Hjemmeopgavesæt 2

34334 Avancerede datanet

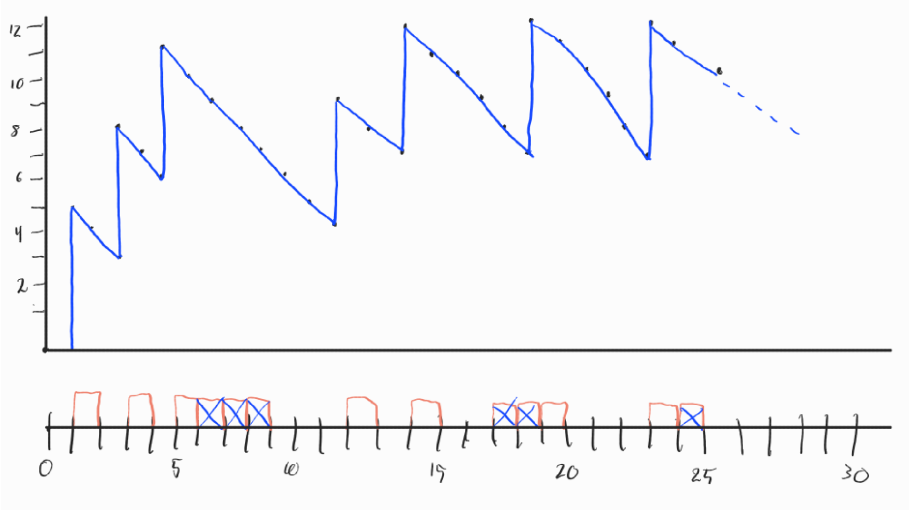
Daniel Brasholt s214675

December 2022

Indhold

Opgave 1 Leaky bucket

Spm 1.1.

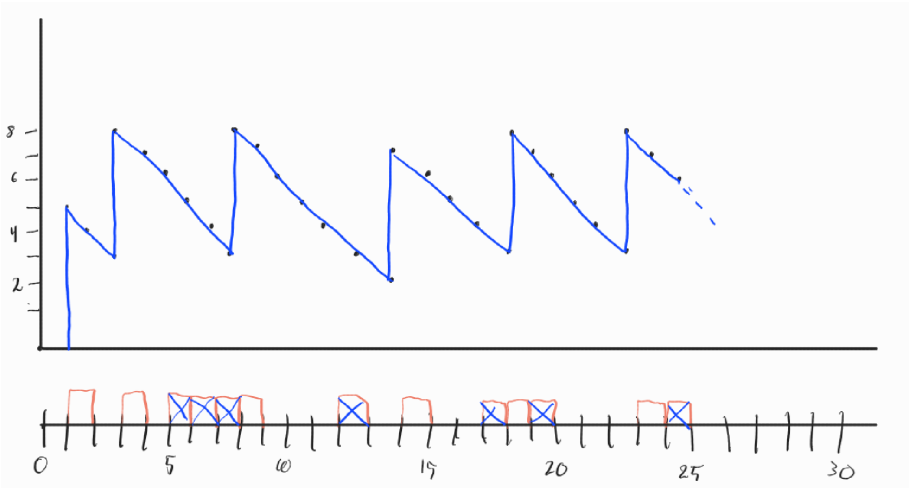


Figur 1: *Diagram over bucket og modtagne pakker, bucket size* $= 12$

På ovenstående figur 1 kan det ses, hvordan de modtagne pakker lægger sig i spanden. Hver gang en pakke modtages, stiger spandens indhold med 5. For hver tidsenhed bliver spandens indhold sænket med 1. Når en ny pakke skal lægges i spanden, kræves det, at spanden kan indeholde de ekstra 5 dataenheder; der må da kun være 7 dataenheder i spanden, når en ny pakke modtages. Derfor bliver følgende pakker markeret som non-conforming:

$$p_6, p_7, p_8, p_{17}, p_{18}, p_{24}$$

Spm 1.2.



Figur 2: *Diagram over bucket og modtagne pakker, bucket size* $= 8$

På figur 2 ses, hvordan pakkerne bliver sendt gennem spanden, når bucket size er ændret til 8. Det er også markeret, hvilke pakker, der er non-conforming. Disse er:

$$p_5, p_6, p_7, p_{12}, p_{17}, p_{19}, p_{24}$$

Spm 1.3.

Hvis pakker er non-conforming, risikerer netværket ikke at kunne overholde QoS. Da skal disse pakker håndteres med en policing-funktion, som enten kan virke ved blot at droppe pakkerne eller alternativt tagge pakkerne, så disse kan droppes som det første i tilfælde af network congestion. Dette gøres for at sørge for, at netværket kan efterleve aftalen om kvalitetssikring for enhederne i netværket.

Opgave 2 TCP Capture

Spm 2.1.

2.1.1

Det kan ses i pakke capture under 3-way handshake, at både klient og server vælger $MSS = 1460byte$. Denne værdi kommer fra, at den mest normale MTU i et pakkenetværk er sat til $1500byte$. MSS markerer den maksimale længde, indholdet i et `TCP`-segment kan have; herfra ekskluderes headeren fra både `TCP` og `IP`. Da begge disse normalt er sat til $20byte$, anvendes ofte $MSS = (1500 - 20 - 20)byte = 1460byte$, hvilket også er tilfældet i kommunikationen mellem klient og server. Netop denne værdi vælges, da det giver størst sandsynlighed for, at en pakke ikke skal fragmenteres undervejs, men alligevel så stor som muligt, da dette nedsætter antallet af `TCP`-segmenter, der skal sendes.

2.1.2

Formålet med Window Scale er at ændre værdien af modtagernes *receive window*, som ellers er $65535byte$. `TCP`-kommunikation er begrænset af *congestion window* og *receive window*. Det første er for at undgå congestion undervejs i netværket; den anden er for at undgå, at modtageren bliver overvældet med for meget data på én gang. Værdien $WS = 128$ (som er det vindue, både klient og server giver i den givne TCP-kommunikation) betyder da, at *receive window* kan skaleres med en faktor 128.

Spm 2.2.

2.2.1

Den normale Window Size er, som nævnt ovenover, 65535 byte. Det vil da tage $65535 \cdot 8/(2 \cdot 10^9)$ sekunder at transmittere hele vinduet. Dette svarer til under et milisekund. Der vil da ikke være nok tid til at dataen kan propagere hele vejen gennem netværket og at en ACK når tilbage. For at A skal kunne sende hele tiden, skal en ACK nå at komme tilbage inden hele vinduet er transmitteret. Da skal tiden, det tager A at transmittere vinduet, være dobbelt så stor som $t_{prop} = 120ms$. Dette gør sig gældende ved Window Scale lig 10, da det så tager følgende tid at sende hele vinduet:

$$2^{10} \cdot \frac{65535 \cdot 8}{2 \cdot 10^9} \approx 0.268s > 2 \cdot 0.12s$$

Dette svarer netop til at dataen kan propagere gennem netværket og at en ACK kan nå tilbage, inden hele vinduet er transmitteret.

2.2.2

Ifølge RFC-5681, hæves cwnd med N hver runde, hvor N er det antal bytes, der er blevet kvitteret for i ACK, som ikke er kvitteret for før. Dette betyder, at cwnd starter med at være $3 \cdot SMSS$, når første runde starter. Når næste runde er i gang, har A fået 3 ACK for de første 3 segmenter. Vinduet er da blevet hævet til $6 \cdot SMSS$. Dette vindue vil fordobles hver gang en runde starter. Efter 14 runder, vil vinduet være tilstrækkeligt stort til, at det tager mere tid for A at transmittere et helt vindue, end det gør for første pakke at ankomme hos B og for B at sende en ACK tilbage. Ligesom i ovenstående spørgsmål, skal det tage længere tid for A at sende hele vinduet end $2 \cdot t_{prop}$, hvilket sker efter 14 runder:

$$
\begin{aligned}
&2^{14} \cdot \frac{3 \cdot SMSS \cdot 8}{2 \cdot 10^9} \\
&= 2^{14} \cdot \frac{3 \cdot 1460 \cdot 8}{2 \cdot 10^9} \\
&= 0.287s > 2 \cdot 0.12s
\end{aligned}
$$

Svaret er da svarmulighed *b:* 14. Tiden, det tager for de 14 runder, må være omtrent 14 gange t_{prop}, da denne er så stor i forhold til tiden, det tager at sende første segment. Da er tiden cirka

$$14 \cdot 120ms = 1680ms = 1.68s$$

2.2.3

Når sstresh bliver sat til halvdelen af Flightsize, vil det betyde, at det bliver sat til cirka halvdelen af cwnd, da det er denne mængde data, der på det tidspunkt ikke er kvitteret for. cwnd vil da gå fra $2^{14} \cdot 3 \cdot SMSS$ til $2^{13} \cdot 3 \cdot SMSS$. Når cwnd stiger med $SMSS$ for hver round trip time, vil det betyde, at det tager cirka 16800 round trips, før cwnd er tilpas stor til at A kan transmittere hele tiden, da

$$\frac{\left(2^{13} \cdot 3 \cdot SMSS + 16800 \cdot SMSS\right) \cdot 8}{2 \cdot 10^9} = 0.2416 > 2 \cdot 0.12$$

Disse 16800 round trips vil tage cirka $16800 \cdot 120ms = 2016s$ til A's congestion window er stort nok til at transmittere hele tiden.

Opgave 3 Overførsel og bitfejl

Spm 3.1.

3.1.1

Propagation delay, t_{prop}, er givet ved den mængde tid, det tager for en bit af en ramme at komme fra sender til modtager; denne forsinkelse er da givet ved:

$$t_{prop} = \frac{150km}{2 \cdot 10^5 \frac{km}{s}} = 0.00075s = 0.75ms$$

Transfer time, t_f er givet ved den mængde tid, det tager at sende selve pakken. Det vil da være pakkens størrelse, $12MB = 96Mbit$, delt med bitraten[1].

$$t_f = \frac{12 \cdot 10^6 \cdot 8bit}{10 \cdot 10^9 \frac{bit}{s}} = 0.0096s = 9.6ms$$

3.1.2

Sandsynligheden for, at en enkelt bit overføres korrekt, er $1 - p$. Sandsynligheden for at alle bit i en ramme overføres korrekt, må da være $(1-p)^n$, hvor n er antallet af bit i rammen. Da vil sandsynligheden for en korrekt overførsel af filen være:

$$P\{\text{fejlfri}\} = (1-p)^n = (1 - 10^{-7})^{96 \cdot 10^6} = 0.0000677 \approx 0.00677\% \tag{1}$$

Som det kan ses i (1), er sandsynligheden for at kunne sende hele filen i en enkelt overførsel over linket meget lav.

[1] Jeg vil gå ud fra, at $12MB = 12 \cdot 10^6$ byte. Dette vil også ændre resultatet af **3.1.3**, men ved at gøre dette, giver resultaterne bedre mening

3.1.3

Fra forelæsningsslides er det givet, at den gennemsnitlige tid, det tager at transmittere en ramme fejlfrit, er givet ved:

$$E\{total\} = \frac{t_0}{1 - p_f}$$

Såfremt timeout-værdien er tilstrækkeligt tæt på tiden, det tager at sende en ramme og modtage kvittering. I ovenstående er $1 - p_f$ lig resultatet i (1). Derudover er t_0 givet ved:

$$2 \cdot t_{prop} + t_f + 2 \cdot t_{proc} + t_{ack}$$

Dog er det givet ud fra forudsætningerne, at t_{ack} samt t_{proc} kan ignoreres. Dette giver:

$$t_0 = 2 \cdot t_{prop} + t_f = 2 \cdot 0.75ms + 9.6ms = 11.1ms$$

Derved kan gennemsnitstiden findes:

$$E\{total\} = \frac{t_0}{1 - p_f} \approx 164s$$

Da kan det ses, at svaret er *c:* $164s/2.73min$[2].

Spm 3.2.

Når rammestørrelsen ændrer sig, vil t_f også ændre sig til

$$t_f = \frac{16384}{10 \cdot 10^9}$$

Derudover vil sandsynligheden for at sende en fejlfri ramme ændres til

$$(1 - p)^{16384} = 99.84\% \tag{2}$$

For så at finde tiden, det tager at sende hele filen, kan tiden det tager at sende en pakke multipliceres med antallet af pakker. Antallet af pakker vil være givet ved $12MB/16KB = 768$. Dette giver den gennemsnitlige transmissionstid:

$$E\{total\} = \frac{2 \cdot t_{prop} + t_f}{1 - p_f} \cdot 768 = 1.16s$$

Dette resultat stemmer ikke helt overens med svarmulighederne, men er dog tæt nok på *c:* $1.12s$ til at resten kan tilskrives afrunding.

Spm 3.3.

Fra forelæsningsslides er det givet, at har man m bit til at angive sekvensnummer, er det maksimale sende-vinduesstørrelse givet ved 2^{m-1}. Da der benyttes 1 byte til at angive sekvensnummer, må dette svare til $m = 8$, hvilket giver maksimal vinduesstørrelse

$$2^{8-1} = 2^7 = 128$$

Denne vinduesstørrelse er nok passende til dette kommunikationssystem. Dette er eftersom det vil tage i gennemsnit $200ms$ at sende 128 pakker af $16KB$, hvilket giver rigeligt tid til at en ACK eller NACK kan nå tilbage til afsenderen, eftersom t_{prop} er så lav i forhold til tiden det tager at sende hele vinduet. Det er også udregnet i (2), at sandsynligheden for at en pakke af denne størrelse kan sendes uden fejl er tilstrækkeligt stor til, at et større vindue ikke er nødvendigt.

[2]Denne udregning gik på, at $96Mbit$ er $96 \cdot 10^6$ bit, hvilket ikke helt følger definitionen. Laver man i stedet udregningen med $96 \cdot 1024^2$, får man svaret $261s$, hvilket ikke passer med svarmulighederne. Svarmulighed d kan dog fås ved at sætte t_{ack} til 60% af t_f.