

Oppgave 1. Generelt (35%)

a)

Micro computer – de minste av datamaskiner. Økonomiske og brukervennlige, PCer (personlige computere passer inn her)

Mini computer – kraftigere og dyrere computere. Populære blant litt større firmaer som trenger mer kraft.

Mainframe computer – store computer som kan håndtere store mengder data. Disse hører hjemme i større organisasjoner som har behov for mye lagring og høy hastighet.

Super computer – de største computerne. Disse er ekstremt dyre, men høyst effektive. Disse blir som regel brukt på nasjonalt nivå for lagring av data av f.eks. personalia i Kina.

Micro computere er enkeltbruker da de kun benyttes av en bruker av gangen.

De andre computerene som er nevnt er multibruker typer. Det vil si at flere brukere kan koble seg på computeren samtidig, som en server.

b)

Enkel-prosessering betyr at maskinvaren kan kun behandle en prosess av gangen. Maskinen kan da ikke kjøre en ny prosess før den forrige er ferdig. Det vil da si at ved

multiprosessering (multi-tasking), kan maskinen motta flere handlinger av gangen.

Multitasking har muligheten til å motta flere prosesser, starte og avbryte en prosess utfra hvilken viktighetsgrad den har og ha flere prosesser delvis behandlet, liggende i mente og ventende for å bli tatt opp igjen. Dette er noe begrenset, da en CPU-kjerne kan fortsatt kun behandle en prosess av gangen. Det vil si at jo flere kjerner, jo flere prosesser kan CPU'en utføre samtidig.

c)

En switch er en enhet som kobler sammen enheter i et data-nettverk. Disse behandler data som vi ønsker å sende gjennom data-nettverket til den enheten som datapakken er ment for.

Den viktigste forskjellen mellom en hub og en switch er at en hub oppfører seg som en form for kringkastingsenhet og vil sende ut informasjonen til alle enheter som er koblet til nettverket. En switch er derimot mer presis. Denne benytter ligger på link-layeret, det vil si at den har muligheten til å benytte seg av MAC adressene til enhetene på nettverket og kan sende informasjon til kun de enhetene som informasjonen er ment for.

d)

NAT fungerer som døren mellom det lokale nettverket og internettet. Denne koblingen fungerer som en retur-adressen for en postboks på et brev. Når en data-pakke blir sendt fra et lokalt nettverk, stempler NAT denne pakken med retur-adressen til postboksen, IKKE senderen, slik at destinasjonen vet hvor data-pakken har kommet fra og kan sende en respons tilbake. Dette vil da sørge for at returen vil komme til riktig person, men oppgir ikke den private tilkoblingen.

For eksempel. Hvis du ønsker å åpne en nettside, er det kun du som ønsker å få opp den siden på din maskin. Hvis du ikke har kartlagt pakken din med denne retur-adressen (IP adressen) vet ikke nettsiden hvem som har spurt om denne. Det ville vært dumt om alle enheter fikk opp siden du spurte om, eller at du ikke fikk den opp i det hele tatt. Derfor er det viktig at dette blir merket straks det blir sendt ut.

Dette åpner for muligheten til at private IP-adresser kan bli gjenbrukt og at private nettverk er tryggere fordi den er skjult for offentligheten.

e)

Vi må ha en kilde- og destinasjonsport i TCP og UDP er å fortelle hvilken applikasjon det kommer fra, og den destinasjonen vi ønsker å oppnå.

Se for deg at du har solgt noe på Finn.no. Når du møter selgeren for å overlevere varen og motta penger har dere avtalt på forhånd at dere skal overføre pengene via vipps. Da starter dere noe kalt en «handshake». Selger oppgir telefonnummeret som kjøper skal betale til. Dette telefonnummeret forteller da om kilden til TCP og hvor «svaret» (pengene) skal overføres til. Kjøper bekrefter da ved å lese opp navnet til vipps-kontoen og sånn, er koblingen opprettet og dere er klare for å sende det som skal bli sendt.

f)

Flytkontroll (flow) er en måte for mottakeren å håndtere flyten av trafikken fra senderen. Mens metningskontroll (congestion) er en trafikken fra senderen til nettverket kontrollert. Det er TCP som tar i bruk disse to kontrollene.

g)

Når TCP har sendt informasjon til mottakeren, tar den regelmessig en pause hvor den venter på et svar fra mottakeren om at pakken er levert, før den fortsetter igjen.

For eksempel.

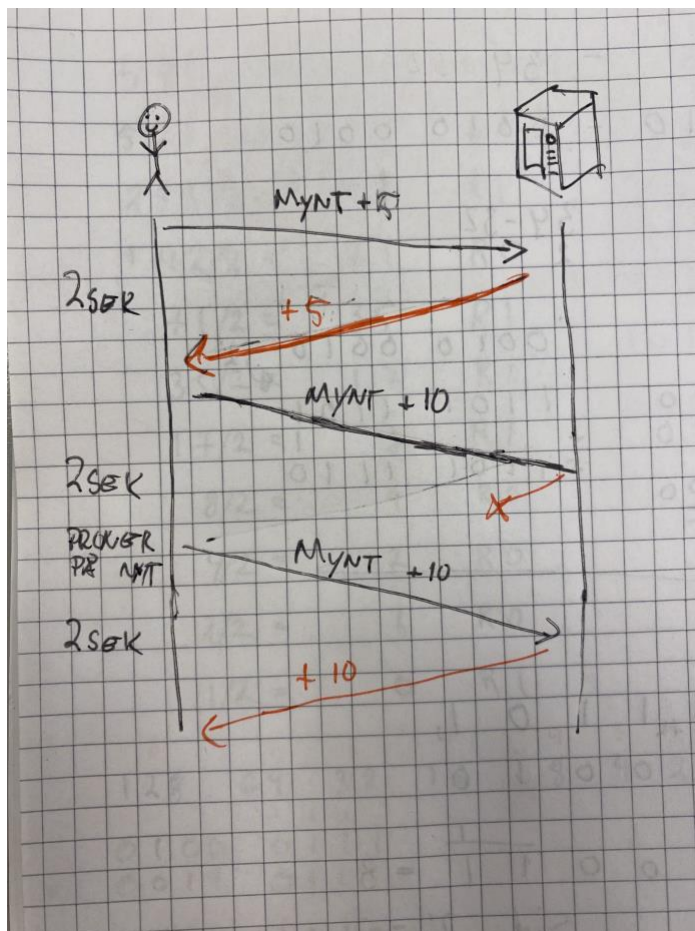
Du skal kjøpe en brus i en brus-automat med mynter. Når du putter på penger, oppdaterer verdien seg og du vet hvor mye du har puttet på. Du putter på en mynt, venter til den oppdaterer seg før du putter på en ny.

Hva skjer når denne mynten ikke blir registrert av maskinen?

Du har ventet så lenge du mener det burde tatt før den skulle oppdatert seg. Du tar da mynten ut av retur-hullet og sender putter den på nytt for den ble ikke mottatt forrige gang.

På samme måte sjekker TCP om at data er mottatt ved at mottaker sender en bekreftelse.

Hvis den ikke får dette innen rimelig tid vil den forsøke å sende forrige pakke på nytt.



BESKRIVELSE:

Bruker putter på 5kr,-

Venteperiode 2 sek.

Maskin oppdaterer display med +5.

Bruker putter på 10kr,-

Venteperiode på 2 sek.

Maskin oppdateres ikke.

Bruker prøver på nytt med samme pakke på 10kr,-

Maskin oppdater display med +10

Oppgave 2. Formater and binære tall (35%)

a)

9	8	7	6	5	4	3	2	1	0
512	256	128	64	32	16	8	4	2	1

$$\begin{array}{r}
 865_{10} \rightarrow ?_2 \\
 865 - 512 \quad (2^9) \quad \uparrow \\
 353 - 256 \quad (2^8) \\
 97 - 64 \quad (2^6) \\
 33 - 32 \quad (2^5) \\
 1 - 1 \quad (2^0)
 \end{array}$$

$$\underline{0000 \ 0011 \ 0110 \ 0001}_2$$

$$\begin{array}{r}
 974_{10} \rightarrow ?_{16} \\
 974 \div 16 = 60 \quad R 14 = E \uparrow \\
 60 \div 16 = 3 \quad R 12 = C \\
 3 \div 16 = 0 \quad R 3
 \end{array}$$

$$\underline{974_{10} \rightarrow 3CE_{16}}$$

$$\begin{array}{l}
 60.875 \\
 3.75
 \end{array}$$

$$\underline{865_{10} = 0000 \ 0011 \ 0110 \ 0001_2}$$

$$\underline{974_{10} = 3CE_{16}}$$

b)

Jeg tar utgangspunkt i at det er 16-bit presisjon, som presisert i forrige oppgave, siden det ikke er opplyst om noe annet i oppgave B.

$$\begin{array}{r}
 1111 \ 1 \\
 b) \ 1010 \ 1101_2 \\
 + 0111 \ 0110_2 \\
 = \underline{10010 \ 0011}_2
 \end{array}$$

$$\begin{array}{r}
 1010 \ 1101_2 \\
 \text{xor} \\
 0111 \ 0110_2 \\
 = \underline{1101 \ 1011}_2
 \end{array}$$

IF BOTH BITS
ARE THE SAME
PUT '0'
ELSE 1

$$\begin{array}{r}
 1010 \ 1101 \ 0111_2 \\
 - 0000 \ 0111 \ 0110_2 \\
 = \underline{1010 \ 0110 \ 0001}_2
 \end{array}$$

c)

Et sterkt passord er et passord som tar lang tid å finne ut av ved «brute-force». Dette passordet inneholder gjerne spesielle karakterer, en kombinasjon av upper-case og lower-case, tall og har ingen meningsfull rekkefølge eller et betydningsfullt ord.

F.eks. *Passord123* = Dårlig, *T1g€r_ch4!R12* = Sterkt

Vi kan sammenligne dem ved å se på mulige karakterer som er brukt og lengden på passordet.

Passord 1 er 8 karakterer langt og a-z = 26 bokstaver.

Passord 2 er 4 karakterer langt og har a-å = 29 bokstaver og 0-9 = 10

Ganger vi dette sammen ser vi at passord 1 har 208 muligheter og passord 2 har 1160 mulige kombinasjoner og er derfor sterkere.

d)

Alice til Bob:

Never tell anyone that my password is the first 2 words!

Bob til Alice:

You did it right now!

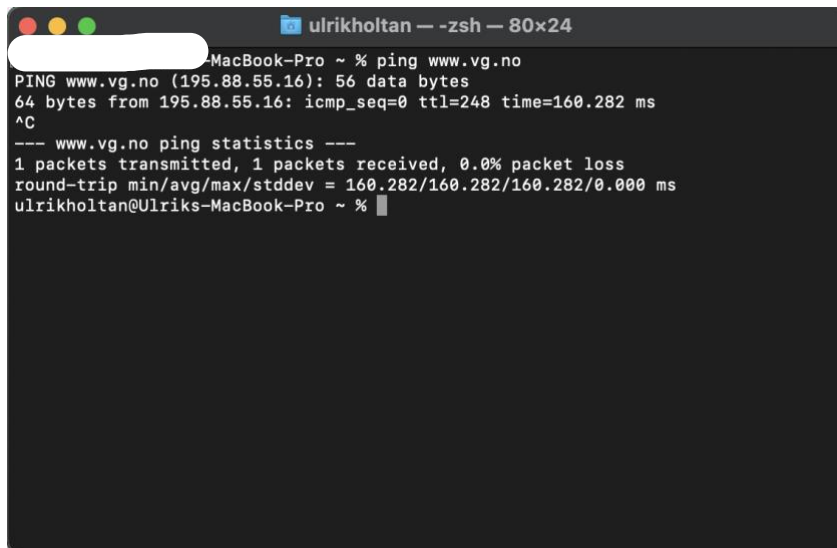
e)

The image shows two handwritten hexadecimal addition problems on a grid background. The first problem adds 1 to 0xAB0, resulting in 0xAB1. The second problem adds 1 to 0xA0F5, resulting in 0xA0FE, with a carry of 9 noted.

$0 \times 0AB0$	$B + 1 = 10 + 1 = 11 \Rightarrow C$
$+ 0 \times 011C$	$A + 1 = 9 + 1 = 10 \Rightarrow B$
$= \underline{0 \times 0B2C}$	
$0 \times A0F5$	$F + A = 15 + 10 = 1 \text{ CARRY } 9$
$+ 0 \times 01A9$	
$= \underline{0 \times A299}$	

Oppgave 3. Praktiske oppgave (30%)

1)

A terminal window titled 'ulrikholtan - zsh - 80x24' on a MacBook-Pro. The user has entered the command 'ping www.vg.no'. The output shows a successful ping to 195.88.55.16 with 56 data bytes and a round-trip time of 160.282 ms. The user then presses Ctrl-C (^C) and the terminal shows the ping statistics: 1 packet transmitted, 1 packet received, 0.0% packet loss, and a round-trip time of 160.282/160.282/160.282/0.000 ms.

```
ulrikholtan@MacBook-Pro ~ % ping www.vg.no
PING www.vg.no (195.88.55.16): 56 data bytes
64 bytes from 195.88.55.16: icmp_seq=0 ttl=248 time=160.282 ms
^C
--- www.vg.no ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 160.282/160.282/160.282/0.000 ms
ulrikholtan@Ulriks-MacBook-Pro ~ %
```

Jeg pinget www.vg.no. Med : ping www.vg.no

Jeg fikk tilbake IP-adressen 195.88.55.16, 56 data bytes og det tok 160.282 ms for pakken å bli sendt, mottatt, returnert og mottatt av min datamaskin igjen.

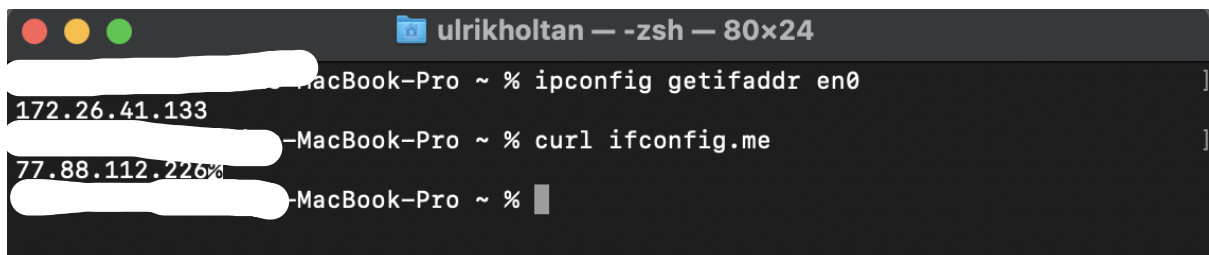
2.

Lokal IP ved: ipconfig getifaddr en0

Resultat: 172.26.41.133

Offentlig IP ved: curl ifconfig.me

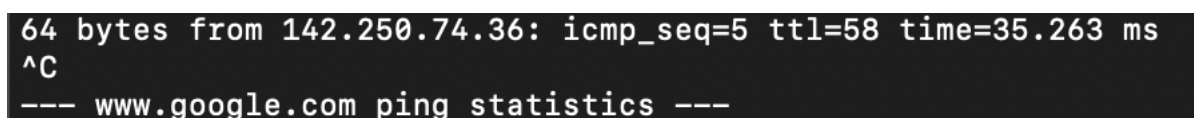
Resultat: 77.88.122.226

A terminal window titled 'ulrikholtan - zsh - 80x24' on a MacBook-Pro. The user enters 'ipconfig getifaddr en0' and the output is '172.26.41.133'. Then the user enters 'curl ifconfig.me' and the output is '77.88.112.226'. The user then presses Ctrl-C (^C) and the terminal shows the prompt again.

```
ulrikholtan@MacBook-Pro ~ % ipconfig getifaddr en0
172.26.41.133
ulrikholtan@MacBook-Pro ~ % curl ifconfig.me
77.88.112.226
ulrikholtan@MacBook-Pro ~ %
```

d)

Google sin IP er 142.250.74.36, funnet i terminal.

A terminal window showing the output of a ping command to Google. The output shows a successful ping to 142.250.74.36 with 64 data bytes and a round-trip time of 35.263 ms. The user then presses Ctrl-C (^C) and the terminal shows the ping statistics: 1 packet transmitted, 1 packet received, 0.0% packet loss, and a round-trip time of 35.263/35.263/35.263/0.000 ms.

```
64 bytes from 142.250.74.36: icmp_seq=5 ttl=58 time=35.263 ms
^C
--- www.google.com ping statistics ---
```


f)

Wi-Fi: en0

Apply a display filter ... <*>

No.	Time	Source	Destination	Protocol	Length	Info
61	8.510488	172.26.41.133	162.159.133.234	TCP	66	61933 → 443 [ACK] Seq=55 Ack=34 Win=2047 Len=0 TSval=3152627757 TSecr=
62	8.520339	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=3/768, ttl=58 (request in 59)
63	9.448003	172.26.41.133	142.250.74.36	ICMP	98	Echo (ping) request id=0xac24, seq=4/1024, ttl=64 (reply in 64)
64	9.461283	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=4/1024, ttl=58 (request in 63)
65	10.031709	172.26.41.133	224.0.0.251	MDNS	328	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D4
66	10.031766	fe80::10a9:4b77:7e...	ff02::fb	MDNS	348	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D4
67	10.273885	172.26.41.133	172.26.127.255	UDP	86	57621 → 57621 Len=44
68	10.453186	172.26.41.133	142.250.74.36	ICMP	98	Echo (ping) request id=0xac24, seq=5/1280, ttl=64 (reply in 69)
69	10.488277	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=5/1280, ttl=58 (request in 68)
70	10.571897	172.26.41.133	34.247.209.155	TCP	54	63749 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
71	10.642858	34.247.209.155	172.26.41.133	TCP	66	[TCP ACKed unseen segment] 443 → 63749 [ACK] Seq=1 Ack=2 Win=300 Len=0
72	11.080636	172.26.41.133	224.0.0.251	MDNS	328	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D4
73	11.080732	fe80::10a9:4b77:7e...	ff02::fb	MDNS	348	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D4
74	11.455847	172.26.41.133	142.250.74.36	ICMP	98	Echo (ping) request id=0xac24, seq=6/1536, ttl=64 (reply in 75)
75	11.550570	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=6/1536, ttl=58 (request in 74)
76	12.960149	172.26.41.133	23.38.201.146	TCP	54	63761 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
77	12.960252	172.26.41.133	23.38.201.146	TCP	54	63760 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
78	12.965992	23.38.201.146	172.26.41.133	TCP	66	[TCP ACKed unseen segment] 443 → 63760 [ACK] Seq=1 Ack=2 Win=501 Len=0

> Frame 68: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
 > Ethernet II, Src: Apple_72:6d:4d (f4:d4:88:72:6d:4d), Dst: PaloAlto_e0:40:
 > Internet Protocol Version 4, Src: 172.26.41.133, Dst: 142.250.74.36

Internet Control Message Protocol

Type: 8 (Echo (ping) request)
 Code: 0
 Checksum: 0xa9f2 [correct]
 [Checksum Status: Good]
 Identifier (BE): 44068 (0xac24)
 Identifier (LE): 9388 (0x24ac)
 Sequence Number (BE): 5 (0x0005)
 Sequence Number (LE): 1280 (0x0500)
 [Response frame: 69]
 Timestamp from icmp data: Dec 20, 2022 11:20:44.706232000 CET
 [Timestamp from icmp data (relative): 0.000100000 seconds]

> Data (48 bytes)

Internet Control Message Protocol (icmp), 64 bytes

Packets: 81 - Displayed: 81 (100.0%) - Dropped: 0 (0.0%) - Profile: Default

g)

Data:

08090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f202122232425262728292a2b...

Wireshark packet capture showing an ICMP Echo (ping) request. The packet is highlighted in blue. The packet details pane shows the ICMP Echo (ping) request structure with fields like Identifier, Sequence Number, and Sequence Number (LE). The packet bytes pane shows the raw data of the packet.

h)

Wireshark packet capture showing an ICMP Echo (ping) reply. The packet is highlighted in blue. The packet details pane shows the ICMP Echo (ping) reply structure with fields like Identifier, Sequence Number, and Sequence Number (LE). The packet bytes pane shows the raw data of the packet.

i)

Innholdet i pakken som ble sendt fra min IP og fra Google sin IP er identiske.

Data:

08090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f202122232425262728292a2b...

Wi-Fi: en0

Apply a display filter: <X>

No.	Time	Source	Destination	Protocol	Length	Info
61	8.510488	172.26.41.133	162.159.133.234	TCP	66	61933 → 443 [ACK] Seq=55 Ack=34 Win=2047 Len=0 TSval=3152627757 TSecr=934437218
62	8.520339	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=3/768, ttl=58 (request in 59)
63	9.448003	172.26.41.133	142.250.74.36	ICMP	98	Echo (ping) request id=0xac24, seq=4/1024, ttl=64 (reply in 64)
64	9.461283	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=4/1024, ttl=58 (request in 63)
65	10.831709	172.26.41.133	224.0.0.251	MDNS	328	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D488726D40@Ulrik's MacBook Pro._raop._tcp.local
66	10.831766	fe80::10a9:4b77:7e...	ff02::fb	MDNS	348	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D488726D40@Ulrik's MacBook Pro._raop._tcp.local
67	10.273885	172.26.41.133	172.26.127.255	UDP	86	57621 → 57621 Len=44
68	10.453186	172.26.41.133	142.250.74.36	ICMP	98	Echo (ping) request id=0xac24, seq=5/1280, ttl=64 (reply in 69)
69	10.488277	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=5/1280, ttl=58 (request in 68)
70	10.571897	172.26.41.133	34.247.209.155	TCP	54	63749 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
71	10.642650	34.247.209.155	172.26.41.133	TCP	66	[TCP ACKed unseen segment] 443 → 63749 [ACK] Seq=1 Ack=2 Win=300 Len=0 TSval=3795180546 TSecr=1685269650
72	11.000630	172.26.41.133	224.0.0.251	MDNS	328	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D488726D40@Ulrik's MacBook Pro._raop._tcp.local
73	11.000732	fe80::10a9:4b77:7e...	ff02::fb	MDNS	348	Standard query response 0x0000 TXT, cache flush NSEC, cache flush F4D488726D40@Ulrik's MacBook Pro._raop._tcp.local
74	11.455847	172.26.41.133	142.250.74.36	ICMP	98	Echo (ping) request id=0xac24, seq=6/1536, ttl=64 (reply in 75)
75	11.558570	142.250.74.36	172.26.41.133	ICMP	98	Echo (ping) reply id=0xac24, seq=6/1536, ttl=58 (request in 74)
76	12.960149	172.26.41.133	23.38.201.146	TCP	54	63761 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
77	12.960252	172.26.41.133	23.38.201.146	TCP	54	63760 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
78	12.965992	23.38.201.146	172.26.41.133	TCP	66	[TCP ACKed unseen segment] 443 → 63760 [ACK] Seq=1 Ack=2 Win=501 Len=0 TSval=2287072261 TSecr=4015167664
79	12.965994	23.38.201.146	172.26.41.133	TCP	66	[TCP ACKed unseen segment] 443 → 63761 [ACK] Seq=1 Ack=2 Win=501 Len=0 TSval=2287072261 TSecr=1522079126
80	13.461520	172.26.41.133	210.239.34.36	TCP	54	63759 → 443 [ACK] Seq=1 Ack=1 Win=2048 Len=0
81	13.473443	210.239.34.36	172.26.41.133	TCP	66	[TCP ACKed unseen segment] 443 → 63759 [ACK] Seq=1 Ack=2 Win=269 Len=0 TSval=72302048 TSecr=1327657843

> Frame 69: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, id 0
> Ethernet II, Src: PaloAlto-e0:40:45 (08:0c:25:e0:40:45), Dst: Apple_72:6d:4d (f4:d4:88:72:6d:4d)
> Internet Protocol Version 4, Src: 142.250.74.36, Dst: 172.26.41.133
v Internet Control Message Protocol
Type: 0 (Echo (ping) reply)
Code: 0
Checksum: 0xb1f2 [correct]
[Checksum Status: Good]
Identifier (BE): 44868 (0xac24)
Identifier (LE): 9380 (0x24ac)
Sequence Number (BE): 5 (0x0005)
Sequence Number (LE): 1280 (0x0500)
[Request frame: 68]
[Response time: 35.091 ms]
Timestamp from icmp data: Dec 20, 2022 11:20:44.706232000 CET
[Timestamp from icmp data (relative): 0.035191000 seconds]
v Data (48 bytes)
Data: 08090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f202122232425262728292a2b...
[Length: 48]

Data (data.data), 48 bytes

Packets: 81 (100.0%) - Dropped: 0 (0.0%)

Profile: Default