RIP PROTOKOL OPPGAVE 2 INF142

David Ronen og Espen Kleivane

Forbindelse etablering mellom klient/server

Det første protokollen må gjøre er å oppnå en "connection-oriented" forbindelse mellom klient og server. Dette er gjort ved at klienten og serveren utfører en sekvens bestående av 3 steg:

1. Serveren lytter, klienten initierer kontakt(syn) sender en forespørsel som inneholder b.l.a server-nummer (navn) og port nummer (server) samt hvilken ip den har og porten som til hører prosessen hvor ønsket kommunikasjon skal foregå.
2. Dette medfører at server lager en socket dedikert til den forbindelsen, server fortsetter å lytte til mulige andre via sin hoved-socket. Server sender svar på forespørsel (synak), under denne prosessen blir flere variabler tilordnet: forventet første sekvensnummer av klientpakker, nødvendige buffere o.l.
3. Klienten vil i dette steget svare på (synak), svaret kan (men må ikke) inneholde «payload»(data). Når forbindelsen er etablert kan de utveksle pakker som inneholder data.

Vår protokoll vil simulere en forenklet versjon av de ovennevnte stegene for å etablere forbindelse mellom klient og tjener.
For eksempel siden vår kanal går over UDP, vill ikke severen og klienten ha en dedikert «rør» seg imellom, men serveren vil da «huske» kilenten, og vil inngå i en tilstand der har venter dataoverføring fra klienten. I tillegg vil vi simulere syn og synack på en forenklet måte.

Pålitelig data overføring

Pålitelig data overføring innebærer at alle pakker kommer frem, kommer i riktig rekkefølge og uten feil. For å oppnå dette kreves en mekanisme som håndterer mulig svikt (f.eks. Loss av pakker, korrupsjon av pakker osv.).Vi implementerer her en GBN protokoll. Denne protokollen bidrar til bedre utnyttelse av nettet siden sender kan sende flere pakker uten å vente på server bekreftelse (som er praksis i «stop and wait» protokollen) og samtidig sørger for at pakke tap o.l. blir håndtert. Dette målet blir oppnådd ved bruk av positiv tilbake melding fra mottakeren for pakker som er riktig mottatt. Samtidig har senderen en timer-funksjon som tar tiden fra sent til forventet mottak av positiv tilbakemelding fra mottaker.
I tilfelle av pakke tap, eller tids forsinkelse (timeout) vil sender sørge for å sende noen eller alle pakker en gang til. Dette innebærer at senderen vil sende ut de pakker som ligger i bufferen som det ikke er kommet positiv tilbake melding på. Disse sendes på nytt til mottakeren.

Klient:

- Pakker som er klar til sending, inneholder et sekvens-nr. Disse blir lagt i senderens buffer i.h.t dette nummeret(f.o.m. base = 0 t.o.m. n-1). Senderens «sliding vindu» er > 1(antall tillatte pakker i røret). Vinduet flyttes fram for hver riktig-mottatt ack.

Server:

- Pakker som er mottatt i riktig rekkefølge hos mottaker blir lagt i mottakers buffer for å bli videre sendt til applikasjonen.

- Mottaker har et vindu = 1 (dette sørger for riktig rekkefølge av mottatt pakker (en og en) og kaster pakker som ikke er i riktig rekkefølge, samtidig som han re-akke den siste riktig mottatt pakke.

- Server mottar pakke 1 og sender positiv tilbake melding til klienten og formidler pakken opp til applikasjonslaget. Neste innkommende pakke kan nå behandles.

FSM

De ovenfor beskrevet prosessen innebærer at både sender og mottaker må reagere på flere hendelser. Klienten (sender) må i tillegg til initiering tilstand reagere på tre typer hendelser:

1. signal fra applikasjonslaget
2. ventetid er slutt (timeout)
3. mottak av positiv bekreftelse

Sender:

Hendelse 1;
klienten, prosessen står i en vente tilstand, når applikasjonslaget vil sende data vil det bli signalisert til prosessen som vil da sjekke buffer plass, hvis plass - motta data:

- send data
- start timer
- øke neste sekvens-nr
- gå til vente tilstand

Hvis ikke plass i bufferen, nekt å motta data gå til vente tilstand.

Hendelse 2;
klienten, får ikke tilbake melding innen tidsfristen (Ack timeout), eller får en tilbakemelding på en allerede akket pakke:

- start timer på nytt
- send alle pakker som ikke er acket (tilbakemelding) og er inni vinduet, en etter en til mottaker.
- gå til vente tilstand

Hendelse 3:
klienten, mottar positiv tilbakemelding:

- neste pakke til sending (base) = acknr + 1 (buffer = neste pakke til sending (base) til og med n-1), sliding vinduet flyttes fram til første ikke akket pakke.
- stopp timer for akket pakke.
- Start timer for neste (tidligst sendt) ikke akket pakke.
- Vent på neste akk.
- .
- .
- .
- Ga til vente tilstand.

Mottaker.

Hendelse 1:
Server starter i vente tilstand og er klar for å motta pakker:

- hvis pakke er riktig (ønsket sekvens nummer og ikke inneholder feil):
- send positiv tilbakemelding
- lever data til applikasjonslaget
- vente tilstand

Hvis pakken sitt sekvensnummer ikke er sist mottatt + 1:

- kast pakke
- send tilbakemelding for siste riktig-mottatt pakke.
- vente tilstand

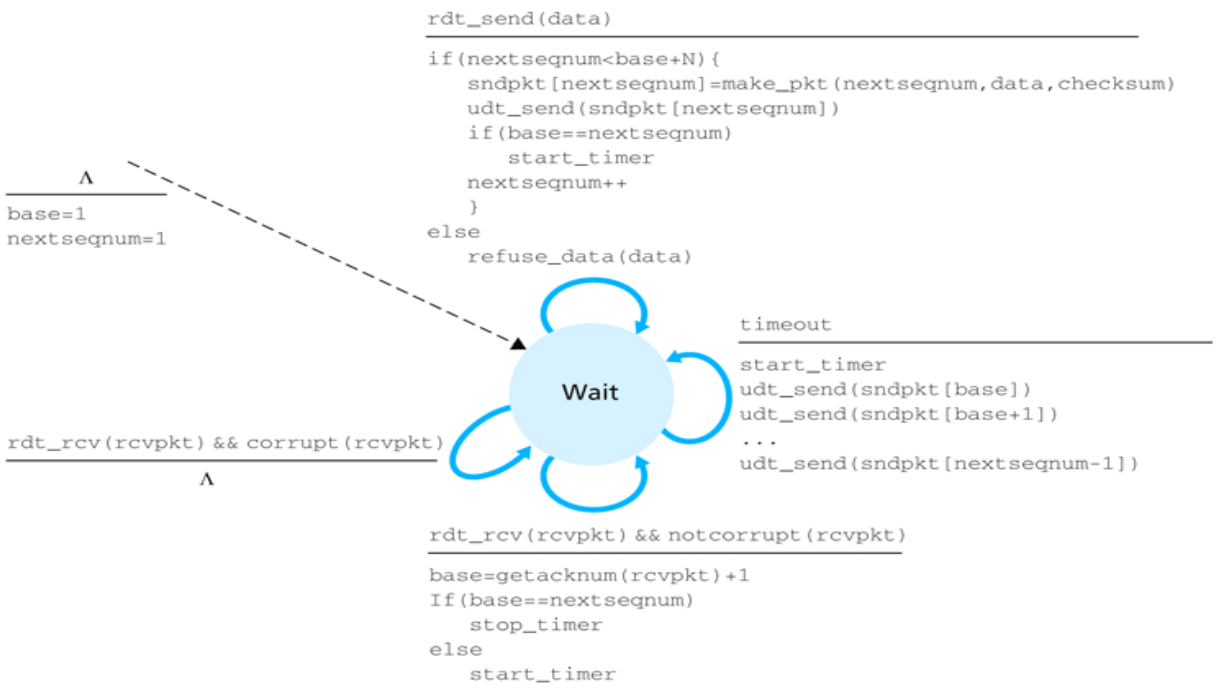
Avslutte forbindelsen

1. Klient sender slutt melding til server.
2. Server mottar slutt melding, og svarer med positiv tilbakemelding, begynner å stoppe prosessen og sender slutt melding til klienten.
3. Klienten mottar slutt melding og svarer med positiv tilbakemelding, går i vente tilstand.
4. Server mottar positiv tilbakemelding, fra klienten og lukker prosessen helt
5. Etter «timeout»hos klienten blir forbindelsen slutt

Vår protokoll vil simulere en forenklet versjon av de ovennevnte stegene for å avslutte forbindelse mellom klient og tjener.

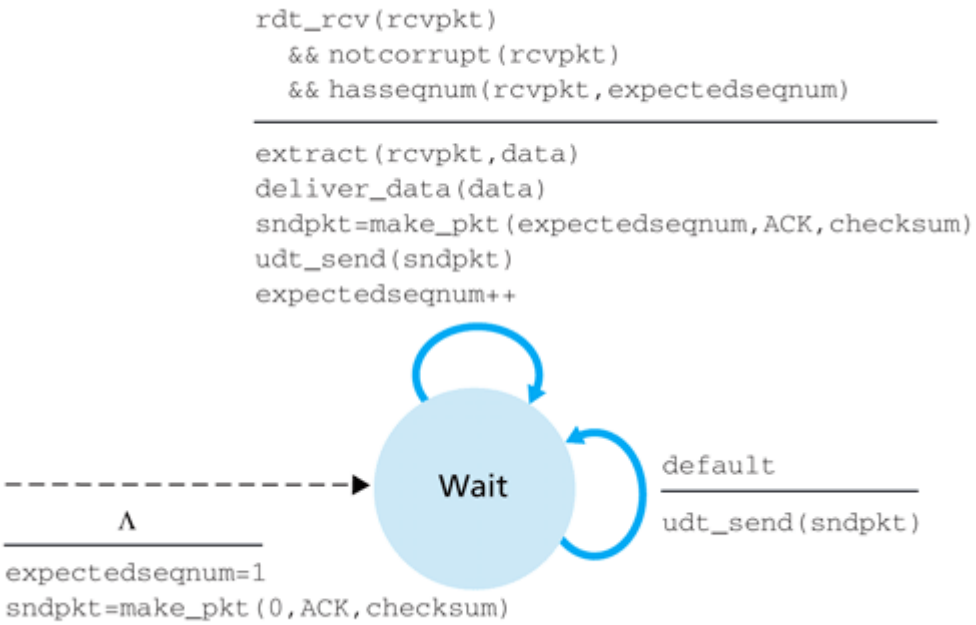
For en abstraksjon av kommunikasjonskanalen se vedlegg(142[skisse])

<u>Klient FSM</u>



Bilde 1 klient «Finite state machine»

Server FSM



Bilde 2 server «finite state machine»

Kilder:

Bilde 1 < http://www.utsc.utoronto.ca/~rosselet/cscd58w12/tut/t4/gbn_sndr.png>
[lastet ned 19.03.2012]

Bilde 2 **< http://www.utsc.utoronto.ca/~rosselet/cscd58w12/tut/t4/gbn_rcvr.png>**
[lastet ned 19.03.2012**]**