Uke 11 - gruppe

INF3190

Dagens mål

- Metningskontroll

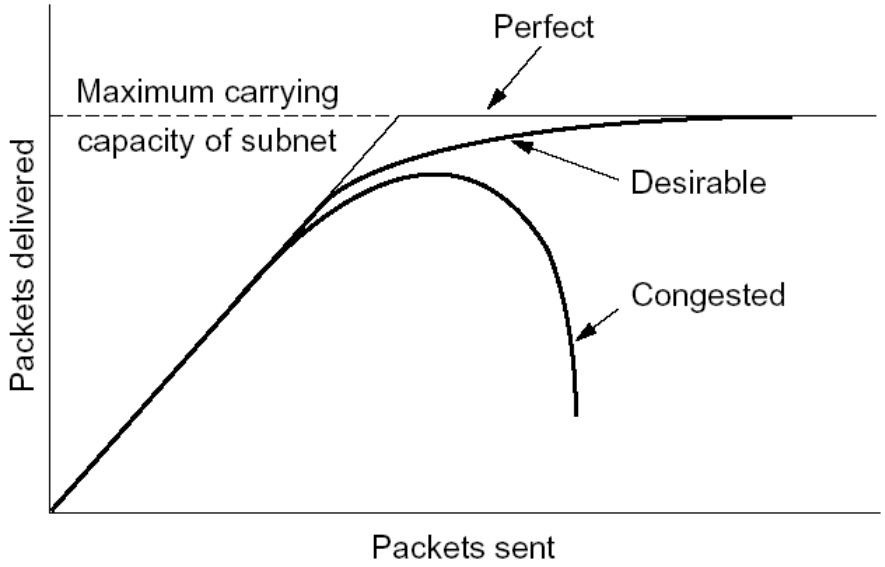
Hvordan?

- Gruppearbeid
- Diskusjon

Ukesoppgaver

Metningskontroll del1

1. Definer begrepene 'metning' og 'metningskontroll' i forhold til nettverk. Hvor og hvordan oppstår metning i et nettverk?
   - Flere pakker i nettet enn nettet kan håndtere.
   - Håndtere metning
   - I rutere, som mottar mer enn de kan håndtere

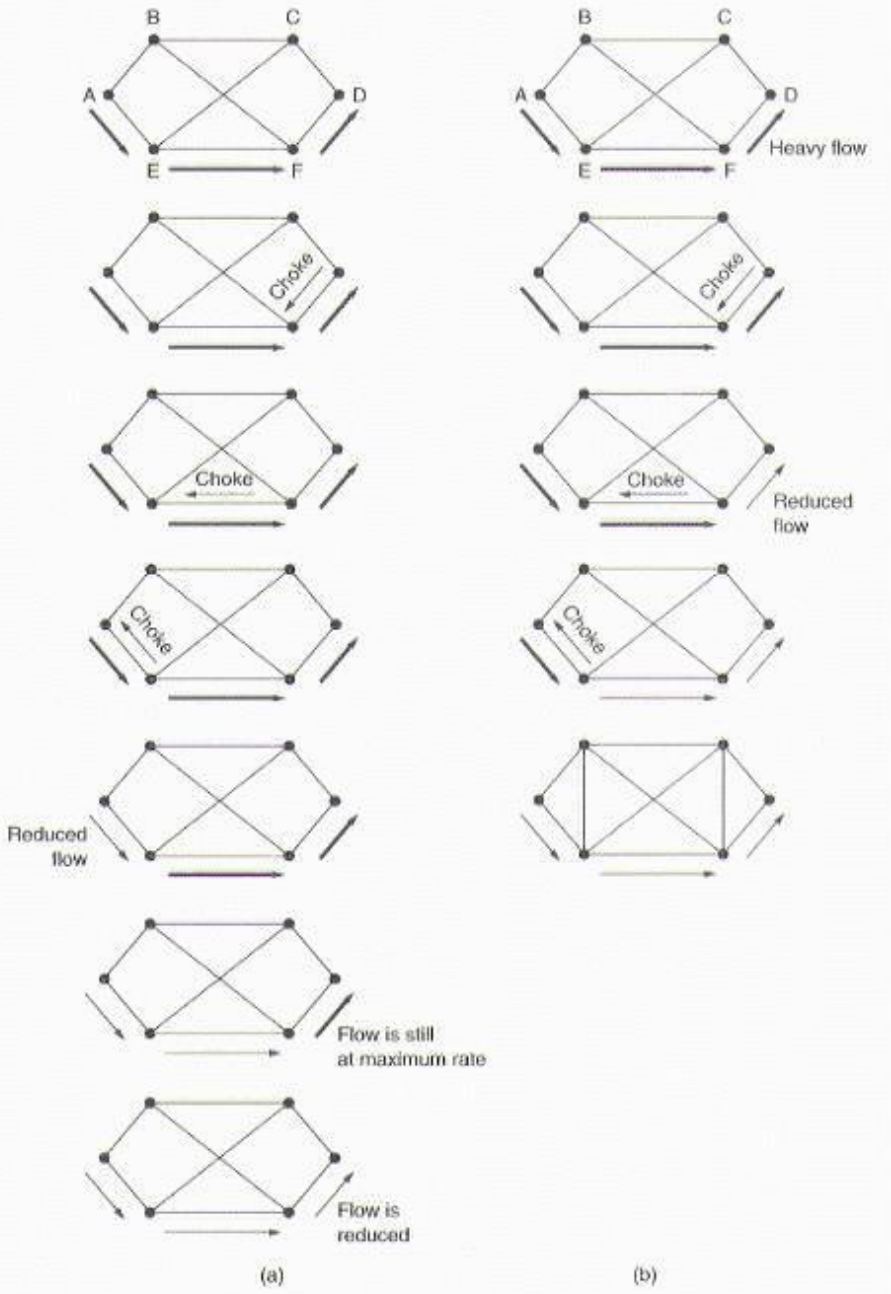


Metningskontroll del2

2. *Er det noen forskjell på metningskontroll i hhv. forbindelsesorienterte og -frie nettverk?*
   - Frie: Må håndtere metningskontroll
   - Forb. orientert: setter (ofte) av ressurser (Circuit switching) slik at vi unngår metning. Dersom ressurser ikke blir satt av kan vi oppleve metning.

Metningskontroll del3

3. Forklar begrepene *back pressure* og *soft state*.
   - *Back pressure: Bruker hele nettet til å bufre pakker når en node får metningsproblemer*
   - *Soft state: dynamisk ressurs allokering i ruter -> Gi ekstra plass når det trengs*



(a) A choke packet that affects only the source. (b) A choke packet that affects each hop it passes through.

Metningskontroll del4

4. Redegjør for hvilke lag i OSI-modellen vi finner metningskontroll, hvordan mekanismen fungerer på disse lagene, og evt hvordan de samspiller (utnytter tjenester).
   - Link – Retransmisjon, kvittering, flytkontroll, caching av out of order rammer.
   - Nettverkslaget – Ruting, kaste politikk, betjenings rekkefølge, TTL etc.
   - Transportlaget – Retransmisjon, kvittering, flytkontroll, timeout, caching av out of order datagram.
   - Problemet må håndteres i trans. laget.

Metningskontroll del5

5. *Redegjør for ulike typer kødisiplin (relatert til switcher/rutere) , og diskuter fordeler/ulemper ved disse typene.*
   - FIFO – venlig med taildrop – kaste pakker når vinduet er fullt. Tar ikke hensyn til hvilken flyt pakken kommer fra.
   - FQ – (fair queuing) ulik flyt plasseres i ulike køer som håndteres i round robin stil. En ¨misbehaving¨ flyt skaper ikke problemer for andre.

Metningskontroll del6

6. Beskriv ulike strategier for å unngå metning.
   - ECN bit i IP header. Informerer om at metning er under oppbygning. (Kan detekteres ved at snitt kø er mer enn 1 pakke)
   - RED: (random early detection) Når metning nærmer seg, fjern en tilfeldig pakke. TCP i transportlaget vil detektere pakketap og dermed redusere senderaten sin. Resurskrevende linker vil i snitt tape flest pakker grunnet tilfeldig valg av droppet pakke.

Metningskontroll del7

7. Forklar hvordan metningskontroll er implementert i TCP, og knytt din forklaring til begrepene *slow start*, *fast retransmit og fast recovery.*
   - AIMED control law – øk senderaten sakte, reduser fort ved deteksjon av metning
   - Slow start: start med vindu på 4 segmenter. For hver ACK, øk vindu med 1 inntil øvrig grense (dobler vindu for hver RTT). Ved timeout, sett øvrige grense til halve sendevinduet og start ¨additive increase¨(+1 per RTT).
   - Fast retransmit: Mottar ACK (x-1) 3 ganger -> retransmitter segment x og halverer øvrig grense.
   - Fast recovery: Tell mottatte duplikat ACK etter halvering av vindu. Når antall ACK = øvrig grense, send en ny pakke for hver mottatte duplikat ack

Metningskontroll del8

8. Anta at vi har en rundetid på 10 msec og ingen metning. Mottakervinduet er på 24 KB og maksimumssegmentet (maximum segment size/MSS) er 2 KB. Hvor lang tid tar det før det første fulle vinduet blir sendt?
   - 0RTT: vindu 2KB
   - 1RTT: vindu 4KB
   - 2RTT: vindu 8KB
   - 3RTT: vindu 16KB
   - 4RTT: vindu 24KB (nådd maks mottaker vindu størrelse)
   - -> Totalt 4RTT => 4*10 = 40msec

Metningskontroll del9

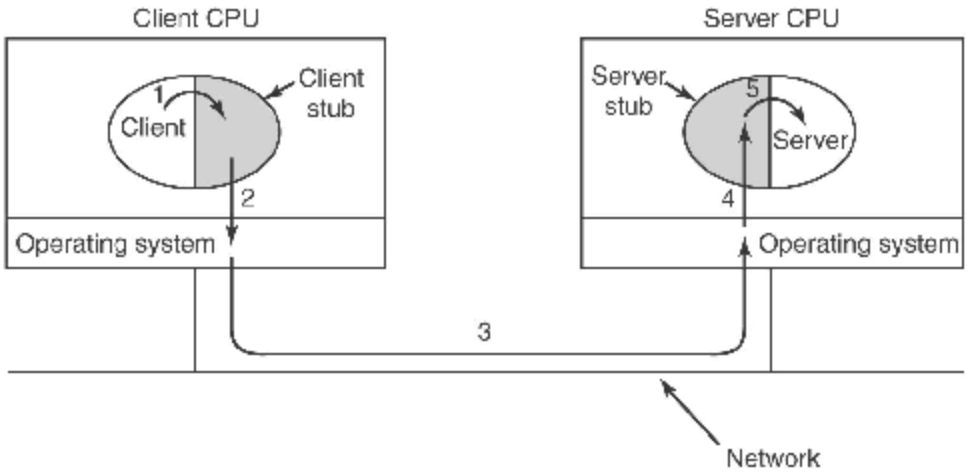
9. Anta at TCP sitt metningsvindu er på 18 KB og en timeout forekommer. Hvor stort vil vinduet være hvis de neste 4 tranmisjonene er vellykkede?
   - Anta MSS på 2KB
   - 0: Treshold = 18/2 = 9KB, Window = 2KB
   - 1: Treshold = 9KB, Window = 4KB
   - 2:Treshold = 9KB, Window = 8KB
   - 3:Treshold = 9KB, Window = 9KB
   - 4:Treshold = 9KB, Window = 10KB

Metningskontroll del10

10. *En TCP maskin sender fulle vinduer på 65.535 bytes over en 1-Gbps linje med en 10 msec forsinkelse. Hva er maksimal gjennomstrømning (throughput)? Hvor mye av linjen utnyttes?*
    - 10ms for å sende vindu + 10ms for å motta AKC -> gjennomstrømning = 65535/20ms = 3276750B/s = 26214000bps.
    - Utnyttelse = 26214000/(2^30) = 2,4%

RPC del1

1. *Hva er RPC, og beskriv hvordan det fungerer?*
   - Remote procedure call.



   - Stub: gjemmer bort faktum at prosedyren utføres på en annen maskin.

RPC del2

2. Redegjør for de fire ulike kall semantikkene RPC kan operere med.
   - *RPC kan ha ulik leveringsgaranti:*
   - *exactly-once: Kallet utføres en gang. Garanterer ikke resultat. Kan ikke skille mellom tjener og nettverksfeil*
   - *at-most-once*: Utføres maks en gang. Alle duplikat kall blir oversett
   - *at-least-once: Duplikater blir ikke oversett*
   - *maybe: Klient retransmittere ikke ved timeout. Vet ikke om kallet ble mottatt.*

Takk for nå!

Neste uke: Applikasjons laget