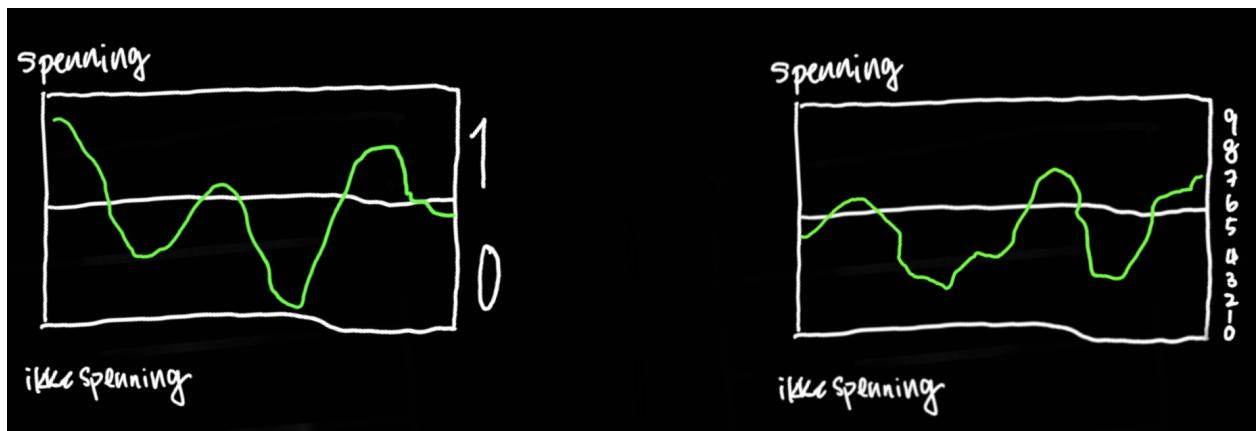


Oppgave 1. Generelt

- a) Datamaskin er en maskin som utfører matematiske operasjoner der hvor data blir representert som tall. Maskinen har en aritmetisk regne metode basert på boolsk logikk, true og false notert som tall 1 og 0. Tall 1 vil være spenning og tall 0 vil det ikke være spenning. Datamaskinen har logiske strøm kretser ofte kalt transistorer, hvor disse vil angi om det vil være 0 eller 1.

Vi bruker binære tallsystem fordi det er enkel og pålitelig. Det vil si at vi kan vite om det faktisk er 0 eller 1 selv om det kan være støy, lav spenning eller høy spenning, verdien vil være det nærmeste den kommer til. Vi kan forestille oss en måler hvor det er to punkter 1 og 0. Uansett om spenningen havner rundt 1 så vil det være i det området, og det samme blir tolket hvis den havner på 0 siden. Det vil være vanskelig å si hvis vi hadde en datamaskin som skulle tolke dette med et titallssystemet. Hvis spenningen havner mellom to tall eller ca halvveis midt i mellom, er det umulig å vite om det faktisk er den eller det andre.



Binære systemet er enkel å bruke, pålitelig som har lite sannsynlighet til å ta feil kontra andre avanserte strømkretser som er kompliserte og øker sannsynligheten for feil.

Det er tre fundamentale logiske kretser som vil være konjunksjon, disjunksjon og negasjon. Et eksempel på en konjunksjon vil maskinen utføre en AND matematisk operasjon hvor det vil være strøm KUN hvis transistor får tilførsel av strøm på begge sider. Transistorer vil fungere som en “placeholder” for verdien strømkretsen strømmet inn. Sammensatte 4 transistorer vil midlertidig lagre 4 bits som vil være det minste verdien maskinen vil forstå og utføre en oppgave.

- b) Transportlaget tilhører til en del av internett protokollen. Hovedfunksjonen på dette laget er å danne en forbindelse mellom klient og tjener. Forbindelsen som blir opprettet vil være en ende til ende kommunikasjon. Det vil si en sending fra sender og mottaker mottar en sending. Vi har to typer protokoller som er TCP og UDP.

TCP står for “*Transmission Control Protocol*” hvor vi har en kobling mellom klient (browser/bruker) og tjener (server/host). Det vil foregå en forespørsel og hvis koblingen er stabilt så skal det være en respons for hver forespørsel. Her er det en garanti på at sending kommer fram til mottaker. Det forveksles 3 sendinger som vi kaller for “*3-way handshake*”.

UDP står for “*User Datagram Protocol*” som ikke danne en fast kobling og det skaper usikkerhet fordi vi ikke har kontroll over om sendingen kom fram, derfor er det ingen garanti i sendingen. Det som er vesentlig forskjell er i TCP vil meldingen eller “segmentet” komme fram i riktig rekkefølge kontra UDP er det ikke noe syntaks på hvilken melding skal komme først fram. Et eksempel vil UDP være fremfor å bruke når

man strømmer video over nett. Hvis du mister noen segmenter her og der vil du ikke se sekunder av innholdet, derimot hvis du sender en viktig fil med melding og mister segmenter innimellom vil filen bli ubruklig eller “korrupte”.

- c) For å lagre data må vi ha en plass. Her vil det være mange typer av datalagringsmuligheter. En av de heter “floppy disk” som er tynne fleksible plastikk forseglet magnetdisketter med lagringsplass på ca 1.4MB. Dette vil være i stand til å lagre data fysisk og permanent på disketten. På magnet disken er det mange punkter som har oppgaven til å lagre en bit. Disse bits’ene vil inneholde siffer 0 eller 1. Et eksempel på bruk av datalagring er filmer fysisk på CD eller DVD. Etter som behovet for lagringsplassene måtte ekspanderes fordi filer (samling av bits) blir større bruker vi f eks SATA disk som har større lagringskapasitet.

Oppgave 2. Tall og binære data

a)

2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^10	2^11	2^12	2^13	2^14	2^15	2^16
1024	512	256	128	64	32	16	8	4	2	1	0	0	0	0	1	1
0	1	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1
$527 = 512 + 15 = 2^9 + 15$																
$15 = 8 + 7 = 2^3 + 7$																
$7 = 4 + 3 = 2^2 + 3$																
$3 = 2 + 1 = 2^1 + 1$																
$1 = 1 + 0 = 2^0 + 0$																
$= 0000\ 0010\ 0000\ 1111_2$																

527 i desimal til binær ved 16bit presisjon

b)

$ \begin{array}{r} 0111 \\ + 0110 \\ \hline = 1110 \end{array} \quad \begin{array}{l} 1000_2 \\ /101_2 \\ \hline 0101_2 \end{array} $
$ \begin{array}{r} 1110 \\ + 1100 \\ \hline = 1001 \end{array} \quad \begin{array}{l} 1100_2 \\ /1111_2 \\ \hline 1100_2 \end{array} $

c) Bruker **16 bit presisjon** her:

d) UTF 8 inneholder tabellen som ASCII hvor 32 første tegn er kontrolltegn og resterende er "amerikanske tegn". Utvidelsen på ASCII vil være som f eks Windows 1252 og ISO 8859-1, hvor det innføres alle "vestlige tegn" som "æøå". Kontroll tegnene vil være det

samme. Benytter en Byte for ASCII (eller to Byte for kodetegn som 1252 og 8859-1) i Unicode. Dekker meste behovet for standard bruk av engelsk. UTF 16 inneholder spesialtegn og andre språk i verden. Øst asiatiske språk som f eks kinesisk “科技”, japansk”テクノロジー” og koreansk “기술” vil det benyttes 1-3 Bytes.

e)

e)	$U+4D5 = \text{Hex D395 } \alpha$
$A=10$	4
$B=11$	D
$C=12$	5
$D=13$	
$E=14$	
$F=15$	
	0100 1101 0101 = 12 bits
	110X XXXX : 10XX XXXX = 2Byte
	1101 0011 . 1001 0101
	D 3 9 5

Her benyttes det 2 Byte for U+4D5 som er D395 i HEX

f)

$0xA1 \text{ AND } 0xCC = 0x80$	$0x48 \text{ AND } 0x92 = 0x00$
$A=10$ $B=11$ $C=12$ $D=13$ $E=14$ $F=15$ $0xA1 = 1010 0001$ & $0xCC = 1100 1100$ <u><u>$0x80 = 1000 0000$</u></u>	$0x48 = 0100 1000$ & $0x92 = 1001 0010$ <u><u>$0x00 = 0000 0000$</u></u>

$0x67 \text{ OR } 0x14 = 0x77$ $\begin{array}{r} 0x67 = 0110\ 0111 \\ \\ 0x14 = 0001\ 0100 \\ \hline 0x77 = 0111\ 0111 \end{array}$	$0xF0 \text{ OR } 0x89 = 0xF9$ $\begin{array}{r} 0xF0 = 1111\ 0000 \\ \\ 0x89 = 1000\ 1001 \\ \hline 0xF9 = 1111\ 1001 \end{array}$
$0x51 \text{ XOR } 0x95 = 0xA4$ $\begin{array}{r} 0x51 = 0101\ 0001 \\ ^ \\ 0x95 = 1001\ 0101 \\ \hline 0xA4 = 1100\ 0100 \end{array}$	

- g) Passord bitstyrke måles i form av “*brute force*” Formelen for å måle styrken på et passord er **lengden på passordet * log2(mulige tegn i passord -n)** der log2 er 2 i base altså hvor mange ganger skal 2 gange med seg selv til (-n)
- Et eksempel på passord: ” tkerbest ” er det 8 tegn * log2(26) $\approx 8 * 4.700 \approx 37.6$ bit passordet ” tkerbest ” vil ha ca 37.6bit i brute force.

Oppgave 3. Praktiske oppgaver

- a) jeg skriver nslookup -query=ns . > table i terminal på OSX vil jeg gjøre en DNS “Domain Name Server” spørring mot root-serveren altså på toppen av server hierarkiet, også skriver det ut i en fil.

<pre> Non-authoritative answer: . nameserver = a.root-servers.net. . nameserver = b.root-servers.net. . nameserver = c.root-servers.net. . nameserver = d.root-servers.net. . nameserver = e.root-servers.net. . nameserver = f.root-servers.net. . nameserver = g.root-servers.net. . nameserver = h.root-servers.net. . nameserver = i.root-servers.net. . nameserver = j.root-servers.net. . nameserver = k.root-servers.net. . nameserver = l.root-servers.net. . nameserver = m.root-servers.net. Authoritative answers can be found from: a.root-servers.net internet address = 198.41.0.4 b.root-servers.net internet address = 199.9.14.201 c.root-servers.net internet address = 192.33.4.12 d.root-servers.net internet address = 199.7.91.13 e.root-servers.net internet address = 192.203.230.10 f.root-servers.net internet address = 192.5.5.241 g.root-servers.net internet address = 192.112.36.4 h.root-servers.net internet address = 198.97.190.53 i.root-servers.net internet address = 192.36.148.17 j.root-servers.net internet address = 192.58.128.30 k.root-servers.net internet address = 193.0.14.129 l.root-servers.net internet address = 199.7.83.42 m.root-servers.net internet address = 202.12.27.33 a.root-servers.net has AAAA address 2001:503:ba3e::2:30 b.root-servers.net has AAAA address 2001:500:200::b </pre>	<p>Her vil vi se en liste over root navne-tjenester på name server vi gjorde en spørring. Navne tjenester er en protokoll for binde ip-adresser med navn som er lettere å lese. Som f eks .no er et top-level domain som ligger et steg under root serveren.</p> <p>På bildet til venstre ser vi ip adressen til disse root serverene.</p> <p>F eks domain a.root-server.net har en ip adresse på 198.41.0.4</p>
---	--

- b) På OSX i terminal skriver jeg følgende kommando: “ftp speedtest.tele2.net” og logger mer inn med brukernavnet: anonymous. Jeg skriver så: “ls” for å gi meg oversikt over innholdet på serveren, videre vil jeg laste ned **1024 Feb 19 2016 1KB.zip**

Skriver følgende kommando: “get 1KB.zip”

<pre> 230 Login successful. ftp> ls 200 EPRT command successful. Consider using EPSV. 150 Here comes the directory listing. -rw-r--r-- 1 0 0 1073741824000 Feb 19 2016 1000GB.zip -rw-r--r-- 1 0 0 107374182400 Feb 19 2016 100GB.zip -rw-r--r-- 1 0 0 1024000 Feb 19 2016 100KB.zip -rw-r--r-- 1 0 0 104857600 Feb 19 2016 100MB.zip -rw-r--r-- 1 0 0 10737418240 Feb 19 2016 10GB.zip -rw-r--r-- 1 0 0 104857600 Feb 19 2016 10MB.zip -rw-r--r-- 1 0 0 1073741824 Feb 19 2016 10GB.zip -rw-r--r-- 1 0 0 1024 Feb 19 2016 1KB.zip -rw-r--r-- 1 0 0 1048576 Feb 19 2016 1MB.zip -rw-r--r-- 1 0 0 209715200 Feb 19 2016 200MB.zip -rw-r--r-- 1 0 0 20971520 Feb 19 2016 20MB.zip -rw-r--r-- 1 0 0 2097152 Feb 19 2016 2MB.zip -rw-r--r-- 1 0 0 3145728 Feb 19 2016 3MB.zip -rw-r--r-- 1 0 0 524288000 Feb 19 2016 800MB.zip -rw-r--r-- 1 0 0 53687091200 Jul 24 2014 50GB.zip -rw-r--r-- 1 0 0 52428800 Feb 19 2016 50MB.zip -rw-r--r-- 1 0 0 524288 Feb 19 2016 512KB.zip -rw-r--r-- 1 0 0 5242880 Feb 19 2016 5MB.zip drwxr-xr-x 2 106 109 81920 Dec 13 18:49 upload 226 Directory send OK. ftp> get 1KB.zip 200 EPRT command successful. Consider using EPSV. 150 Opening BINARY mode data connection for 1KB.zip (1024 bytes). 226 Transfer complete. 1024 bytes received in 0.000486 seconds (2.01 Mbytes/s) ftp> </pre>	<p>Får en 200 OK som er godkjent på spørringen fra meg. Det tok 0.000486 sekunder for å laste ned filen på 1KiB.</p>
--	--

c) Jeg skriver i Chrome adressefeltet følgende ut i fra URL syntaks

protocol://host/path?parameter

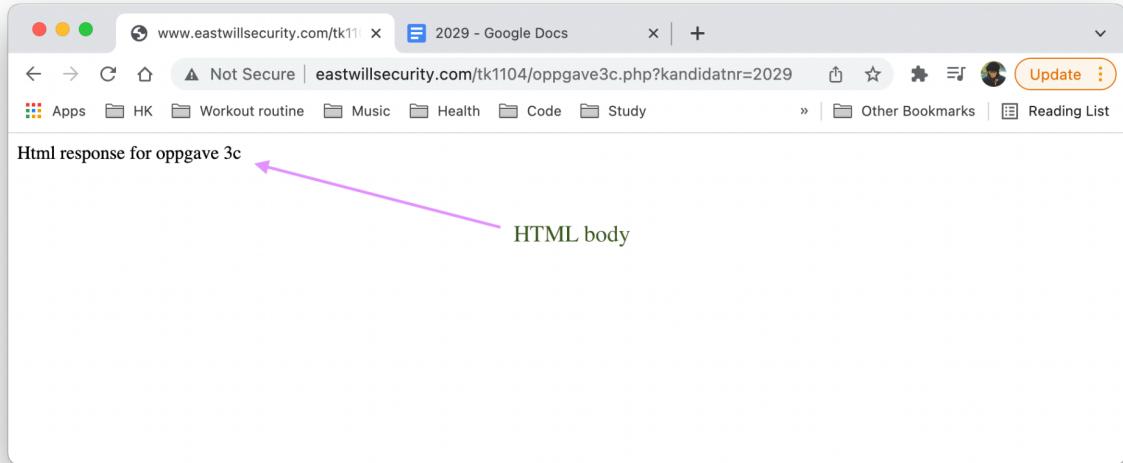
URL: **http://www.eastwillsecurity.com/tk1104/oppgave3c.php?kandidatnr=2029**

Protokoll: http://

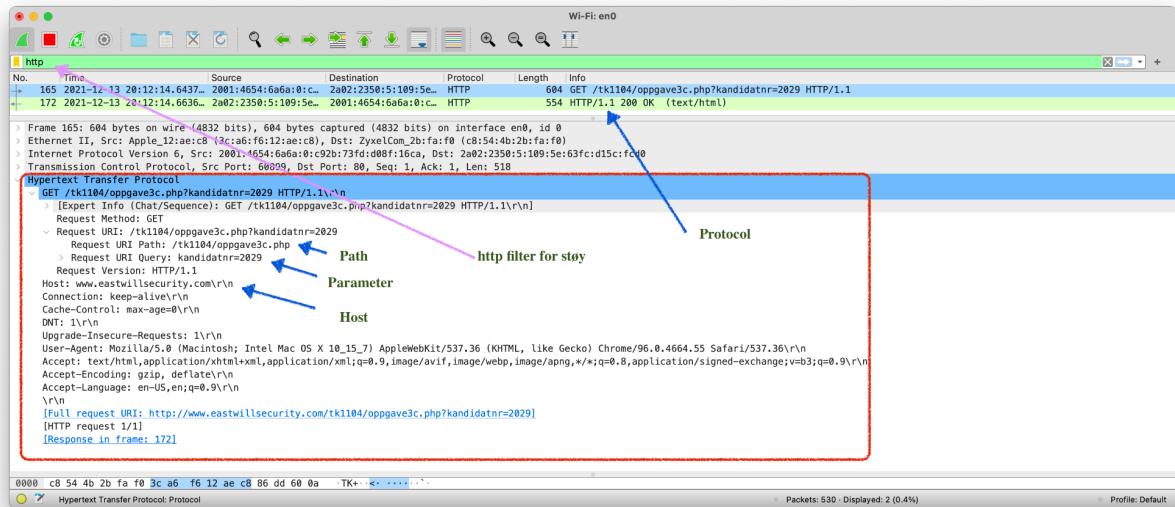
Host: www.eastwillsecurity.com

Path: /tk1104/oppgave3c.php

Query parameter: ?kandidatnr=2029



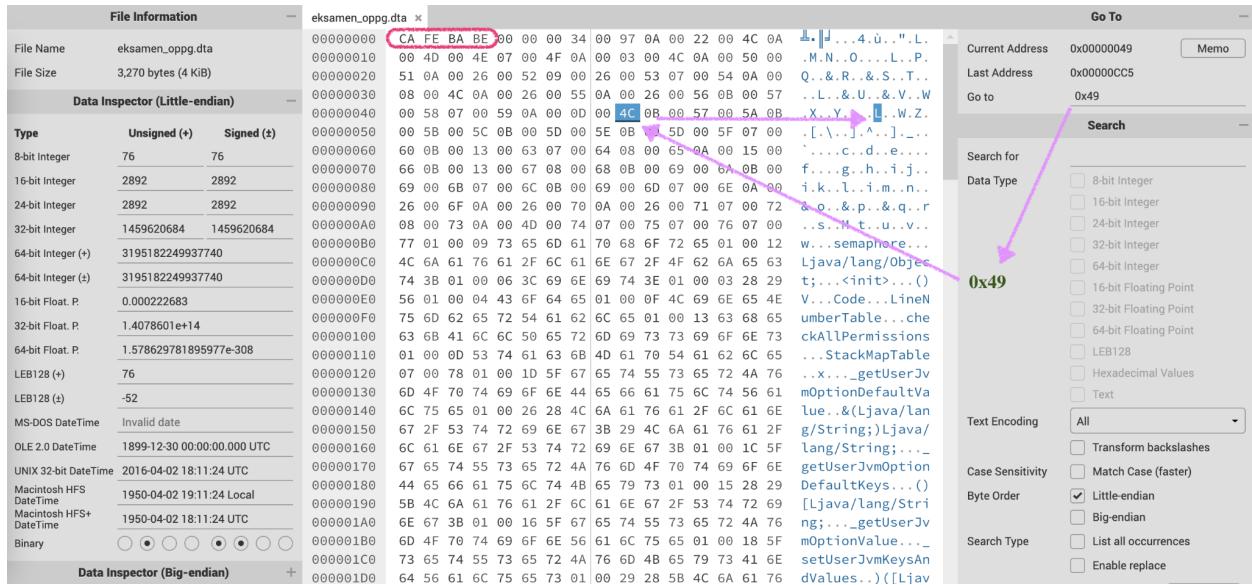
Jeg åpner så wireshark og trykker på Wi-Fi: en0 siden jeg er tilkoblet på en WiFi hjemme. Jeg inspiserer på datatrafikken som vises på Wireshark at her foregår det masse trafikkstrøm. I Wireshark filteret skriver jeg inn http hvor jeg ser kun http protokoller.



Som det vises på bildet er det jeg får opp i Wireshark med mye informasjon. Det jeg er interessert i er hva som er HTTP header meldingen her som jeg har satt en rød ramme rundt. Skulle egentlig ha lagt til BODY meldingen her også som vil være HTTP/1.1 200 OK (text/html) men har tydeligvis oppdatert siden og kjørt spørringen alt for mye så har visst blitt utestengt. Men meldingen i BODY vil være “*Html response for oppgave 3c*”

- d) Filen åpner jeg i min HEX editor og her ser jeg dette er en Java class fil fordi Magic

Number er 0xCA 0xFE 0xBA 0xBE ≈ CAFEBABE som står øverst.



videre taster jeg inn “0x49” som jeg får oppgitt i oppgaven og får da en hex verdi “0x4C”

som er karakter “L”

e)

```
[jacks-macbook-air:~ rinseo$ ls
1KB.zip      Documents      Movies      Public
Applications Downloads      Music       table
Desktop      Library       Pictures    txtWithPingCmd
[jacks-macbook-air:~ rinseo$ mkdir OPPG_3E
[jacks-macbook-air:~ rinseo$ ls
1KB.zip      Documents      Movies      Pictures      txtWithPingCmd
Applications Downloads      Music       Public
Desktop      Library       OPPG_3E    table
jacks-macbook-air:~ rinseo$ ]
```

I terminal skriver jeg først “ls” for å få fram listene og orientere meg videre. Herunder

skriver jeg “mkdir OPPG_3E” også “ls” en gang til for å se om mappen ble opprettet.

```
[jacks-macbook-air:~ rinseo$ ls
1KB.zip      Documents      Movies      Public
Applications Downloads      Music       table
Desktop      Library       Pictures    txtWithPingCmd
[jacks-macbook-air:~ rinseo$ mkdir OPPG_3E
[jacks-macbook-air:~ rinseo$ ls
1KB.zip      Documents      Movies      Pictures      txtWithPingCmd
Applications Downloads      Music       Public
Desktop      Library       OPPG_3E    table
[jacks-macbook-air:~ rinseo$ cd OPPG_3E
[jacks-macbook-air:OPPG_3E rinseo$ pwd
/Users/rinseo/OPPG_3E ←
[jacks-macbook-air:OPPG_3E rinseo$ ]
```

Jeg går inn på mappen som ble opprettet ved å skrive “cd OPPG_3E” etterfulgt med “pwd” for å sjekke hvor jeg befinner meg nå.

```
[jacks-macbook-air:OPPG_3E rinseo$ pwd
/Users/rinseo/OPPG_3E
[jacks-macbook-air:OPPG_3E rinseo$ echo TESTTESTTEST > TEST.TXT
[jacks-macbook-air:OPPG_3E rinseo$ ls
TEST.TXT
[jacks-macbook-air:OPPG_3E rinseo$ ]
```

Jeg skriver kommando “echo TESTTESTTEST > TEST.TXT” også skriver jeg “ls” for å se om “TEST.TXT” filen ble opprettet

```
/Users/rinseo/OPPG_3E
[jacks-macbook-air:OPPG_3E rinseo$ echo TESTTESTTEST > TEST.TXT
[jacks-macbook-air:OPPG_3E rinseo$ ls
TEST.TXT
[jacks-macbook-air:OPPG_3E rinseo$ cat TEST.TXT
TESTTESTTEST
[jacks-macbook-air:OPPG_3E rinseo$ ]
```

Jeg skriver kommando “cat TEST.TXT” og får opp innholdet opp i terminalen
“TESTTESTTEST”

```
[jacks-macbook-air:OPPG_3E rinseo$ cat TEST.TXT
TESTTESTTEST
[jacks-macbook-air:OPPG_3E rinseo$ rm TEST.TXT
[jacks-macbook-air:OPPG_3E rinseo$ ls
jacks-macbook-air:OPPG_3E rinseo$ ]
```

Her bruker jeg kommando “rm TEST.TXT” for å slette filen jeg opprettet, også “ls” for å sjekke om filen har blitt fjernet fra mappen “OPPG_3E”

```
[jacks-macbook-air:OPPG_3E rinseo$ ls
[jacks-macbook-air:OPPG_3E rinseo$ cd ..
[jacks-macbook-air:~ rinseo$ pwd
/Users/rinseo
[jacks-macbook-air:~ rinseo$ ls
1KB.zip Applications Desktop Documents Downloads
Library Movies Music OPPG_3E Pictures
Public table txtWithPingCmd
[jacks-macbook-air:~ rinseo$ rmdir OPPG_3E
[jacks-macbook-air:~ rinseo$ ls
1KB.zip Documents Movies Public
Applications Downloads Music table
Desktop Library Pictures txtWithPingCmd
jacks-macbook-air:~ rinseo$ ]
```

Til slutt må jeg ut av mappen “OPPG_3E” for å slette den. Først skriver jeg “cd ..” én hierarki opp. Skriver “pwd” for å orientere meg hvor jeg befinner meg nå. Skriver følgende “ls” for å se “OPPG_3E” etterfulgt av “rmdir OPPG_3E” også “ls” igjen. Nå har mappen “OPPG_3E” blitt slettet og eksisterer ikke i strukturen jeg befinner meg i.

Oppgave 4. Forståelse av nettverk

- Hub, switch og en Router opereres på linklaget i TCP/IP modellen. Her benyttes det Ethernet kabel på et lokalt nettverk for å danne en forbindelse mellom enhetene for å koble seg på internett. En av de mest vanlige kabel som benyttes er UTP “*Unshielded Twisted Pairs*” eller en Coaxial kabel, men signaler som går gjennom kabelen har

begrenset rekkevidde og derfor benyttes det en Hub for å “forlenge” signalene. Et slik enhet tar imot kobling fra ISP, f eks modem fra Telenor. Datapakker som kommer inn i Hub’en vil bli gjentatt og output det samme til alle sine porter der Hub’en vil oppdage at det er en kobling. Switch derimot er mer intelligente hvor enheten benyttes en Switch tabell hvor enheter som er koblet på Switch’en har sin MAC adresse eller fysisk adresse. Alle enheter har en innebygd fysisk adresse. Når et segment kommer inn til Switch vil den kun sende til destinasjon porten og ikke til alle portene. Hub og Switch benyttes for å danne et lokalt nettverk, for å koble seg på nettverket utenfor sitt lokalnettverk må vi bruke en Router. Dette er som en gateway som kobler på internett hvor det benyttes IP adresser for å sende pakker. Hovedoppgaven til en router er å videresende pakken som kommer utenfor sitt nettverk inn til destinasjonen hvis adressen samsvarer. En router må benyttes hvis flere enheter skal koble seg på samme internett.

- b) Datapakker som sendes på linklaget via Ethernet kabel kan foregå i en kollisjon. Altså hvis PC1 og PC2 begge sender pakke samtidig vil det skje en kollisjon med hverandre. En løsning på dette problemet var CSMA/CD “*Carrier Sense Multiple Access Collision Detection*” der datamaskin vil oppdage hvis nettverkskablen er fri for trafikk, så vil det sendes datapakke, hvis det foregår trafikk så vil datamaskinen vente med å sende pakken. Man kan ikke utelukke at kollisjoner ikke vil skje, så vil datamaskinene sende ut en melding for å varsle andre om det har foregått en kollisjon. De som har kollidert med hverandre blir satt i en “nedteller status modus” og vil gjenopprettes tilfeldig for å unngå kollisjon igjen.

- c) DHCP står for “*Dynamic Host Configuration Protocol*” er en server ofte innebygd i en Router som gir en enhet IP adresse. IP adressen kan være innstilt til å være statiske eller dynamiske. I starten måtte man manuelt konfigurere enhver enhet for å koble seg på internett, innstillinger som IP adresse, subnet mask, gateway til router og DNS server. Man måtte være nøyne med å tildele unikt IP adresser til enhetene for å unngå adresse konflikter. Hvis to eller flere enheter får tildelt samme IP adresse vil de ikke kunne koble seg på nettverket. Enklere oppsett av dette kan man få tildelt IP adressene automatisert ved å velge dynamiske IP adresser. Enheten vil da sende en DHCP discover melding, og DHCP tjenesten vil svare tilbake med DHCP offer og “lease” en IP adresse for denne enheten. Det å “lease” en IP adresse er for å sikre hvis enheten har koblet av fra nettverket vil IP adressen forfalle og DHCP server kan gjenbruke samme adressen for en annen enhet. Enheter vil sende signaler til DHCP server om å få fornye IP adressen sin periodisk.
- d) SMTP står for “*Simple Mail Transfer Protocol*” bruker samme ende til ende kobling mellom klient og server som TCP protokoll hvor vi starter med “3-way handshake” *SYNchronize* får da *SYNchronize + ACKnowledge* i svar, og svarer tilbake med *ACKnowledge*. Som det vises på bildet vil SMTP bruke en TCP port 25 og vil garantere at sendingen ankommer til mottakeren. Her har også SMTP en format sammensatt av SMTP header, mail header og mail body som vil være meldingen. SMTP brukes både mellom forskjellige klienter som PC1 og PC2, mellom mail server og klient til server.

Kort forklart så vil SMTP protokoll dirigere mailen frem til destinasjonen.

