn bruk av en ren BE-tjeneste under de samme trafikkforholdene. Dette er ikke bare en indikasjon hvor viktig selve designet av QoS-arkitekturen er. Det er også en klar illustrasjon av hvor stort problemet med konfigurering og vedlikehold av konfigureringsinformasjon vil bli et NbF-nett.

9 TAKTISK OMRÅDENETT MED 8 MBIT/S LINKER

Et 8 Mbit/s taktisk områdenett etableres for å støtte en rekke applikasjoner og luftvernapplikasjonen må derfor konkurrere om ressursene mot andre applikasjoner som kanskje har enda strengere krav til forsinkelse, jitter og pakketap. Det er derfor sannsynlig at luftverntrafikken ikke vil gis den beste tjenestekvaliteten og merkes som EF-trafikk, men heller merkes som AF-trafikk. I tillegg må den kanskje konkurrere mot andre applikasjoner som også er merket som AF-trafikk. Fokus for simuleringene i dette scenariet er derfor forsinkelse og jitter når sensortrafikken kjøres som en av mange trafikktyper som ikke er beskyttet av prioritetskøing.

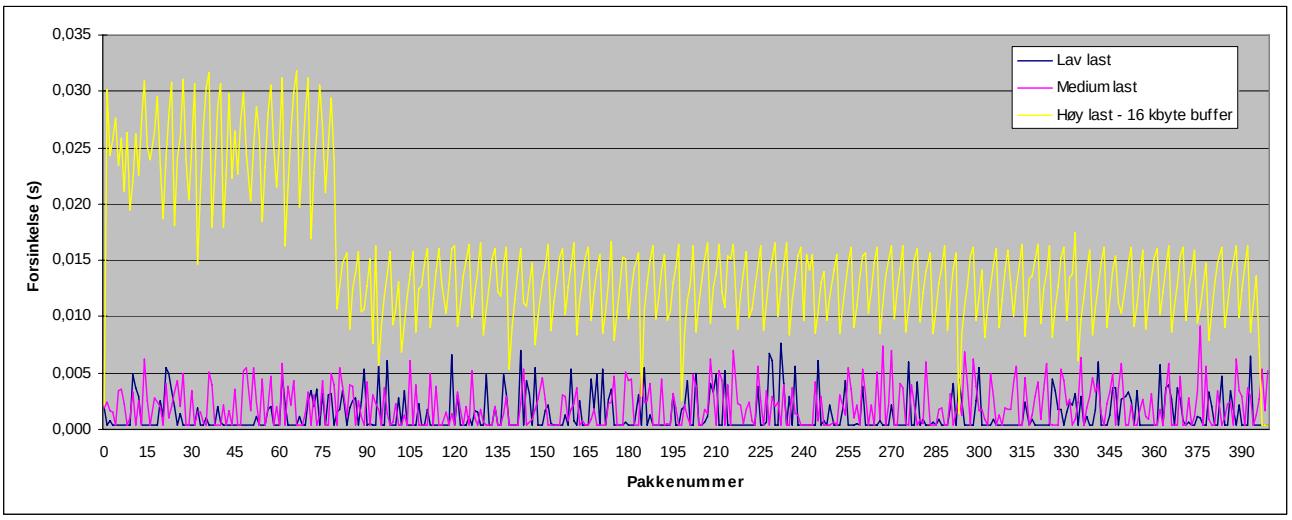
9.1 Endring i topologi

Den grunnleggende topologien er den samme, men antallet aksessnoder har økt. Dette er gjort for å reflektere at antallet kryssende trafikkstrømmer vil øke. Samtidig vil sensornodene kun være en av flere andre typer noder i nettverket.

9.2 Uten QoS-mekanismer

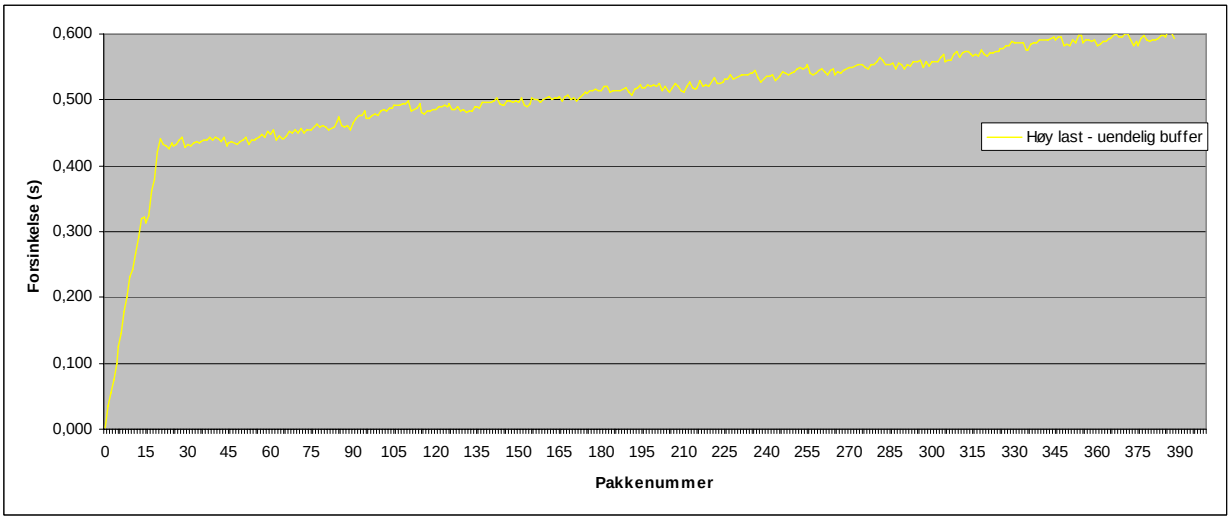
Med høykapasitetslinjer uten bruk av QoS-mekanismer blir resultatene tilsvarende som ved lavkapasitetslinker. Dersom man har full kontroll med trafikkbelastningen i nettverket er forsinkelsen og jitteret til sensortrafikken moderat. Ved lav belastning (30%) har

sensortrafikken en gjennomsnittlig forsinkelse på 1,4 ms og jitter på 7 ms. De tilsvarende tallene ved medium belastning (50%) er 2 ms forsinkelse og 9 ms jitter.



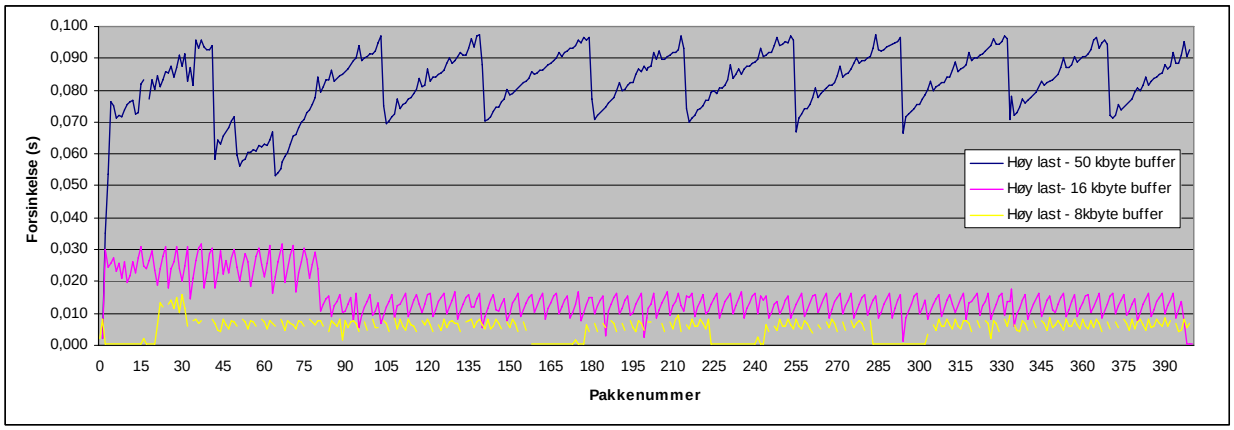
Figur 9.1 Forsinkelse ved 8 Mbit/s linjer for ulike belastninger og bufferstørrelse på 16 kbyte

I et 8 Mbit/s nettverk er det naturlig å forvente at ved full belastning vil bakgrunnstrafikken består av flere TCP-strømmer. Gitt store nok buffere vil TCP-strømmene øke antallet utestående pakker lineært inntil hver strøm har nådd full vindusstørrelse. Dette medfører at forsinkelsen for sensortrafikken også vil øke lineært. Dette er vist i Figur 9.2.



Figur 9.2 Forsinkelse for 8 Mbit/s linker ved høy trafikkbelastning og uendelig buffer

Som tidligere nevnt kan forsinkelsen kontrolleres ved å begrense bufferstørrelsen i ruterne. Når bufferne er fulle, vil pakker bli kastet og trafikkpåtrykket fra TCP-strømmene reduseres. Figur 9.3 viser forsinkelsen for ulike bufferstørrelser ved høy belastning i nettet.



Figur 9.3 Effekten av ulike bufferstørrelser på forsinkelsen ved høy belastning i nettet

Sagtannmønsteret i kurvene reflekter metningskontrollmekanismen i TCP. Avhengig av bufferstørrelsen i ruteren er forsinkelsen ved relativt store buffer (50 kbyte) ca 82 ms i gjennomsnitt. Ved medium (16 kbyte) og små (8 kbyte) bufferstørrelser er gjennomsnittsforsinkelsen henholdsvis ca 15 og 5 ms, men jitteret i gjennomsnitt er 31 og 16 ms. Dette oppnås kun ved å akseptere pakketap. Når bufferne er fulle vil pakker bli kastet. TCP oppfatter dette som indikasjon på metning i nettet og reduserer trafikkpåtrykket. Med 50 kbyte buffer er tapet 0,5%, mens den for 8 kbyte buffere er pakketapet på 11,5%. Pakketapene blir såpass moderate på grunn av den selvregulerende mekanismen til TCP. Avhengig av operasjonelle krav kan en avveie forsinkelse imot pakketap ved å kontrollere bufferlengden i ruterne.

Denne delen av simuleringsscenariet er tatt med for å illustrere at selv uten QoS-mekanismer er det mulig å oppnå moderate forsinkelser selv under høy belastning. Forutsetningen er at hovedvekten av trafikk i nettet er TCP og en moderat kostnad i form av pakketap. Vi tror ikke dette er en akseptabel arkitektur siden forutsetningen den hviler på vil være veldig begrensende. Det er også sannsynligvis en operasjonell begrensning på hva som er en akseptabel pakketapsrate. Scenarioet er tatt med for å vise bredden av designtiltak som kan brukes i et nettdesign.

9.3 Med QoS-mekanismer

Antagelsen for scenariet med 8 Mbit/s linker er at mange ulike applikasjoner utnytter nettet. Standard rutere kan forventes å støtte en prioritetskø. Dersom kun sensortrafikken allokeres til denne vil en kunne oppnå en bedre grad av isolasjon mellom sensortrafikken og annen trafikk enn hva som ble vist i scenariet med 256 kbit/s linker. Isolasjonen blir bedre siden annen trafikk kun kan forsinke høyprioritetspakker med maksimalt utklokkingstiden for en påbegynt lavprioritetspakke. Når utklokkingshastigheten på pakker er 32 ganger høyere, så vil interferensen fra annen trafikk bli lavere. Forsinkelse og jitter for sensortrafikk blir derfor lavere over høykapasitetslinker enn over lavkapasitetslinker dersom denne typen trafikk anses som den viktigste.

Fra et perspektiv der kontroll av forsinkelsen er viktig er bruk av AF-klassene vanskeligere. Her

får hver klasse tildelt en viss andel av utgående båndbredde, og intern køing og interferens mellom klassene kan skape forsinkelser og jitter. Styrken med en slik arkitektur er at mange ulike klasser kan brukes og en er mindre avhengig av hva ruterleverandøren har implementert. For å vise et bedre spektrum av designalternativer ble to ulike AF klasser + best effort brukt i simuleringen. Sensortrafikken ble lagt i AF2-klassen som også ble tildelt mindre andel av ressursene enn AF1. Ulik vekting mellom AF1, AF2 og best effort (BE) ble simulert.

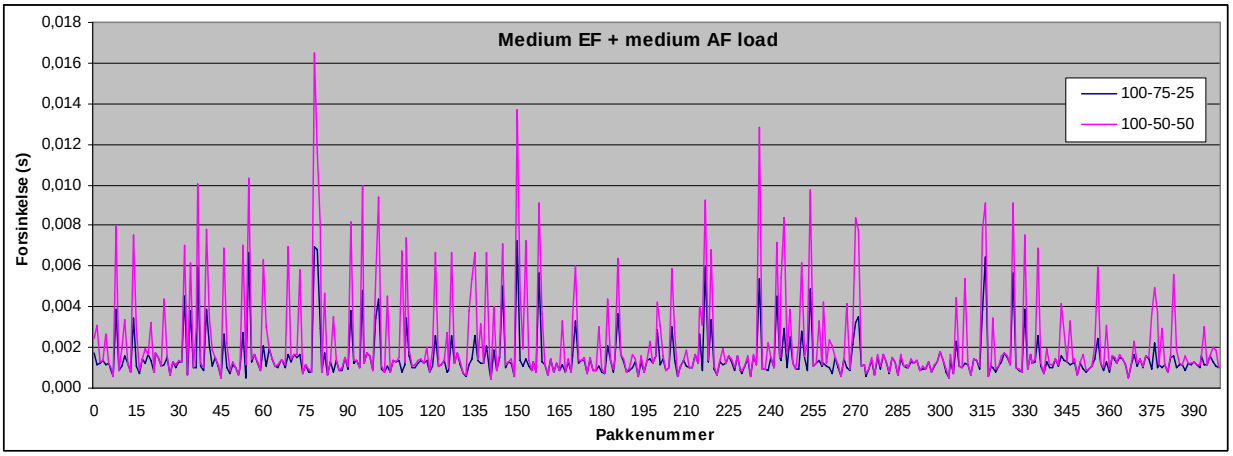
9.3.1 Trafikkmatrise

Tre ulike typer trafikkmatriser ble brukt, en med medium last og to med høy last. Hensikten med de ulike trafikkmatrisene er å vise 1) at sensortrafikken fint kan håndteres som mediumprioritert trafikk i høykapasitetsnett, 2) QoS-parameterne har innvirkning, og 3) adgangskontroll er fortsatt en viktig.

9.3.2 Medium trafikkbelastning

Ved medium trafikkbelastning blir AF1 (klassen som får høyest andel av ressursene) lastet opp med 2 Mbit/s over den mest belastede linken. Lasten i denne klassen er av TDM-typen, dvs konstant bitrate trafikk med fast pakkelengde. Lasten i AF2-klassen, er 612 kbit/s hvorav sensortrafikken utgjør 32 kbit/s over den mest belastede linken. Sensortrafikken er konstant, mens resten av AF2-klassen er variable både med hensyn på pakkelengde og pakkerate. BE-klassen er belastet med flere TCP-strømmer som forsøker å bruke så mye kapasitet som mulig.

Vi registrerer ikke pakketap i sensortrafikken med denne belastningen. Forsinkelsen er i gjennomsnitt ca 1,5 ms med jitter på 7 ms. Gjennomsnittelig forsinkelse og jitter øker når AF2 klassen får mindre vekt, se Figur 9.4.



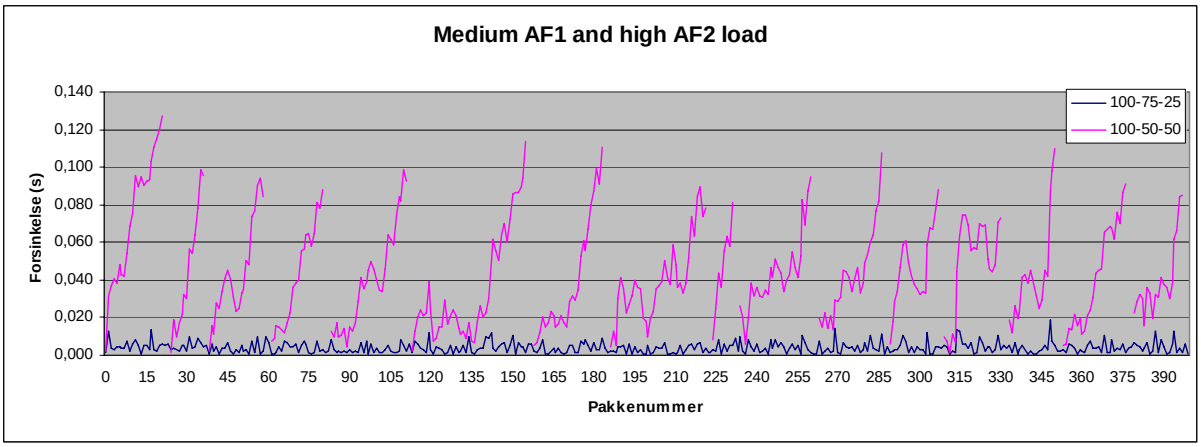
Figur 9.4 Forsinkelse per pakke ved ulik vekting mellom AF1, AF2, og BE trafikk (vektingen er henholdsvis 100-75-25 for blå kurve og 100-50-50 for rosa)

9.3.3 Høy belastning med omfordeling mellom klassene

Dette eksemplet vil vise effekten av å fordele ledig kapasitet mellom klasser. Belastningen i AF2-klassen økes til 2,9 Mbit/s over den mest belastede linken. Sensortrafikken utgjør kun 32 kbit/s, slik at forsinkelse og jitter reflekterer mengden av den resterende variable trafikken. BE-

klassen lastes opp med TCP-trafikk som vil ekspandere og forbruke all tilgjengelig kapasitet. Med denne trafikkmatrisen vil ikke høyprioritets trafikken bruke alle de tildelte ressursene.

Når AF2-klassen har en vekt på 75, tilsvarer dette at grovt 3 Mbit/s av linkbåndbredden tildeles denne klassen. Sensortrafikkens forsinkelse skyldes da intern køing på grunn av at resten av trafikken i klassen er svært variabel. Forsinkelsen er ca 4 ms med 18 ms jitter, Figur 9.5. Det er ikke pakketap siden klassen har fått allokert mer båndbredde enn hva behovet tilsier.



Figur 9.5 Forsinkelse per pakke ved ulik vekting (hneholdsvis 100-75-25 for blå kurve og 100-50-50 for rosa) mellom AF1, AF2, og BE-trafikk for medium AF1 og høy AF2 belastning

Når vekten til AF2-klassen endres til 50, allokeres grovt sett 2 Mbit/s av båndbredden til denne klassen. Trafikkpåtrykket overskrider dette og resultatet blir køing. Totalt sett er det ca 2 Mbit/s i "ubrukt" AF1-båndbredde. Denne fordeles mellom AF2- og BE-klassen i henhold til vektingen. Pakkene køes lengre og den gjennomsnittlige forsinkelsen for sensortrafikken øker til 43 ms med 126 ms jitter. Etter hvert som køene fylles opp vil pakker kastes, og pakketapsraten kommer opp i 7,5%. Pakketapet vil ikke bli jevnt fordelt siden mesteparten av trafikken i klassen er variabel og dermed vil også pakketapene få en variabel fordeling. Dette kan få mer alvorlige operasjonelle implikasjoner enn hva selve tapsraten skulle tilsi.

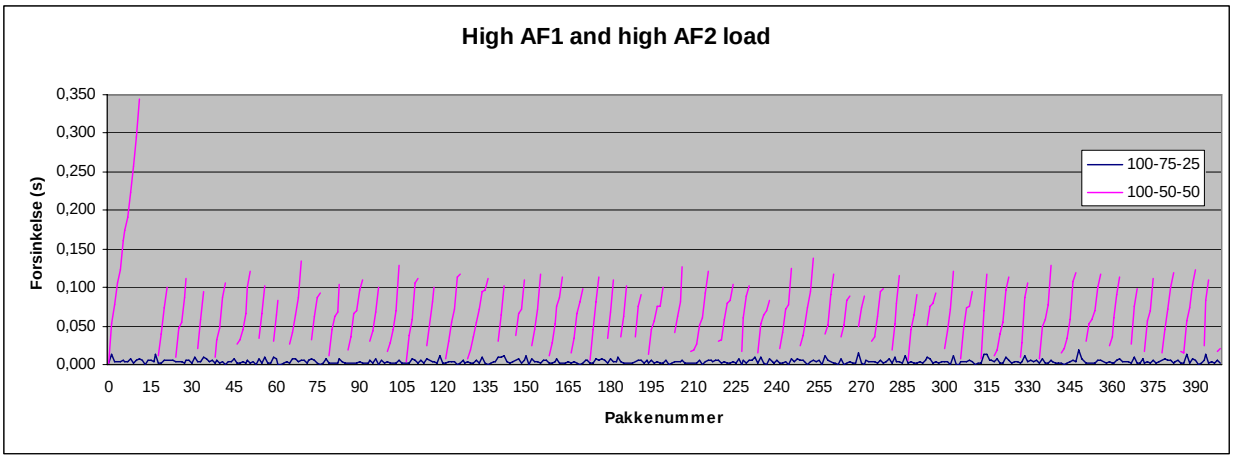
Simuleringene viser at WRR-skedulering (Weighted Round Robin) gir god isolasjon mellom klassene så lenge trafikkpåtrykket er i samsvar med vektingen. Dersom det er høyere trafikkpåtrykk en hva som korresponderer til vekting er ikke denne metoden spesielt egnet til fordeling av de tilgjengelige frie ressursene.

9.3.4 Høy belastning uten mulighet for omfordeling

Det siste eksemplet representer et tilfelle hvor linkene er lastet helt opp. AF1-klassen, som tildeles mest ressurser, er lastet opp med 4 Mbit/s TDM-type trafikk. AF2-klassen hvor sensortrafikken er allokert, fortsetter med en last med 2,9 Mbit/s, mens BE-klassen fortsetter å kjøre TCP. Med denne trafikkfordelingen vil det ikke finnes ledig kapasitet og resultatene aksentuerer betydningen av å sette vekting på de ulike klasser riktig.

Når AF2-klassen har en høy nok vekt (dvs tilstrekkelig ressurser) i forhold til påtrykket (vekt 100-75-25) er forsinkelsen 4 ms med 18 ms jitter, Figur 9.6. Det oppstår ikke pakketap. Dette viser den gode isolasjon som oppnåes mellom klassene når vektingen tilsvarer påtrykket.

Når AF2-klassen får lavere vekt samtidig som det ikke er ledige ressurser som kan tas fra høyere klasser, blir sensortrafikken køet og/eller kastet. Dette er reflektert i en forsinkelse på 68 ms og 342 ms jitter. Pakketapet er under dette scenariet 33,5%.



Figur 9.6 Forsinkelse per pakke ved ulik vekting mellom AF1, AF2, og BE-trafikk for høy AF1- og høy AF2-belastning. Vektingen er gitt i legend.

Scenariet illustrerer sårbarheten til trafikk i en klasse hvor påtrykket overskrider den andelen av ressursene som er tildelt. Siden den andre AF2-trafikken er variabel blir også pakketapet variabelt.

9.4 Konklusjon for et bredbånds taktisk nett

Sensortrafikken er ikke avhengig av å bli definert som høyprioritetstrafikk for å oppnå akseptabel ytelse (dvs lavere enn i det dedikerte nettet) i et bredbåndsnett. Sensortrafikken er forutsigbar og kan lett isoleres fra annen trafikk gjennom å bli tildelt sin egen DiffServ-klasse. Det er ikke nødvendig å bruke en streng prioritering mellom ulike typer trafikk. Fordelen er at en slik arkitektur kan tilpasses kommersielt utstyr. Det er nødvendig med en diskusjon om hvilke typer trafikk som skal prioriteres, men det viktigste er valget av parametre. Dersom klassen med sensortrafikk blir underfordelt, vil økt pakketap og forsinkelse bli konsekvensene. Fordi sensortrafikken er meget forutsigbar er ikke feil parametersetting en reell risiko, men det utgjør heller et potensielt nettverk-mangement problem forårsaket av operatørfeil.

Simuleringen illustrer behovet for trafikkontroll (admission control) under en DiffServ-arkitektur. Uten kontroll av trafikkpåtrykket, enten direkte eller gjennom begrensninger i applikasjonene, er det alltid en risiko for overbelastning. Overbelastningen kan isoleres til de trafikkstrømmene som er allokert til samme klasse, men en viss risiko vil fortsatt eksistere. Simuleringene illustrerer også at så lenge hovedvekten av trafikken bruker TCP, vil TCP-metningskontroll kombinert med nøye valg av bufferlengder i ruterne gi muligheter for å

kontrollere forsinkelse og jitter.

10 DYNAMISK RUTING

For å velge ruter gjennom nettet benyttes en standardisert rutingprotokoll; OSPF (Open Shortest Path First) 32. Den velger rute hovedsakelig basert på antall ruterhopp og veien med det laveste antall ruterhopp prioriteres. Dersom denne feiler vil protokollen prøve å finne en annen vei med enten like mange eller flere ruterhopp.

I de foregående scenariene har vi fokusert på forsinkelse og jitter i et stabilt nett. Her ser vi på konsekvensene ved feil i nettet, enten feil som skyldes at linker eller rutere feiler eller at noder flytter seg og dermed fører til endringer i topologien. Begge tilfellene vil føre til at trafikken automatisk rerutes. Dynamisk reruting fører til at all trafikken over en feilet link eller ruter finner en ny vei. Denne kan potensielt bestå av flere hopp og forsinkelsen vil øke og dermed også jitteret i en overgangsperiode. Rerutingen trigges av at en feil oppdages, informasjon om dette distribueres til de andre nodene i nettet og en ny rute beregnes. Denne prosessen tar noe tid og vil nødvendigvis føre til tap av pakker.

10.1 Referansetopologier og feilscenario

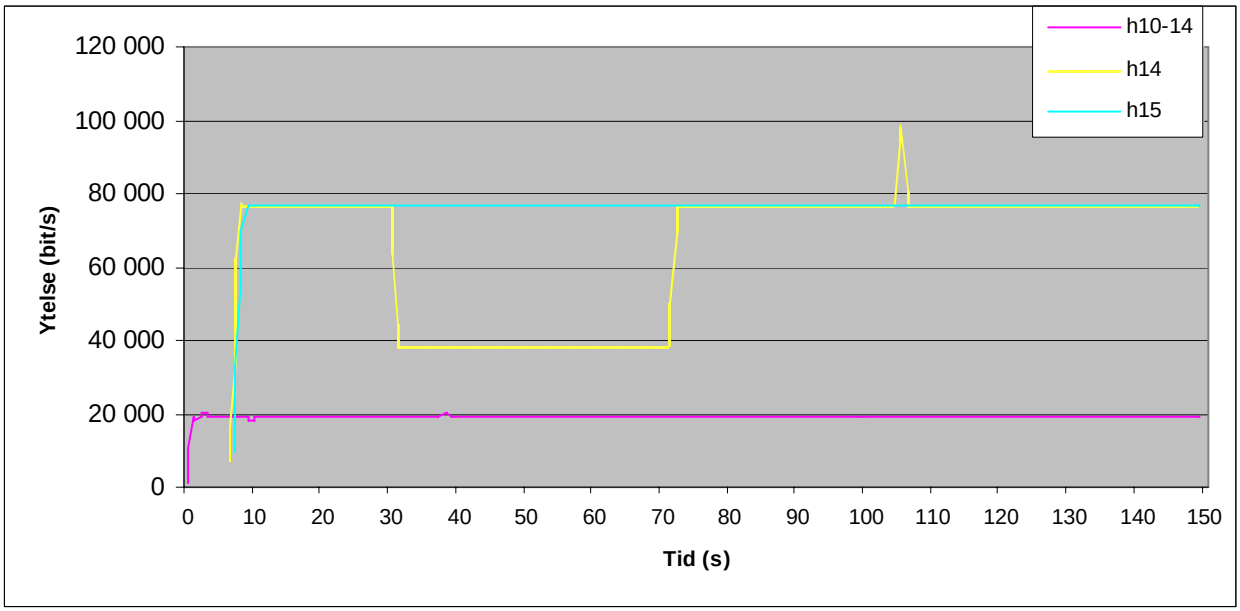
Reruting er gjort med utgangspunkt i samme referansetopologi som i Figur 7.1. Linkene er 256 kbit/s og QoS-oppsettet består av en EF-klasse for sensortrafikken og en BE-klasse for TCP-trafikk.

10.2 Resultater

Figur 10.1 viser trafikken fra h10 til h14 og h15 samt den totale mengden trafikk som ankommer h14 og h15.

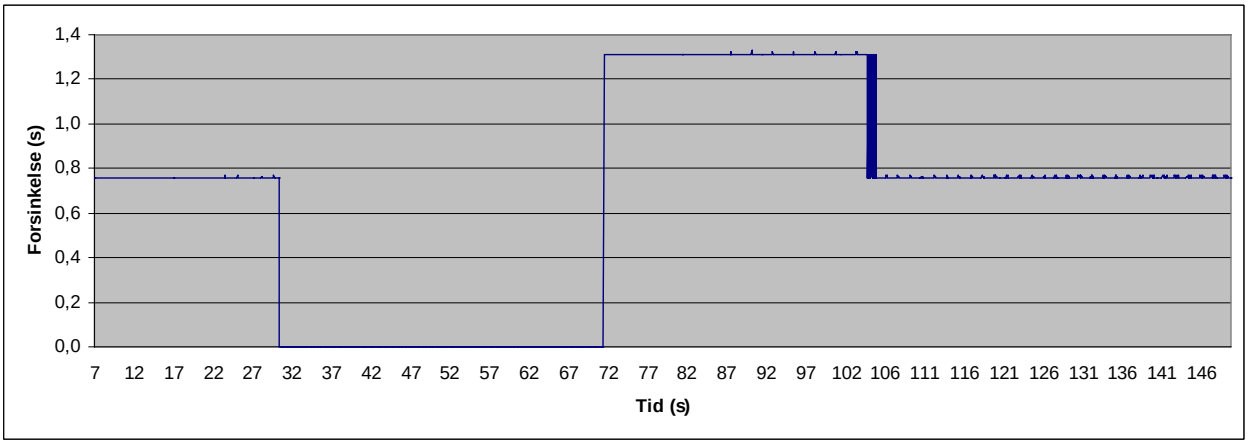
Resultatene viser at det tar noe tid før rutene etableres og trafikken begynner å flyte, ca 6s. Etter 30s feiler linken mellom n0 og n3 og trafikken mellom h10 og h14 kommer ikke fram noe som fører til lavere trafikk inn til h14. OSPF detekterer linkfeilen og etter ca 40 sekunder er en alternativ rute etablert via n2.

Tiden det tar å etablere en alternativ rute tilsvarer at tre Hallo-meldinger med 10s mellomrom ikke besvares pluss den tiden det tar å informere de andre nodene og beregne en ny rute. Når ruten mellom n0 og n3 igjen blir operativ føres trafikken tilbake igjen.



Figur 10.1 Trafikken fra h10 til h14 og h15 samt totalt mottatt trafikk hos h14 og h15.

Den alternative ruten via n2 har ett ruterhopp mer enn den opprinnelige ruten og dette bidrar til å øke ende-til-ende forsinkelsen til trafikken mellom h10 og h14. Som Figur 10.2 viser så reduseres forsinkelsen igjen når den opprinnelige ruten igjen etableres.



Figur 10.2 Forsinkelsen mellom h10 og h14

Figur 10.1 viser at ytelsen til trafikken mellom h10 og h14 øker rett etter at linken mellom n0 og n3 kommer opp igjen. Grunnen til dette er at forsinkelsen på den nye ruten er lavere enn på den gamle slik at en del av pakkene som sendes langs den nye ruten ankommer før de rerutede pakkene og dette skaper en midlertidig økt ytelse, men også kortvarige variasjoner i forsinkelsen. En del av pakker ankommer utenom sekvens og disse må stokkes om for å hindre pakketap. Ved bruk av standard transportprotokoller gis pakkene sekvensnummer slik at omstokking skal være mulig, men det forutsetter også at mottakeren har tilstrekkelig lagringskapasitet. For enkelte applikasjoner vil sent ankomne pakker ikke lenger være av interesse og disse kan da kastes.

10.3 Konklusjoner

Fordelen med automatisk reruting er at det ikke er behov for at nettadministratoren tar aksjon ved feil eller bortfall av linker og rutere i nettet. Tiden det tar fra en feil oppdages til en ny rute er operativ avhenger i hovedsak av følgende parametere:

- Hallo-intervallet i OSPF er den viktigste parameteren. Den er typisk satt til 10 sekunder, men kan konfigureres til lavere eller høyere verdier. Et mindre Hallo-intervall vil føre til økt overhead, men også bidra til at feil oppdages raskere.
- Størrelsen på nettet vil påvirke hvor raskt informasjon om topologi-endringer spres.

Reruting av trafikken kan føre til endringer i tjenestekvaliteten siden karakteristikken til den alternative ruten kan være annerledes enn karakteristikken til den opprinnelige ruten. I vårt tilfelle førte endringene til at forsinkelsen økte siden den alternative veien hadde ett ekstra ruterhopp.

11 TIDSSYNKRONISERING OG ESTIMERING AV FORSINKELSE I IP-NETT

Flere anvendelser er avhengig av kunnskap om absolutt tid slik at de kan synkronisere handlinger eller filtrere ut den mest oppdaterte informasjonen. I NbF-sammenheng vil sensorer og effektorer samhandle på tvers av informasjonsinfrastrukturen. Dersom effektorene skal nyttegjøre seg sensorinformasjonen eller agere på bakgrunn av den må de ha et forhold til informasjonens alder. Et eksempel på dette kan være systemer hvor effektorer mottar lokasjonskoordinater på bevegelige mål fra ulike sensorer. For at effektorene skal kunne beregne den eksakte lokasjonen til målet er det nødvendig å vite leveringsforsinkelsen.

Eksisterende Eurocom-nettverk har tidsforsinkelse som en del av innholdet i hver pakke. Pakken blir tidsstemplet både når den mottas av et nettelement og når den forlater nettelementer. Delta mellom disse tidspunktene blir lagt til forsinkelsen i pakken. Vi antar at utklokkingshastigheten for pakken også legges til den totale forsinkelsen når pakken forlater nettverksgrensesnittet.

Fordelen med denne metoden er at klokkene i kildene, transittsvitsjene, og mottakerne ikke trenger å være synkrone. Hver node legger kun til forsinkelsen til den interne prosesseringen. Alternativet ville kunne være å bruke synkrone klokker hos kilde og mottaker og kun tidsstemple pakken når den sendes ut, slik det gjøres i sanntids transportprotokollen RTP (9).

11.1 Tidssynkronisering

GPS-satellittene gir en ekstern referanseklokke. Det er imidlertid begrensninger i hvor GPS-signalene kan mottas. Selve systemet ligger også utenfor egen kontroll og det er mulig å jamme signalene. GPS vil inngå som en komponent i et ”blue force” tracking system, men et tidssynkroniseringssystem kan ikke kun baseres på at GPS-signalet vil være tilgjengelig. Det bør være i stand til å nyttiggjøre seg GPS-signalet dersom det blir tilgjengelig for kortere eller lengre tid. Innenfor rammen av sensor-effektor, er det ikke et absolutt behov for en streng synkronisering av tid opp imot absolutt tid. Det er tilstrekkelig kun å synkronisere klokkene på

effektorene opp imot sensorene. Kravet kan formuleres som et behov for å synkronisere opp imot en felles tid, som nødvendigvis ikke sammenfaller med UTC.

Network Time Protocol (NTP) (10) er funksjonelt i stand til å utnytte ulike tidskilder. Den synkroniserer klokker opp imot en felles beste referansetid. Den er således en egnet plattform for videre utvikling av et tidssynkroniseringssystem.

11.1.1 Real-time Transfer Protocol (RTP)

I IP-arkitekturen er det Real-time Transfer Protocol (RTP) (9) som brukes for overføring av tid. RTP-protokollen har et tidsstempel tilpasset Network Time Protocol (NTP) og hver pakke tidsstemples når den behandles i transportlaget. Dersom klokkene hos mottaker og på avsender er synkrone kan forsinkelsen estimeres. Nøkkelen ligger derfor i en synkronisering av klokkene i de ulike nodene.

11.1.2 Network Time Protocol (NTP)

NTP (10) er en protokoll beregnet for tidssynkronisering av klokker opp imot et hierarki av tidstjenere. NTP er en standardisert protokoll og inngår som en standardkomponent i de fleste operativsystemer som leveres. Protokollen har vært brukt siden midten av nittitallet og mekanismene og filtrene er blitt tilpasset over tid.

NTP korrigerer offset og klokkedrift gjennom filtrering av tidsstempel fra tidstjenere. Dette krever at nettets forsinkelse og jitter kan estimeres. Klientene sender derfor ut probepakker og på bakgrunn av historien korrigeres offset og frekvens på den lokale klokken. Probepakkene er små (76 bytes) og kan kombineres med multicast fra tidstjenerne. Forutsetningen er imidlertid at probepakker benyttes innimellom for å estimere forsinkelse og jitter. Fordi protokollen har blitt tilpasset over lang tid finnes en rekke funksjoner og filtre for å fange opp avvik og endringer i forsinkelsen. Forventet nøyaktighet ligger i ms området innenfor lokalnett og i rundt 20 ms over WAN-nett.

11.2 Utfordringer

NTP er en stabil protokoll. Den er tilpasset et nettverk hvor tidstjenerne og klientene er oppe og tilgjengelig i flere dager av gangen. For å minske belastningen på servere og tilknytningskostnader over telefonlinjer er det lagt vekt på å minimere antallet probepakker.

Et taktisk nettverk vil representere et helt annet scenario. Vi kan forvente at både tidstjenere og klienter vil være tilknyttet i relativt korte tidsperioder av gangen samtidig som nettets topologi endres oftere. Samtidig er båndbredden i nettene begrenset, slik at belastningen på tidstjenere også må begrenses. Pakkene fra for eksempel en radarsensor kan ut fra et NTP synspunkt erstatte multicasting av tidsstempler. De går periodisk og med kjent frekvens. De representerer derfor et tidsstempel og kan brukes til å estimere jitter. Probepakker for å estimere forsinkelse er fortsatt en nødvendighet.

I et taktisk nett er det mulig å tilpasse ressursprioritering for å optimalisere kvaliteten på tidssynkroniseringen. Probepakkene er små og sendes med lav frekvens. Konsekvensen av å prioritere slike pakker høyt er derfor små. Derimot vil en høy prioritet minimalisere forsinkelsen og jitteret, siden disse komponentene i all vesentlig grad skyldes køing i bufferne. Tidssynkroniseringen vil derfor kunne påvirkes av QoS-regimet som brukes og man bør derfor sikre at alle pakkene som brukes som probepakker benytter samme tjenesteklasse.

NTP kan ikke benyttes direkte i et taktisk system fordi protokollen må tilpasses det taktiske miljøet. Det er tre områder en slik tilpasning må foregå: 1) tilpassing av parametere for hyppig probing, 2) sikring imot manipulering og 3) automatisk konfigurering for å utnytte varierende ressurser.

Dette er et arbeid som alt er initiert av det amerikanske forsvaret. Sitat fra beskrivelsen til et pågående prosjekt ved University of Delaware (http://www.eecis.udel.edu/~emills/ntp.html): *"In order to provide specific accuracy and reliability requirements, NTP requires configuration engineering specific to each time server and client site. However, in a tactical network subject to damage and repair, as well as a widely deployed real-time network such as the Defense Simulation Internet (DSI), manual configuration engineering is not acceptable. Our research effort is designed to develop an autonomous configuration capability for NTP Version 4 using multicast methods to achieve diversity and redundancy, as well as cryptographically secure source discovery."*

11.3 Konklusjoner

NTP-protokollen bør skreddersys til et tidssynkroniseringssystem for taktiske nett. Med mindre tilpasninger kan NTP brukes direkte, men det vil være funksjonelle svakheter innen dynamisk konfigurering. Det er derfor naturlig å implementere NTP gjennom å bruke en to-skritts strategi. En initiell deployering bør enten avgrenses til noen serverer eller noen nett, hvor NTP ikke er det eneste systemet for tidssynkronisering. Dette initielle systemet vil hovedsakelig ha som funksjon å skaffe operasjonell erfaring, slik at en står bedre rustet for å vurdere de tilpasninger som vil bli tilgjengelige til NTP-protokollen for brukt i taktiske nett.

Det er ingen grunn til å bruke mangel på tidssynkronisering som argument for manglende utbygging av taktiske IP-nett. Det er teknisk mulig å oppnå tidssynkronisering ned på 10-20 ms. Den foreløpige svakheten er innen graden av dynamikk på tidssynkroniseringstjenere og på sikkerhetsløsningen i et dynamisk miljø. En tilsvarende mangel på evne til å takle dynamiske endringer finnes i en rekke av de nett som er allerede er i bruk.

De funksjonelle svakheter innen dynamisk konfigurering av NTP representerer ikke fundamentale problemer. Hovedutfordringen er å velge de best egnede mekanismene ut fra et sett av velkjente løsninger.

Bruk av NTP bør først avgrenses til et system. Et initielt system bør derfor bygges rundt for eksempel luftvernssensorer. Et slikt system vil ikke kunne utnytte andre tilfeldige tidskilder, ei

heller tidssynkronisere andre noder. Ut fra erfaringsdata er det ingen grunn til å betvile at et slikt system vil gi en tidssynkronisering med noen få titalls millisekunder. En slik løsning kan på et senere tidspunkt tilpasses en mer generisk NTP-løsning.

12 OPPSUMMERING

Rapporten har to mål. Det første er estimering av forsinkelse og jitter i typiske taktiske nett. Disse tallene må så vurderes om de er operasjonelt akseptable ut i fra hvilke krav applikasjonene stiller til nettet. To ulike typer nett er estimert; et dedikert sensornettverk med moderat båndbredde (256 kbit/s), og et typisk taktisk områdenett med 8 Mbit/s linkbåndbredde. I den første typen antar vi at luftvernsensorer er den prioriterte applikasjonen. I et bredbånds taktisk områdenett gjør vi ikke denne antagelsen.

Gjennom simuleringer viser vi at det er mulig å oppnå en ende-til-ende forsinkelse omtrent på nivå med hva som er utklokkingshastigheten i dagens TADKOM-løsning. Derimot kan ikke jitter unngås i et pakkesvitsjet nett når belastningen ikke kontrolleres. Det er mange ulike nettdesign som gir jitter i størrelsesorden noen få titalls millisekunder. Hvorvidt dette er akseptabelt må vurderes operasjonelt.

Rapportens andre mål er å vise alternative designforslag med korresponderende ytelse. I et dedikert nett er det mulig å gi sensortrafikk forsinkelse og jitter i størrelsesorden noen titalls millisekunder. Dette krever enten meget god kontroll med belastningen eller en prioritering av sensortrafikken. Det siste kan oppnås med standard COTS-rutermekanismer. I et høyhastighets taktisk områdenett kan tilsvarende estimater også oppnås uten å gi sensortrafikken høyeste prioritet. Mekanismene som brukes for å støtte standard DiffServ-klasser gir gode nok garantier. Forutsetningen er at parametrene settes ut fra rimelige anslag av forventet last. Simuleringen viser klart også konsekvensene av enten feilsetting av disse parameterne eller feil estimering av trafikkpåtrykket.

For å komplettere bildet er ytelsen til en standard rutingprotokoll også simulert. Simuleringen gir et konservativt estimat, siden det ikke forutsettes noen knytning mellom feilrapportering på nettverksgrensesnittet og selve rutingprotokollen. Dersom en slik knytning finnes kan rerutingstiden bli svært kort. Uten en knytning er rerutingstiden en funksjon av frekvensen til Hallo-pakkene.

Dagens system gir mulighet for å estimere pakkeforsinkelsen. En tilsvarende funksjonalitet kan implementeres i et IP-nett ved hjelp av standard transportprotokoller og tidssynkrone noder. NTP er en velprøvd Internett-standard for tidssynkronisering som gir tidsavvik på noen få titalls millisekunder for noder på samme subnett. Selv om protokollen er stabil, bør den tilpasses forholdene i et taktisk nett når det gjelder administrasjon, sikkerhet, dynamikk og tilpasningstid. Et slikt arbeid er igangsatt i USA, slik at mangel på tidssynkronisering ikke vil være et reelt hinder for migrasjon til et IP-basert taktisk nett.

Kort oppsummert så finnes det flere ulike nettdesign som gir ende-til-ende forsinkelser og jitter som kan være operasjonelt akseptable. Simuleringene gir kun et grovt estimat, men de er et grunnlag for videre arbeid knyttet opp imot et operasjonelt nettdesign.

Litteratur

(1) Clark, D.D. (1988): The Design Phlisophy of the DARPA internet Protocols In: *Computer Communication Review*, , 106-114.

(2) V. Bollapragada, C. Murphy and R. White (2000): inside Cisco IOS Software Architecture, Cisco Press, ISBN: 1578701813

(3) S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss (1998): An Architecture for Differentiated Services, RFC 2475.

(4) K. Nichols et al (1998): Definition of the Differentiated Services Field (DS-Field) in the IPv4 and IPv6 Headers, RFC 2474.

(5) B. Davie et al (2000): An Expedited Forwarding PHB (Per-Hop Behavior), RFC 3246.

(6) J. Heinanen et al (1999): Assured Forwarding PHB Group, RFC 2597.

(7) R. Braden, D. Clark, S. Shenker (1994): Intergrated Services in the Internet Architecture, RFC 1633.

(8) J. Moy (1998): OSPF Version 2, RFC 2328 (STD 54).

(9) H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson (2003): RTP: A Transport Protocol for Real-Time Applications, RFC 3550.

(10) D. L. Mills (1989): Internet Time Synchronization: The Network Time Protocol, RFC 1129.

(11) Øivind Kure, Ingvild Sorteberg (2004): Network Architecture for Network Centric Warfare, FFI/RAPPORT-2004/01561, Ugradert