

Oppgave 1. Generelt

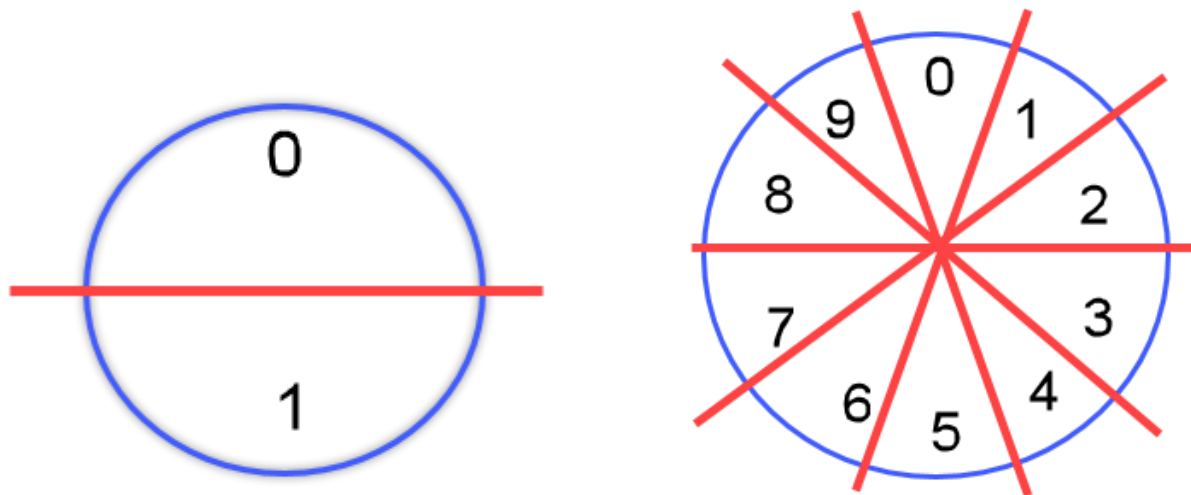
a) Forklar hvorfor datamaskiner bruker binær logikk. Forklar kort hva binær logikk og maskinens fysiske oppbygging har å si for pålitelighet og produksjonskostnader for komponenter i en datamaskin. Tegn gjerne noen figurer som illustrerer dette.

Det er tre ting som gjør at maskiner benytter binær logikk; kompleksitet, pålitelighet og effektivitet. Alt dette har gjort at vi mennesker konkluderte med å lage maskinene våre å benytte to-tallsystemet og ikke ti-tallsystemet som vi mennesker benytter i andre omstendigheter.

Vi kan starte å se på pålitelighet og hvorfor dette trolig er den største faktoren.

Nedenfor har jeg tegnet to tegninger som skal illustrere hva jeg mener med pålitelighet.

På det første bilde hvor sirkelen er delt inn i to skal representere binær systemet og bilde nummer to er for ti-tallsystemet.



Når vi ser på binærsystemet og skal tenke på margin på feil så har hver verdi 50% av sirkelen hvor signaler kan bomme på og selv skåre riktig verdi. Dette skaper en pålitelighet som trengs spesielt når maskiner har så mange måter å få forstyrret signaler på! Se for deg et signalsystem har en feilmargin på 10%. Det krever utstyr og transport metoder for data på et helt annen nivå enn det vi har i dag, som vill lede til neste poeng.

Kompleksitet speiler direkte opp mot prisen på det vi skal lage. Dette gjelder også maskinkomponenter alt fra være moderne hånd holde mini maskiner (referer til mobiltelefoner) til hjemme maskinene under skrivebordet vi brukt til kontor arbeid og de store serverne som bedrifter som Google og Amazon har. Skulle vi lage transistorer som skal håndtere og ti forskjellige signalere og skille mellom dem uten å ta feil ville denne ene biten blitt mye større fysisk, mer kompleks og dyrere å lage. Når vi tenker at moderne mikroprosesser som er på maskinen væres er på et par kvadrat centimeter som har millioner av disse transistorene. Kanskje vi maks kunne hatt per hundre tusen på samme plass med et ti-tallsystem? Jeg ville gjettest at vi har hengt tilbake i forskjell pga. pris og produksjon hvis vi skulle benyttet så komplekse komponenter.

Siste punkt som jeg ikke kommer å gå i dybden i pga. mangel av forståelse, men har blitt undervist som et punkt. To-tallsystemet som vi benytter er teoretisk sett det mest effektive symbolsystemet som finnes. Som konklusjon har man kun fordeler med å lande på og videre benytte et binersystemet på maskiner.

b) Forklar forskjellen på TCP og UDP protokollene på Transportlaget. Gi noen eksempler på situasjoner det vil være fornuftige å velge den ene fremfor den andre.

TCP og UDP er to former for å sende ut segmenter med informasjon til en annen maskin. Begge to har sine fordeler men i moderne tid har TCP landet på toppen av bedre kontroll over sikkerhet og feilsjekking av segmenter.

TCP (Transmission Control Protocol) benytter en handshake system som betyr at to maskiner blir enig på forhånd at data skal sendes mellom hverandre. Hver pakke har sin egen sender ID og mottaker bekreftet alltid mottakelsen før neste sett med pakker sendes ut. Skulle noe ikke stemme med pakken vill mottakende maskin slette mottatte pakker å etterspørre fra punket ting gikk galt og avsender avbryter strøm og starter på nytt fra ønsket punkt.

UDP (User Datagram Protocol) er det vi kaller en fire and forget system. Avsender har fått beskjed hvor den skal sende datagrammene og sender dem dit, men det er ingen kontroll om de kommer fram. Fordelen med denne tjenesten er at man har mulighet å sende ut mye større volum med data og dette er den eneste måten å sende samme pakker til flere mottakere samtidig også kalt broadcasting. Man har en veldig primitiv måte å sjekke om at det er noe galt med pakkene og det er å benytte en sjekksum system som alltid skal ende på et sett med kun 1'ere. Hvis det ikke er tilfelle må mottaker si ifra at en pakke er feil og så må hele sendingen starte på nytt.

Som man fort kan skjønne er at TCP er godt egnet mellom en til en kommunikasjon i de aller fleste tilfeller og der man prioritere sikkerhet og stabilitet i pakkene enn fart. Kan være nedlastninger av filer, meldinger mellom to personer, status fra måler osv. Eksempelene er mange og bruksområde stort. For UDP er dette mest brukt for sosiale medier som video hvor mange skal se ting samtidig (eks: streams fra Twitch eller Zoom møte) hvor et segment med data ikke kom fram, siden et menneske vill mest sannsynlig ikke bry seg meget hvis et bilde mangler eller ser litt feil ut tanke på skjermer i dag viser 30-60 bilder i sekundet i de fleste tilfeller.

Noen siste ord om disse protokollene er fart på et nettverk som er viktig å ha i bakhode som ikke alltid kommer først. Siden TCP benytter handshake systemet har den muligheten å regulere hastigheten den sender ut på hvis den ser at pakker begynner å forsvinne av diverse årsaker. Dette har ikke UDP og kan fort spise opp nettverkskapasitet til et nettverk.

Et godt eksempel kan være tre personer benytter TCP så kan dem bruke 33% av nettet hver, men bruker en person UDP og to TCP så kan det se slikt ut i stedet. UDP tar 50% og TCP tar 25% hver og virker kanskje ikke som et stort problem i et hjemmenettverk, men her må man se ut fra små private nettverk og ut på størres systemer som servere.

c) Forklar hva en «floppy disk» i en PC er (var), hvilket bruksområde denne har?

Floppy disk kan vell best beskrives som den gamle ekvivalent til dagens minnepinner. Det var en firkantet falt diskett som inneholdte en form for plate som benytter magnetisme for å lagre data. Dette mediet ble så stort at vi i den dag bruker dens form som ikon til å lagre ting i dag. I dag er floppy disk minimalt brukt siden andre måter å lagre data på har bedre og større kapasitet med i hastighet og volum av lagring langt forbi gått floppy disk.

Oppgave 2. Tall og binære data

a) Konverter de to følgende desimaltallene til binærtall (16 bits presisjon). (Vis utregning.)

$52710 = ?_2$

$9910 = ?_2$

Eksempelen Oppgave 2.

2) $527_{10} \Rightarrow 512 + 15 = 2^9 + 15 \Rightarrow$

$$\begin{array}{rcl} 8 + 7 & = & 2^3 + 7 \\ 4 + 3 & = & 2^2 + 3 \\ 2 + 1 & = & 2^1 + 1 \\ 1 & = & 2^0 \end{array}$$

$527_{10} = 2^9 + 2^3 + 2^2 + 2^1 + 2^0$

$527_{10} = 0 \times 2^{15} + 0 \times 2^{14} + 0 \times 2^{13} + 0 \times 2^{12} + 0 \times 2^{11} + 0 \times 2^{10} + 0 \times 2^9 + 0 \times 2^8 + \dots$
 $\dots + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

$527_{10} = 0000\ 0010\ 0000\ 1111_2$

$99_{10} \Rightarrow 64 + 35 = 2^6 + 35$

$$\begin{array}{rcl} 32 + 3 & = & 2^5 + 3 \\ 2 + 1 & = & 2^1 + 1 \\ 1 & = & 2^0 \end{array}$$

\Downarrow

$99_{10} = 2^6 + 2^5 + 2^1 + 2^0$

$99_{10} = 0000\ 0000\ 0110\ 0011$

b) Utfør følgende binær addisjon med 8 bits presisjon. (Vis utregning.)

$$0111\ 10002 + 0110\ 11012 = ?2$$

$$1100\ 11012 + 1100\ 11112 = ?2$$

c) Utfør BINÆR subtraksjon av de to desimale tallene 376 minus 77. (Vis utregning ved bruk av toerkomplement, og utregningen MÅ gjøres med binære tall.)

$$376_{10} - 77_{10} = ?2$$

b)

$$\begin{array}{r} 0111\ 1000 \\ + 0110\ 1101 \\ \hline 1110\ 0101 \end{array}$$

$$\begin{array}{r} 1100\ 1101 \\ + 1100\ 1111 \\ \hline 1001\ 1100 \end{array}$$

c) $376_{10} - 77_{10} =$

$$376_{10} = 256 + 120 = 2^8 + 120 = 0000\ 0001\ 0111\ 1000$$

$$64 + 56 = 2^6 + 56$$

$$32 + 24 = 2^5 + 24$$

$$16 + 8 = 2^4 + 8$$

$$8 = 2^3$$

$$-77_{10} = 64 + 13 = 2^6 + 13 = -0000\ 0000\ 0100\ 1111$$

$$8 + 5 = 2^3 + 5$$

$$4 + 1 = 2^2 + 1$$

$$1 = 2^0$$

$$\begin{array}{r} 0000\ 0000\ 0100\ 1111 \\ 1111\ 1111\ 1011\ 0000 \\ + 0000\ 0000\ 0000\ 0001 \\ \hline 1111\ 1111\ 1011\ 0000 \end{array}$$

$$376_{10} = 0000\ 0001\ 0111\ 1000 = 0000\ 0001\ 0111\ 1000$$

$$-77_{10} = -0000\ 0000\ 0100\ 1111 = +1111\ 1111\ 1011\ 0000$$

$$= 209 = 2^8 + 2^5 + 2^3 + 2^1 = 0000\ 0001\ 0010\ 1001$$

d) Forklar forskjellen på UTF-8 og UTF-16, i hvilke tilfeller kan det lønne seg å velge det ene enkodingsformatet over den andre.

Forskjellen mellom kodepunktene UTF-8 og UTF-16 er hvor mange bits en serie inneholder. Et kodepunkt er hvordan man skal gjøre et tegn som vi mennesker skjønner og kan lese på skjermen over til maskinkode (binære tall). Når jeg skrev en bit serie kan man se for seg en «vogn» som inneholder en mengde bits.

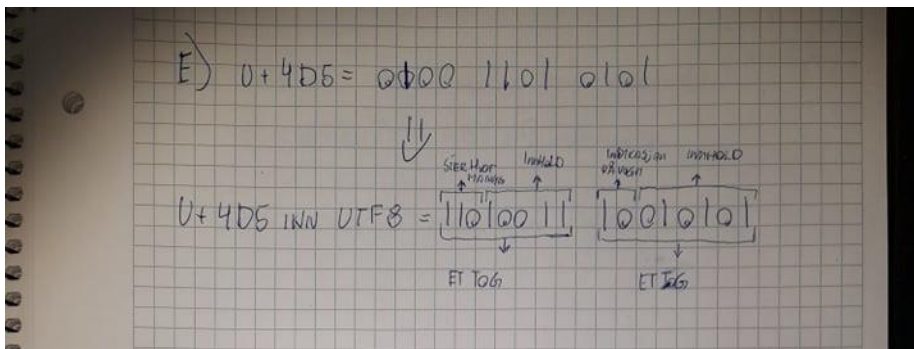
I UTF-8 så inneholder en vogn 8 bits. Hvor første vogn benytter minimum en bit i front for å signalisere hvor stor denne serien med vogner er. Desto flere vogner som kommer etter øker det med en bit. Dette er en meget fin og effektiv måte å lagre mindre verdier. I den vestlige verden er majoriteten av tegn som benyttes på datamaskinene våre, samme tegnsetting fra ASCII i det nye Unicode systemet. I ASCII er alle tegn basert seg på at man skal kunne hente dem opp med kun 7-bits. Siden Unicode som er det nye standard tegn-biblioteket som benyttes så har den betraktelig flere tegn i seg enn kun vestlige Alphabet. Som nevnt har vi fortsatt de samme plassene som tidligere som betyr vi kan hente majoriteten av tegn for den vestlige verden på en til to vogner som i større perspektiv er meget plass effektivt. Problemet er med en gang vi skal ha verdier som er over 11 bits, da trenger vi totale 3 vogner (totalt 24 bits) som blir mer effektivt å benytte UTF-16.

UTF-16 har samme tegnstruktur på front, men forskjellen er at vognene har størrelse på 16 bits hvis man inkludere signalbittene på fronten. Som nevnt er dette ugunstig for lite bit krevende bokstaver som vesten har, men mot østen som asiatiske land som har mange flere enn våre 26 tegn som eks. Kina som i benytter hvert tegn for et ord. Så har dem fått større verdier i Unicode tabellen. Skulle jeg skrevet Kinesisk ville hvert tegn i UTF-8 blitt totalen 24bits og i UTF-16 blitt 16bits.

Så i konklusjon så ønsker man å benytte UTF-8 ved lagring av data som i helhet benytter færre bits som eks. Vestlig skriftspråk. Skulle man ha ting som i bunn trenger større bits så burde man benytte UTF-16 som eks. Kinesisk skriftspråk.

Antall bitt som blir effektivt lagret	UTF-8	
(7 bit) 0xxxxxxx		
(11 bit) 110xxxxx	10xxxxxx	
(16 bit) 1110xxxx	10xxxxxx	10xxxxxx
	UTF-16	
(15 bit) 0xxxxxxxx xxxxxxxx		
(27bit) 110xxxxx xxxxxxxx	10xxxxxx xxxxxxxx	
(40bit) 1110xxxx xxxxxxxx	10xxxxxx xxxxxxxx	10xxxxxx xxxxxxxx

e) Utfør UTF-8 enkodning av Unicode-punktet U+4D5 (et tegn fra Kyrillisk alfabet – som ser likt ut som en bokstav i vårt latinske alfabet). Oppgi svaret i hex.



f) Utfør følgende boolske operasjoner. (Vis utregning.)

$$0xA1 \text{ AND } 0xCC = ?$$

$$0x48 \text{ AND } 0x92 = ?$$

$$0x67 \text{ OR } 0x14 = ?$$

$$0xF0 \text{ OR } 0x89 = ?$$

$$0x51 \text{ XOR } 0x95 = ?$$

Handwritten calculations for hexadecimal bitwise operations:

$$\begin{aligned} \text{F) } 0xA1 &= 1010 \ 0001 \\ \text{AND } 0xCC &= 1100 \ 1100 \\ &= 0x80 = 1000 \ 0000 \\ \\ 0x48 &= 0100 \ 1000 \\ \text{AND } 0x92 &= 1001 \ 0010 \\ &= 0x00 = 0000 \ 0000 \\ \\ 0x67 &= 0110 \ 0111 \\ \text{OR } 0x14 &= 0001 \ 0100 \\ &= 0x77 = 0111 \ 0111 \\ \\ 0xF0 &= 1111 \ 0000 \\ \text{OR } 0x89 &= 1000 \ 1001 \\ &= 0xF9 = 1111 \ 1001 \\ \\ 0x51 &= 0101 \ 0001 \\ \text{XOR } 0x95 &= 1001 \ 0101 \\ &= 0x2A = 0010 \ 1010 \end{aligned}$$

g) Hva er bitstyrken til et passord med 8 tegn hvor passordet består av små bokstaver (a-z) og tall? (Vis utregning.)

Handwritten calculation for password bit strength:

$$\begin{aligned} \text{G) } \text{bitstyrke} &= \lg_2(\text{Ferdigstillelige tegn i passordet})^{\text{Antall tegn}} \\ &= \lg_2(26+10)^{8} = 86.69 \text{ bitstyrke} \\ \text{P.S.} &= \text{Svar er regnet på kalkulator} \end{aligned}$$

Oppgave 3. Praktiske oppgaver

a) Hva gjør kommandoen "nslookup -query=ns . > table" på kommandolinje/terminal? (Du kan svare for det operativsystemet du kjenner best; Windows, Linux eller OSX, men du må oppgi hvilket operativsystem du har brukt i besvarelsen.)

Jeg benytter Windows 10 når jeg beskriver denne oppgaven.

"nslookup -query=ns . > table" er kommando som kan skrives inn i command prompt/powershell for å hente ut informasjon. Hvilken informasjon skal jeg prøve å bryte ned nå.

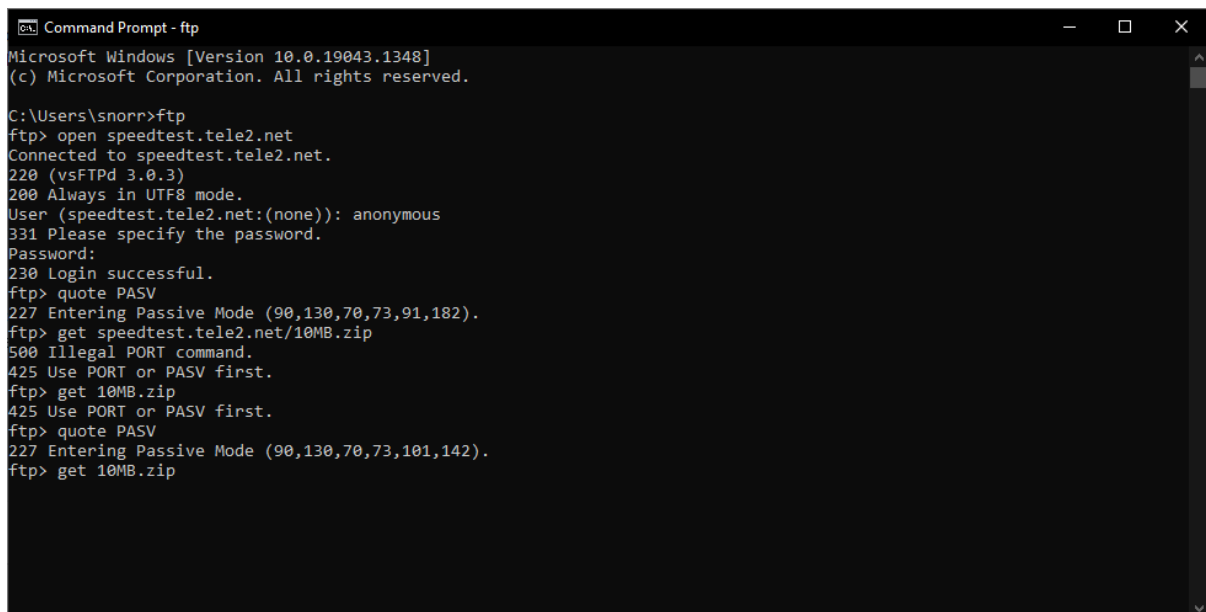
Nslookup: dette er hoved kommandoen som benyttes for å hente informasjon i et DNS infrastruktur.

-query: Dette er en parameter som spisser inn hvilken informasjon du har lyst på, men funker ikke alene og må ha følgende etter seg:

=ns: Dette kaller man en beskrivelse til parameteren og sier hvilken type informasjon man ønsker. Her spør man etter navngitt server.

. > table: Siste biten av spørringen sier at du skal printe ut svaret i et dokument som heter table.

b) Demonstrer kommandoen «ftp» på kommandolinje/terminal, bruk domenet ftp://speedtest.tele2.net og last ned noen testfiler fra serveren. Hvilke nedlastingshastigheter oppnår du? Dokumenter kommandoer du har brukt og resultater enten med tekst eller screenshot.



```
Command Prompt - ftp
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\snorr>ftp
ftp> open speedtest.tele2.net
Connected to speedtest.tele2.net.
220 (vsFTPd 3.0.3)
200 Always in UTF8 mode.
User (speedtest.tele2.net:(none)): anonymous
331 Please specify the password.
Password:
230 Login successful.
ftp> quote PASV
227 Entering Passive Mode (90,130,70,73,91,182).
ftp> get speedtest.tele2.net/10MB.zip
500 Illegal PORT command.
425 Use PORT or PASV first.
ftp> get 10MB.zip
425 Use PORT or PASV first.
ftp> quote PASV
227 Entering Passive Mode (90,130,70,73,101,142).
ftp> get 10MB.zip
```

Kommentar: Jeg kommer meg inn på FTP server, men jeg får ikke til nedlastningen.

Mistenker det er noe jeg gjør galt ved å velge port modus. Når jeg prøver «quote port 20» så blir jeg frakoblet.

c) I denne oppgaven skal du både demonstrere forståelse av bruk av verktøyet Wireshark og web browser, og generell forståelse av URL og web sider.

Åpne opp monitoreringsverktøyet Wireshark for å kunne undersøke nettverksprotokoller.

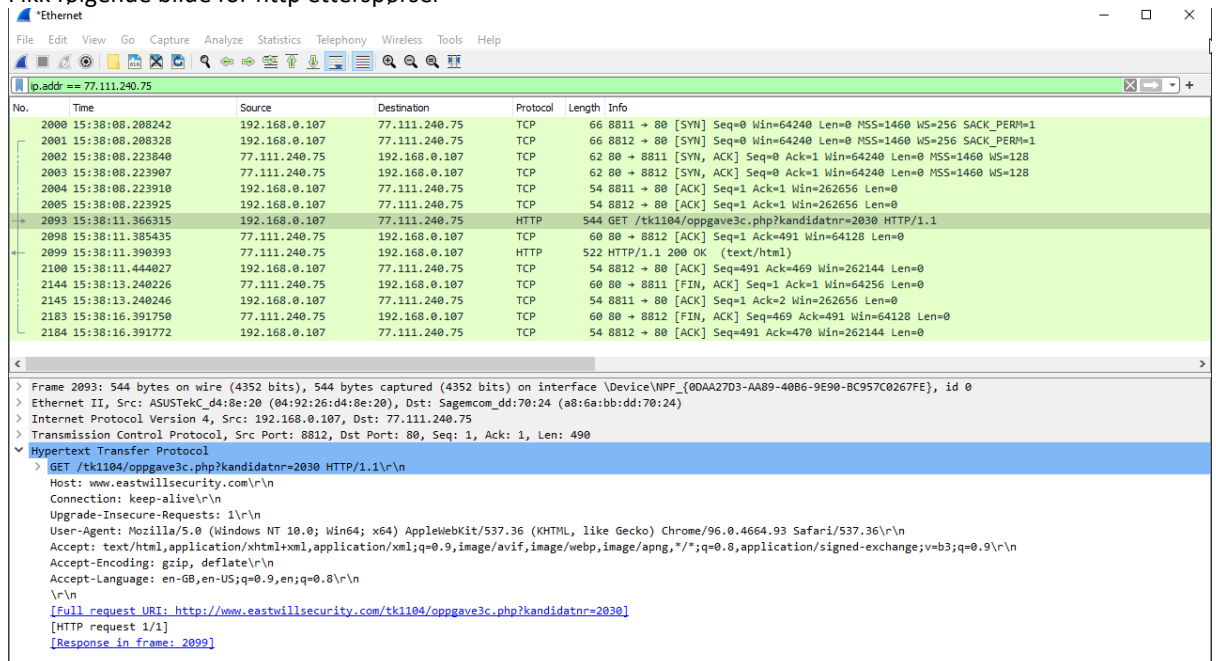
Du skal så åpne en valgfri nettleser (web browser) og gå til en URL hvor PROTOKOLL er http (obs, ikke bruk https, da vil du ikke kunne se svaret i wireshark), HOST er www.eastwillsecurity.com, PATH er tk1104/oppgabe3c.php og QUERY parameter er kandidatnr=XYZ hvor XYZ settes til _ditt_ kandidatnummer på

denne eksamen.

Bruk Wireshark for å inspisere nettverkstrafikken. Hvilke http-headere får du tilbake fra denne serveren (oppgi alle headere og deres verdier)? Oppgi også innholdet på nettsiden du får tilbake. Forklar fremgangsmetoden du brukte for å løse oppgaven.

Min framgang gikk som følgende;

1. Åpner opp wireshark i administrerende modus og velger ethernet som kilde.
2. Bruker CMD nslookup for å finne IPv4 adresse til nettsiden www.eastwillsecurity.com (77.111.240.75)
3. Bruker kommando ip.addr == 77.111.240.75 i wireshark for å filtrere etter hvilken IPv4 adresse wireshark skal logge.
4. gjør klar URL <http://www.eastwillsecurity.com/tk1104/oppgabe3c.php?kandidatnr=2030> i chrome.
5. Fikk følgende bilde for http etterspørsel



Og følgende bilde for mottakende http protokoll.

The image shows a Wireshark packet capture of an HTTP response. The packet list shows a sequence of TCP and HTTP packets. The selected packet (No. 2099) is an HTTP 200 OK response from 192.168.0.107 to 77.111.240.75. The packet details pane shows the following structure:

- Frame 2099: 522 bytes on wire (4176 bits), 522 bytes captured (4176 bits) on interface \Device\NPF_{0DAA27D3-AA89-40B6-9E90-BC957C0267FE}, id 0
- Ethernet II, Src: Sagemcom_dd:70:24 (a8:6a:bb:dd:70:24), Dst: ASUSTekC_d4:8e:20 (04:92:26:d4:8e:20)
- Internet Protocol Version 4, Src: 77.111.240.75, Dst: 192.168.0.107
- Transmission Control Protocol, Src Port: 80, Dst Port: 8812, Seq: 1, Ack: 491, Len: 468
- Hypertext Transfer Protocol**
 - HTTP/1.1 200 OK\r\n
 - Date: Mon, 13 Dec 2021 15:38:11 GMT\r\n
 - Server: Apache\r\n
 - X-Powered-By: PHP/7.4.26\r\n
 - Expires: 0\r\n
 - Cache-Control: must-revalidate\r\n
 - X-Hash: 472047204000203056176398\r\n
 - Vary: Accept-Encoding\r\n
 - Content-Encoding: gzip\r\n
 - Content-Length: 77\r\n
 - Content-Type: text/html; charset=UTF-8\r\n
 - X-Varnish: 460362431\r\n
 - Age: 0\r\n
 - Via: 1.1 varnish (Varnish/7.0)\r\n
 - Accept-Ranges: bytes\r\n
 - Connection: keep-alive\r\n
 - \r\n
 - [HTTP response 1/1]
 - [Time since request: 0.024078000 seconds]
 - [Request in frame: 2093]
 - [Request URI: http://www.eastwillsecurity.com/tk1104/oppgabe3c.php?kandidatnr=2030]
 - Content-encoded entity body (gzip): 77 bytes -> 69 bytes
 - File Data: 69 bytes
- Line-based text data: text/html (1 lines)

Headers er alt som ligger under første linje på http linjen.

Innhold jeg fikk fra nettsiden var i ren tekst «Html response for oppgave 3c»

The image shows a web browser window with the address bar displaying "Not secure | eastwillsecurity.com/tk1104/oppgabe3c.php?kandidatnr=2030". The page content is "Html response for oppgave 3c".

d) I denne oppgaven skal du demonstrere forståelse for filer og filformater, og praktisk bruk av en hexeditor. Først last ned denne filen:

http://www.eastwillsecurity.com/tk1104/oppgave3d/eksamen_oppg.dta

Åpne denne filen i en hexeditor. Hva er MAGIC NUMBER i denne filen, og hva sier det deg om hva slags type fil dette er? (Filen du har lastet ned har endret filending, så den må du se bort fra.) Hvilken byte verdi er på plass 0x49 (byte nummer 0x49) i filen, oppgi verdien i hex. Dokumenter begge svarene ved å ta skjermbilde av hexeditoren du bruker, i tillegg til å forklare i tekstbesvarelsen. Husk at bildene må settes inn på riktig sted i besvarelsen.

Magic number i denne filen er CA FE BA BE – CAFE BABE som betyr at dette er en Java .class fil. For hvilken byte verdi som ligger på 0x49 så er det 4C.

The screenshot shows a hex editor window titled 'HxD - [C:\Users\snorr\Downloads\eksamen_oppg.dta]'. The main window displays the hex data and its decoded text. The decoded text is a Java class file header and some code. The 'Data inspector' panel on the right shows various data types and their values. The 'Byte order' section shows 'Little endian' selected.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	CA	FE	BA	BE	00	00	00	34	00	97	0A	00	22	00	4C	0A	Ëþ%...4.-..".L.
00000010	00	4D	00	4E	07	00	4F	0A	00	03	00	4C	0A	00	50	00	.M.N.O....L..P.
00000020	51	0A	00	26	00	52	09	00	26	00	53	07	00	54	0A	00	Q.&.R.&.S..T..
00000030	08	00	4C	0A	00	26	00	55	0A	00	26	00	56	0B	00	57	..L..&.U..&.V..W
00000040	00	58	07	00	59	0A	00	0D	00	00	00	00	57	00	5A	0B	.X..Y...Z..W.Z.
00000050	00	5B	00	5C	0B	00	5D	00	5E	0B	00	5D	00	5F	07	00	.[.\..].^..]..
00000060	60	0B	00	13	00	63	07	00	64	08	00	65	0A	00	15	00	`....c..d..e....
00000070	66	0B	00	13	00	67	08	00	68	0B	00	69	00	6A	0B	00	f....g..h..i..j..
00000080	69	00	6B	07	00	6C	0B	00	69	00	6D	07	00	6E	0A	00	i..k..l..i..m..n..
00000090	26	00	6F	0A	00	26	00	70	0A	00	26	00	71	07	00	72	&.o..&.p..&.q..r
000000A0	08	00	73	0A	00	4D	00	74	07	00	75	07	00	76	07	00	..s..M..t..u..v..
000000B0	77	01	00	09	73	65	6D	61	70	68	6F	72	65	01	00	12	w...semaphore...
000000C0	4C	6A	61	76	61	2F	6C	61	6E	67	2F	4F	62	6A	65	63	Ljava/lang/Objec
000000D0	74	3B	01	00	06	3C	69	6E	69	74	3E	01	00	03	28	29	t;...<init>...()
000000E0	56	01	00	04	43	6F	64	65	01	00	0F	4C	69	6E	65	4E	V...Code...LineN
000000F0	75	6D	62	65	72	54	61	62	6C	65	01	00	13	63	68	65	umberTable...che
00000100	63	6B	41	6C	6C	50	65	72	6D	69	73	73	69	6F	6E	73	ckAllPermissions
00000110	01	00	0D	53	74	61	63	6B	4D	61	70	54	61	62	6C	65	...StackMapTable
00000120	07	00	78	01	00	1D	5F	67	65	74	55	73	65	72	4A	76	..x..._getUserJv
00000130	6D	4F	70	74	69	6F	6E	44	65	66	61	75	6C	74	56	61	mptionDefaultVa
00000140	6C	75	65	01	00	26	28	4C	6A	61	76	61	2F	6C	61	6E	lue..&(Ljava/lan
00000150	67	2F	53	74	72	69	6E	67	3B	29	4C	6A	61	76	61	2F	g/String;)Ljava/
00000160	6C	61	6E	67	2F	53	74	72	69	6E	67	3B	01	00	1C	5F	umber/String;...
00000170	67	65	74	55	73	65	72	4A	76	6D	4F	70	74	69	6F	6E	getUserJvmOption
00000180	44	65	66	61	75	6C	74	4B	65	79	73	01	00	15	28	29	DefaultKeys...()
00000190	5B	4C	6A	61	76	61	2F	6C	61	6E	67	2F	53	74	72	69	[Ljava/lang/Stri
000001A0	6E	67	3B	01	00	16	5F	67	65	74	55	73	65	72	4A	76	ng;..._getUserJv
000001B0	6D	4F	70	74	69	6F	6E	44	65	66	61	75	6C	74	56	61	mptionValue...
000001C0	73	65	74	55	73	65	72	4A	76	6D	4B	65	79	73	41	6E	setUserJvmKeysAn
000001D0	64	56	61	6C	75	65	73	01	00	29	28	5B	4C	6A	61	76	dValues..)(Ljav
000001E0	61	2F	6C	61	6E	67	2F	53	74	72	69	6E	67	3B	5B	4C	a/lang/String;[L
000001F0	6A	61	76	61	2F	6C	61	6E	67	2F	53	74	72	69	6E	67	java/lang/String
00000200	3B	29	56	01	00	15	5F	67	65	74	55	73	65	72	4A	76	;)V..._getUserJv
00000210	6D	4F	70	74	69	6F	6E	44	65	79	73	01	00	11	67	65	mptionKeys...ge
00000220	74	55	73	65	72	4A	56	4D	4F	70	74	69	6F	6E	73	01	tUserJVMOptio
00000230	00	11	28	29	4C	6A	61	76	61	2F	75	74	69	6C	2F	4D	..()Ljava/util/M
00000240	61	70	3B	07	00	76	07	00	72	07	00	79	07	00	7A	01	ap:...v...e...y..z.
00000250	00	09	53	69	67	6E	61	74	75	72	65	01	00	37	28	29	..Signature..7()
00000260	4C	6A	61	76	61	2F	75	74	69	6C	2F	4D	61	70	3C	4C	Ljava/util/Map<L
00000270	6A	61	76	61	2F	6C	61	6E	67	2F	53	74	72	69	6E	67	java/lang/String
00000280	3B	4C	6A	61	76	61	2F	6C	61	6E	67	2F	53	74	72	69	;Ljava/lang/Stri
00000290	6E	67	3B	3E	3B	01	00	11	73	65	74	55	73	65	72	4A	ng;>...setUserJ
000002A0	56	4D	4F	70	74	69	6F	6E	73	01	00	12	28	4C	6A	61	VMOptions...(Lja
000002B0	76	61	2F	75	74	69	6C	2F	4D	61	70	3B	29	56	07	00	va/util/Map;)V..
000002C0	7B	07	00	7C	07	00	60	01	00	38	28	4C	6A	61	76	61	{...}...8(Ljava
000002D0	2F	75	74	69	6C	2F	4D	61	70	3C	4C	6A	61	76	61	2F	util/Map<Ljava/
000002E0	6C	61	6E	67	2F	53	74	72	69	6E	67	3B	4C	6A	61	76	lang/String;Ljav
000002F0	61	2F	6C	61	6E	67	2F	53	74	72	69	6E	67	3B	3E	3B	a/lang/String;>;
00000300	29	56	01	00	18	67	65	74	55	73	65	72	4A	56	4D	4F)V...getUserJvMO
00000310	70	74	69	6F	6E	44	65	66	61	75	6C	74	73	01	00	08	ptionDefaults...
00000320	3C	63	6C	69	6E	69	74	3E	07	00	75	01	00	0A	53	6F	<clinit>...u...So
00000330	75	72	63	65	46	69	6C	65	01	00	1B	4C	61	75	6E	63	ourceFile...Launc
00000340	68	65	72	55	73	65	72	4A	76	6D	4F	70	74	69	6F	6E	herUserJvmOptio
00000350	73	2E	6A	61	76	61	0C	00	2A	00	2B	07	00	7D	0C	00	s.java...+...+

Offset(h): 49

Special editors

Data inspector

Binary (8 bit)	01001100
Int8	76
UInt8	76
Int16	2892
UInt16	2892
Int24	2892
UInt24	2892
Int32	1459620684
UInt32	1459620684
Int64	3195182249937740
UInt64	3195182249937740
LEB128	-52
ULEB128	76
AnsiChar / char8_t	L
WideChar / char16_t	6C
UTF-8 code point	L (U+004C)
Single (float32)	140786008064000
Double (float64)	1.57862978189598E-308
OLETIME	30/12/1899
FILETIME	16/02/1611 03:03:44
DOS date	12/10/1985
DOS time	01:26:24
DOS time & date	Invalid
time_t (32 bit)	02/04/2016 18:11:24
time_t (64 bit)	Invalid
GUID	{57000B4C-5A00-000B-5B00-5C0B}
Disassembly (x86-16)	dec sp
Disassembly (x86-32)	dec esp
Disassembly (x86-64)	or r8,[rax]

Byte order

☒ Little endian ☐ Big endian

☐ Hexadecimal basis (for integral numbers)

Overwrite

e) I denne oppgaven skal du demonstrere grunnleggende ferdigheter i shell/terminal, oppgaven består av 6 små deler som skal utføres i rekkefølge.
Først - Åpne terminal (Linux/OSX) eller kommandolinje (Windows)

Del 1 - Opprett en ny mappe/katalog som heter

'OPPG_3E'

Del 2 - Gå inn i mappen du opprettet

Del 3 - Opprett en fil som heter 'TEST.TXT' med innholdet 'TESTTESTTEST'

Del 4 - Skriv ut innholdet av filen på skjermen (ut i terminal/kommandolinje)

Del 5 - Slett filen du opprettet

Del 6 - Slett mappen/katalogen du opprettet

Du skal dokumentere løsningen med skjermbilde av terminal/kommandolinje vinduet, og alle delene skal løses med kommandoer i terminal og ikke med noen grafiske verktøy (for eksempel skal du ikke opprette en fil i del 3 ved å starte notepad – selv om du starter den fra terminal, og skrive inn teksten, det vil telle som å ha brukt et grafisk verktøy...)

```

C:\Users\snorr>cd desktop
C:\Users\snorr\Desktop>mkdir OPPG_3E
C:\Users\snorr\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is D69F-FE3E

Directory of C:\Users\snorr\Desktop

13/12/2021  17:06    <DIR>        .
13/12/2021  17:06    <DIR>        ..
13/12/2021  17:06    <DIR>        OPPG_3E
13/12/2021  12:11    <DIR>        privat
               0 File(s)                0 bytes
               4 Dir(s)  827,455,541,248 bytes free

C:\Users\snorr\Desktop>cd OPPG_3E
C:\Users\snorr\Desktop\OPPG_3E>echo TESTTESTTEST > TEST.TXT
C:\Users\snorr\Desktop\OPPG_3E>dir
Volume in drive C has no label.
Volume Serial Number is D69F-FE3E

Directory of C:\Users\snorr\Desktop\OPPG_3E

13/12/2021  17:17    <DIR>        .
13/12/2021  17:17    <DIR>        ..
13/12/2021  17:17                15 TEST.TXT
               1 File(s)                15 bytes
               2 Dir(s)  827,464,404,992 bytes free

C:\Users\snorr\Desktop\OPPG_3E>del TEST.TXT
C:\Users\snorr\Desktop\OPPG_3E>dir
Volume in drive C has no label.
Volume Serial Number is D69F-FE3E

Directory of C:\Users\snorr\Desktop\OPPG_3E

13/12/2021  17:17    <DIR>        .
13/12/2021  17:17    <DIR>        ..
               0 File(s)                0 bytes
               2 Dir(s)  827,463,303,168 bytes free

C:\Users\snorr\Desktop\OPPG_3E>cd..
C:\Users\snorr\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is D69F-FE3E

Directory of C:\Users\snorr\Desktop

13/12/2021  17:17    <DIR>        .
13/12/2021  17:17    <DIR>        ..
13/12/2021  17:17    <DIR>        OPPG_3E
13/12/2021  12:11    <DIR>        privat
               0 File(s)                0 bytes
               4 Dir(s)  827,462,307,840 bytes free

C:\Users\snorr\Desktop>del OPPG_3E
C:\Users\snorr\Desktop\OPPG_3E\*, Are you sure (Y/N)? y
C:\Users\snorr\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is D69F-FE3E

Directory of C:\Users\snorr\Desktop

13/12/2021  17:17    <DIR>        .
13/12/2021  17:17    <DIR>        ..
13/12/2021  17:17    <DIR>        OPPG_3E
13/12/2021  12:11    <DIR>        privat
               0 File(s)                0 bytes
               4 Dir(s)  827,462,885,376 bytes free

C:\Users\snorr\Desktop>rd OPPG_3E
C:\Users\snorr\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is D69F-FE3E

```

Oppgave 4. Forståelse av nettverk

a) Forklar forskjellen på en Hub, en Switch og en Router i et nettverk. Forklar hvordan de fungerer, hvilket TCP/IP lag de opererer på, og hvor i et nettverk de typisk brukes.

Navn på laget	Datamaskin	Hub	Switch	Router
Applikasjonslaget	x			
Transportlaget	x			
Nettverkslaget	x			x
Datalinjelaget	x		x	x
Fysisk	x	x	x	x

Når datamaskiner skal snakke med hverandre har vi den modellen vi kaller TCP/IP modellen som jeg har gjenskapt ovenfor. En datamaskin har en applikasjon som sender en melding ned i lagene som igjen blir pakket inn i demmes overføringsenhet. For enkelhetens grunn kan du se for deg en konvolutt blir lagt opp i en ny konvolutt fire ganger før den når det fysiske laget og blir sendt videre.

En hub har ingen andre egenskaper enn å være et knutepunkt for det fysiske laget og forsterker signalet som går i kablene som er koblet til. Den endrer ingenting av avsender og mottaker informasjon i datalinjelaget. Siden huben ikke rører på rammene som er sendt, så utvider du et nettverk og øker det vi kaller kollisjonsdomenet for sendere. Vi kommer tilbake til protokollene med 4-b.

Switch har samme egenskap som en hub at den forsterker signaler, men har en annen viktig egenskap. Den kan endre på avsender og mottaker på rammen som sendes, og hvis det trengs mellomagrede data hvis det er mye trafikk. Dette gjør at man skaper subnett som minsker kollisjonsdomenet på hovednettet.

Router har et øvrig ansvar i nettet med å fikse og endre IPv4 og/eller IPv6 adresser samt MAC adresser som benyttes på datalinje laget. Siden et nettverk har interne IP adresser og benytter ruterens sin for ekstern kommunikasjon må Router kunne endre både IP og MAC adresser. Router har ofte det vi kaller DHCP server på seg og svarer mer i dybden i 4-C hva det er.

b) Forklar hvordan Linklagsprotokoller basert på Multiple Access (MA) fungerer, forklar også hvordan CSMA protokoller løser utfordringene med Multiple Access.

I et nettverk så har vi mange former for MA protokoller. Kommer å gi eksempler litt senere, men for nå så er MA (Multipel aksess-protokoller) det et nettverk trenger for at ikke alle maskinene skal snakke over hverandre. For å gi en mer menneskelig kontekst så kan du se for deg en gruppe mennesker sitter sammen og snakker sammen i en gruppe. Så er det vanlig at det er kun en og en person som snakker og svarer. MA basere seg på samme prinsipp. I et nettverk så kan kun en maskin få sendt ut sine meldinger til riktig mottaker før noen andre kan sende. Hvis ikke danner dette interferens og meldingene må sendes på nytt pga. de første meldingene er nå forstyrret.

CSMA (Carrier Sense Multiple Access) er mer eller mindre den enkleste av mange protokoller. Her lytter maskinen på signaler, hvis det er stille så sender den ut meldingene sine. Hvis det opptatt så avventer man på et satt tidsrom og sjekker på nytt etter avgitt tid har gått. Eneste som er dumt med denne er at maskiner har avstand og to maskiner kan starte å sende signal samtidig. I denne protokollen så stopper ikke maskinene å sende hvis finner kollisjon og sløser bort tid

CSMA/CD (Collision Detection) Denne har samme egenskapene som tidligere, men i forskjell så stopper den opp umiddelbart når den finner ut at det er interferens for å spare tid.

CSMA/CA (Collision Avoidance) Dette er den siste hoved CSMA'en jeg skal raskt skrive om. Denne benyttes hovedsakelig på trådløs nett hvor ting går saktere og mye lettere å ha interferensen med andre enheter. Her er det innebygd forskjellige løsninger for at enheter ikke skal snakke oppå hverandre direkte, men har parallelle kanaler å gå igjennom. Det kan være forskjellige frekvenser på hvor man kan snakke og kan ha en nøkkel som går på rundgang som sier hvem som får snakke.

c) Forklar hvordan DHCP fungerer, og hvilken oppgave denne protokollen har i et nettverk. Hva skjer hvis en maskin som skal bruke en DHCP server ikke får svar fra den – kan man sette opp et nettverk uten å bruke DHCP?

DHCP (Dynamic Host Configuration Protocol) er en server og ofte Routeren man hjemme har dette innebygd. Denne serveren sin jobb er å gi ut hvilken IP-adresse de forskjellige enhetene på et nettverk skal ha. Siden maskiner fort kommer inn og ut i et nettverk så må serveren kunne vite hvem som har hva og gi ut ledige adresser. I andre tilfeller kan man reservere adresser for ønskede maskiner.

Man kan drive et nettverk uten en DHCP-server, men dette krevet at noen setter opp statisk alle IP-adressene som skal benyttes i det interne nettverket og har en del ekstra arbeid på seg. Hvis nye enheter skal på er det ikke plug and play som det er i det aller fleste nettverk som benytter DHCP-server. Da må man få fikset enhetsnavn opp mot en tabell og legge inn statisk IP-adresse til nye enheten.

d) Forklar hvordan SMTP protokollen (for sending av epost) fungerer. Illustrer forklaringen med en skjematisk tegning som viser dataflyt mellom klient og server.

SMTP (Simple Mail Transfer Protocol) er protokollen man benytter når man skal sende epost. Delt opp i flere deler hvor man har brukere som har sin klient kalt i dette tilfelle agenter. Disse agentene sender eposter med hjelp av SMTP til en server som lagrer alle epostene for brukeren og håndtere videresendingen til andre mailservere med samme SMTP-protokoll.

Når eposten ankommer mottaker serveren så kan bruker agenten ha to nedlastninger protokoller for e-posten som er mottatt. POP3 som lagrer kun en kopi av eposten og det er på agenten sin enhet. Meget ugunstig hvis man trenger samme epostkonto på flere enheter! Eller den mest brukte i dag som heter IMAP som er en synkroniseringsløsning. Hvor agent enheten synker fra mailserver på hva som er riktig status på innboks.

(Bilde er på neste side)

