University of
South-Eastern Norway

www.usn.no

IIA2017: Industrial IT

Assignment 1: Data Communication

Isak Skeie, 245362

Faculty of Technology, Natural sciences and Maritime Sciences
Campus Porsgrunn

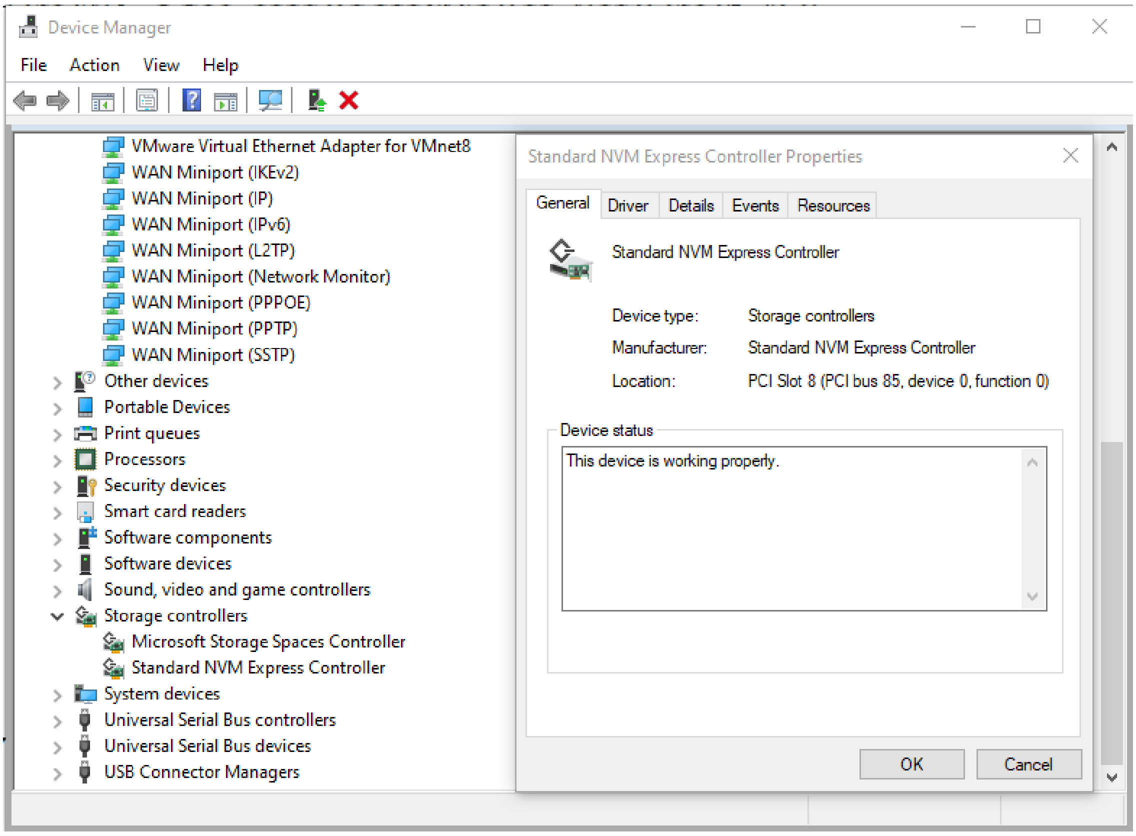
Contents

1 Introduction

As stated in the assignment specification. The purpose is to get experience and become familiar with I/O devices, setup of network devices, and to test the network on a personal computer. This is done by looking through device settings on windows, experimenting with the ping utility, as well as performing tasks surrounding the OSI model. The tasks are performed with a ThinkPad P15 laptop, supplied by work. During the assignment the laptop have been used at different locations. This gives inconsistent answers to the network and device tasks. As the laptop connects to different networks, and in some cases with a docking station.

1.1 System Information

Figure 1 shows the Interface protocol NVM being used between the storage device and the motherboard. Figure 2 shows the us the size of the storage device being 475.69 GB, with 33% of this being available.



Standard NVM Express Controller

Device type: Storage controllers

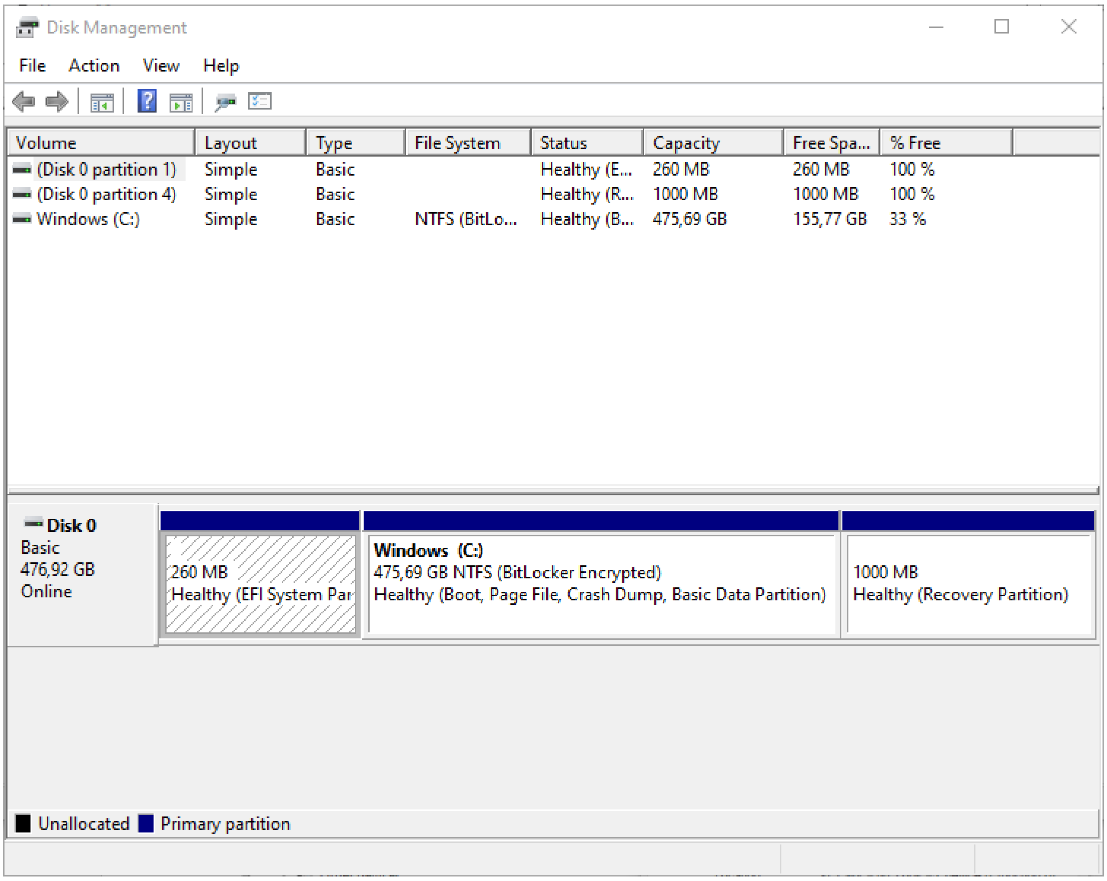
Manufacturer: Standard NVM Express Controller

Location: PCI Slot 8 (PCI bus 85, device 0, function 0)

Device status

This device is working properly.

Figure 1: Device Manager; interface between motherboard and storage device



Volume	Layout	Type	File System	Status	Capacity	Free Spa...	% Free
(Disk 0 partition 1)	Simple	Basic		Healthy (E...	260 MB	260 MB	100 %
(Disk 0 partition 4)	Simple	Basic		Healthy (R...	1000 MB	1000 MB	100 %
Windows (C:)	Simple	Basic	NTFS (BitLo...	Healthy (B...	475,69 GB	155,77 GB	33 %

Disk 0
Basic
476,92 GB
Online

260 MB
Healthy (EFI System Par

Windows (C:)
475,69 GB NTFS (BitLocker Encrypted)
Healthy (Boot, Page File, Crash Dump, Basic Data Partition)

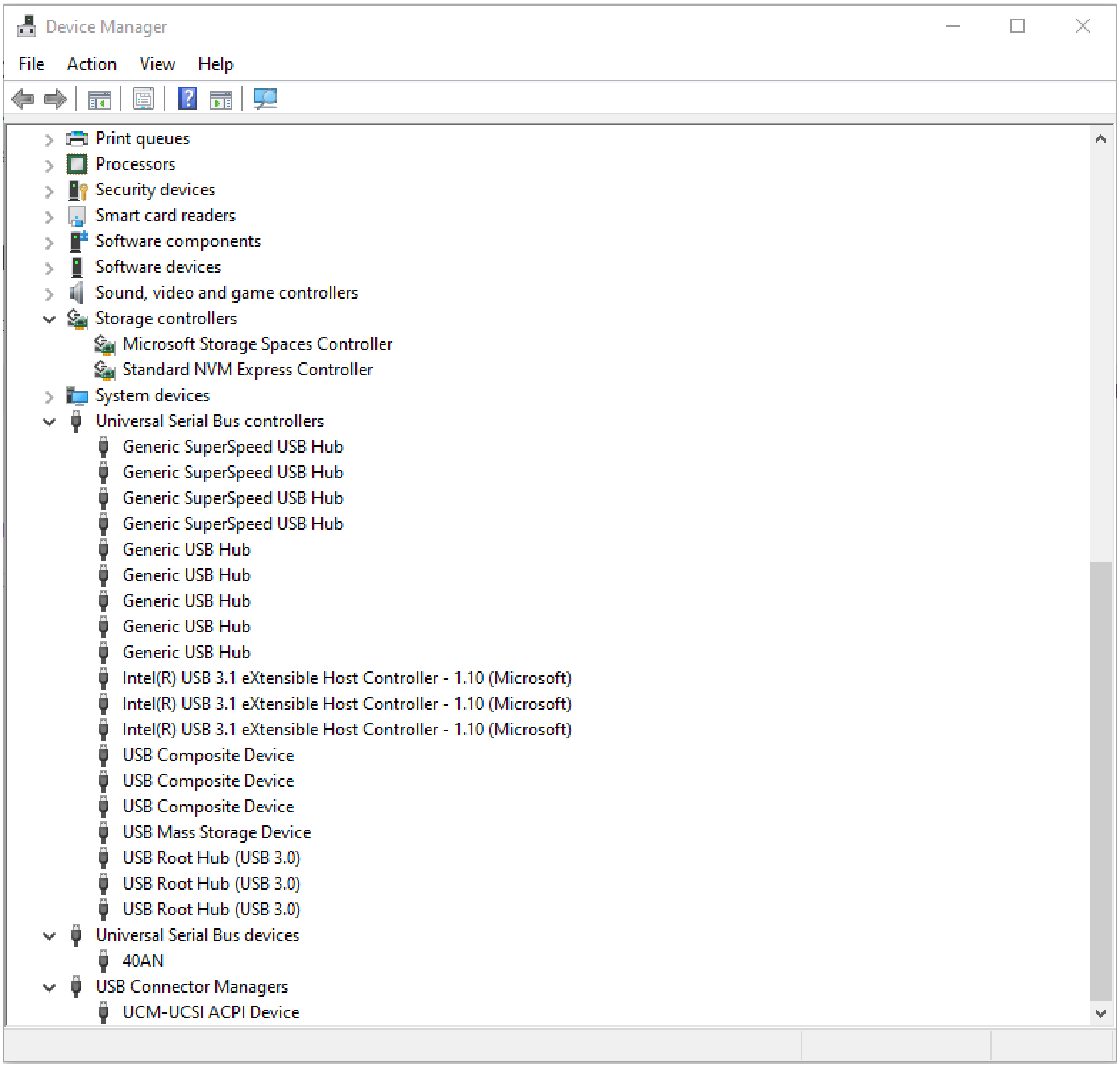
1000 MB
Healthy (Recovery Partition)

■ Unallocated ■ Primary partition

Figure 2: Disk Management

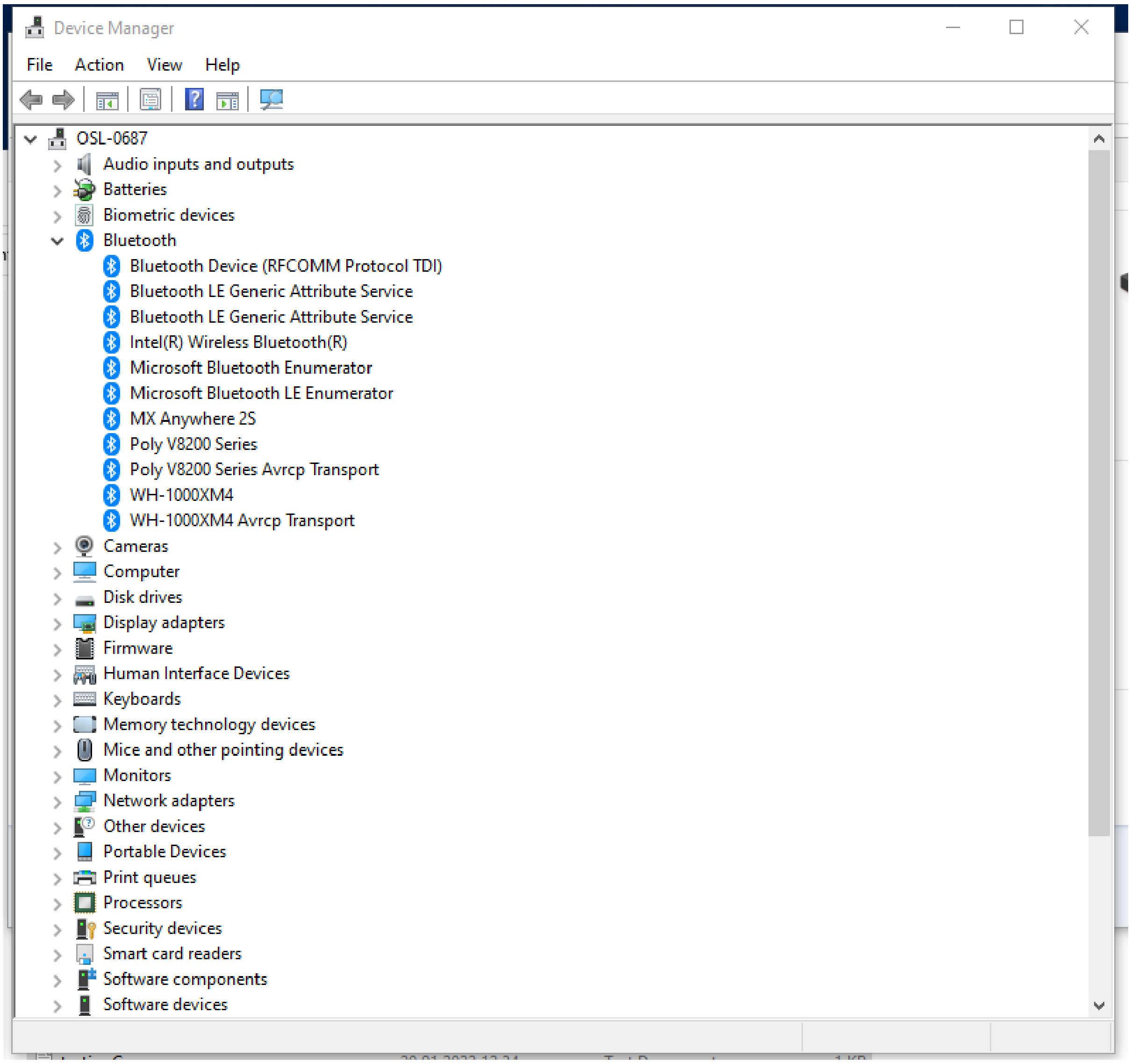
An extensive list of serial ports is shown in figure 3. The large number of serial ports are due to the dock the laptop is connected to at work. Here two additional screens are connected, which have two additional USB-ports each.

Figure 3 shows a list of Bluetooth devices that’s been connected to the laptop.



```
> Print queues
> Processors
> Security devices
> Smart card readers
> Software components
> Software devices
> Sound, video and game controllers
v Storage controllers
    Microsoft Storage Spaces Controller
    Standard NVM Express Controller
> System devices
v Universal Serial Bus controllers
    Generic SuperSpeed USB Hub
    Generic SuperSpeed USB Hub
    Generic SuperSpeed USB Hub
    Generic SuperSpeed USB Hub
    Generic USB Hub
    Generic USB Hub
    Generic USB Hub
    Generic USB Hub
    Generic USB Hub
    Intel(R) USB 3.1 eXtensible Host Controller - 1.10 (Microsoft)
    Intel(R) USB 3.1 eXtensible Host Controller - 1.10 (Microsoft)
    Intel(R) USB 3.1 eXtensible Host Controller - 1.10 (Microsoft)
    USB Composite Device
    USB Composite Device
    USB Composite Device
    USB Mass Storage Device
    USB Root Hub (USB 3.0)
    USB Root Hub (USB 3.0)
    USB Root Hub (USB 3.0)
v Universal Serial Bus devices
    40AN
v USB Connector Managers
    UCM-UCSI ACPI Device
```

Figure 3: Overview of serial ports



```
OSL-0687
    Audio inputs and outputs
    Batteries
    Biometric devices
    Bluetooth
        Bluetooth Device (RFCOMM Protocol TDI)
        Bluetooth LE Generic Attribute Service
        Bluetooth LE Generic Attribute Service
        Intel(R) Wireless Bluetooth(R)
        Microsoft Bluetooth Enumerator
        Microsoft Bluetooth LE Enumerator
        MX Anywhere 2S
        Poly V8200 Series
        Poly V8200 Series Avrcp Transport
        WH-1000XM4
        WH-1000XM4 Avrcp Transport
    Cameras
    Computer
    Disk drives
    Display adapters
    Firmware
    Human Interface Devices
    Keyboards
    Memory technology devices
    Mice and other pointing devices
    Monitors
    Network adapters
    Other devices
    Portable Devices
    Print queues
    Processors
    Security devices
    Smart card readers
    Software components
    Software devices
```

Figure 4: List of Bluetooth devices

2 Network information

<Start each chapter with a short introduction about what the chapter is about.>

2.1 Theory Description

The ping-test utility is used for network testing and diagnostics. It uses the Internet Control Message protocol in the network layer to send Ip-Packets with an echo request. The echo request makes sure several packets are sent back and forth. This makes it possible to confirm if a host in a network is active, the speed/response -time of data travel, and the packet loss in the connection. [1] [2]

2.2 Network info

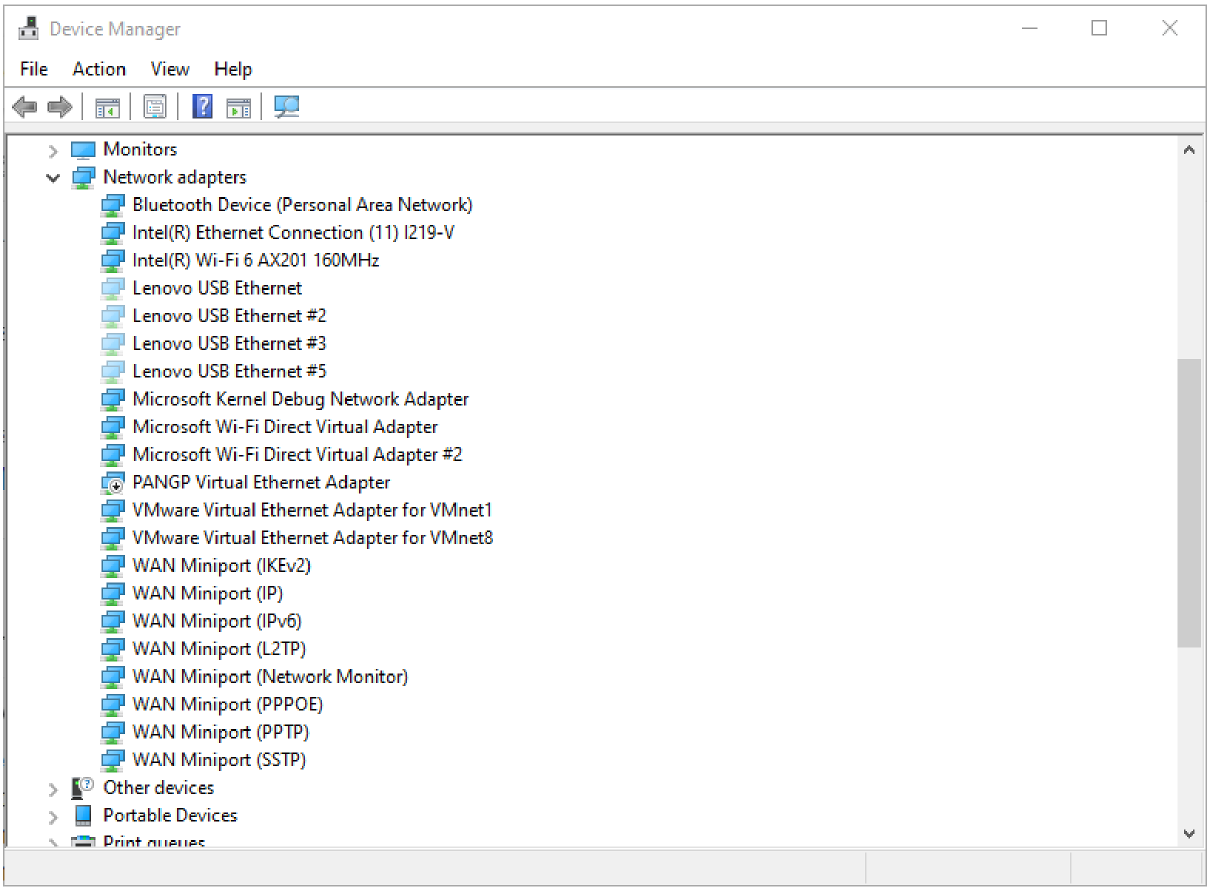
Figure 5 shows a list of network devices from Device Manager. The large number of network devices are due to the docking station at work.

Figure 6 shows a PowerShell window displaying the output from the command *ipconfig -all*. Theres several physical addresses, one for each network device, with the one connected to the local network being the one of interest. At the time of sending the ipconfig command the laptop was connected to the local network through Ethernet adapter Ethernet 3, which has a MAC-Adress of 08-3A-88-5C-D8-2A.

Because of the IP-Address we can assume the connection is using the TCP/IP protocol. The first four bits in the IP-address tells us it’s a class B address, with the subnet-mask being 255.255.255.0. The ipconfig /all command also gives us the IP-address for the DHCP-server (Dynamic Host Configuration protocol), this implies that the address assigned to the ethernet adapter is dynamic.

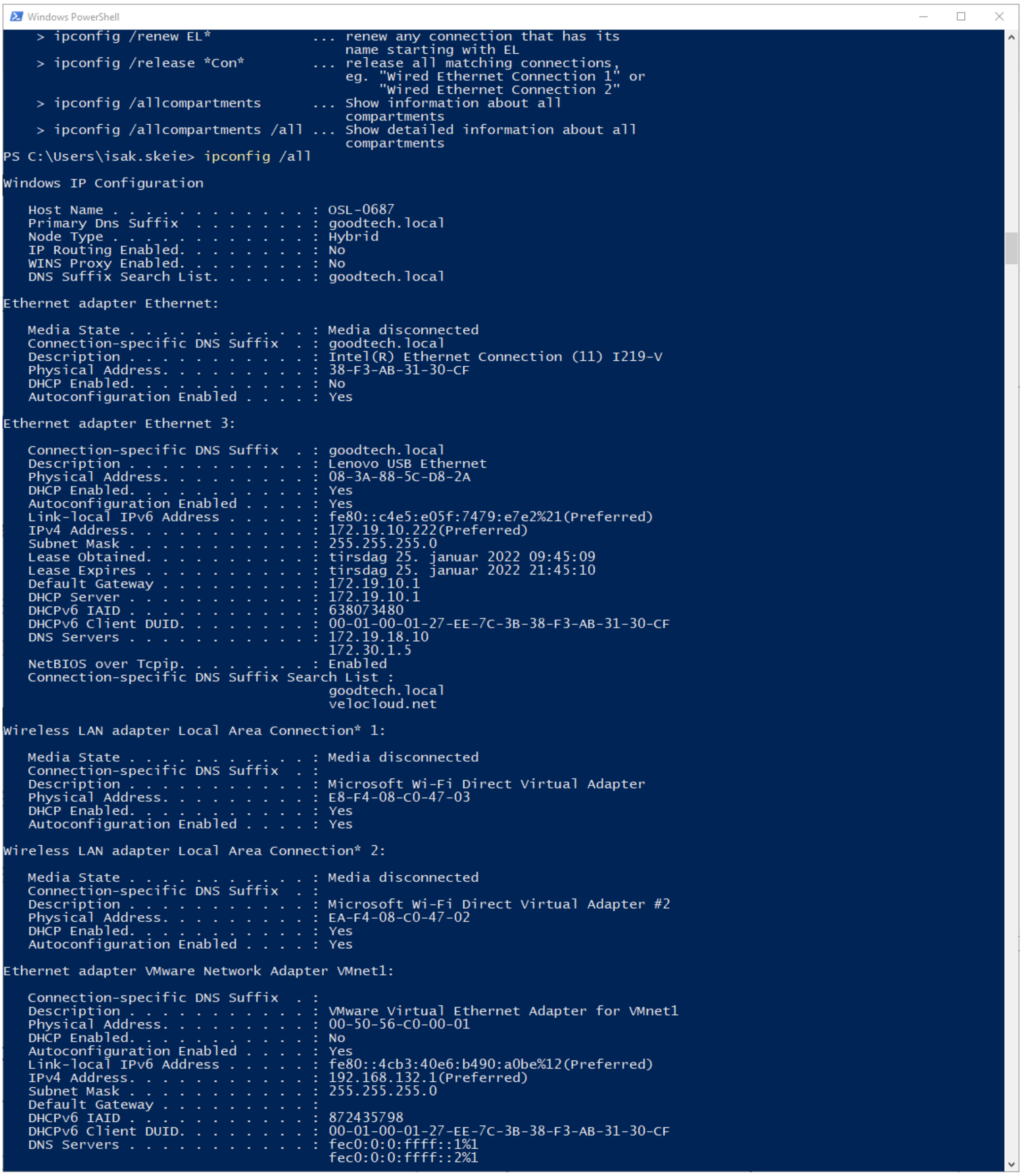
Other than TCP/IP, other protocols on the computer are UDP for real time data transfer, POP for receiving emails, smpt sending emails, http/https for web use, FTP. [3] [4]

From figure 6, the information provided by ipconfig /all in PowerShell, under Ethernet adapter Ethernet 3 the IP6-adress is listed as Link-Local IPv6 address with the address being fe80::c4e5:e05f:7479:e7e2%21. This tells us that the computer supports IPV6.



Monitors
Network adapters
Bluetooth Device (Personal Area Network)
Intel(R) Ethernet Connection (11) I219-V
Intel(R) Wi-Fi 6 AX201 160MHz
Lenovo USB Ethernet
Lenovo USB Ethernet #2
Lenovo USB Ethernet #3
Lenovo USB Ethernet #5
Microsoft Kernel Debug Network Adapter
Microsoft Wi-Fi Direct Virtual Adapter
Microsoft Wi-Fi Direct Virtual Adapter #2
PANGP Virtual Ethernet Adapter
VMware Virtual Ethernet Adapter for VMnet1
VMware Virtual Ethernet Adapter for VMnet8
WAN Miniport (IKEv2)
WAN Miniport (IP)
WAN Miniport (IPv6)
WAN Miniport (L2TP)
WAN Miniport (Network Monitor)
WAN Miniport (PPPOE)
WAN Miniport (PPTP)
WAN Miniport (SSTP)
Other devices
Portable Devices
Print queues

Figure 5: List of network devices from Device Manager



```
    > ipconfig /renew EL*            ... renew any connection that has its
                                         name starting with EL
    > ipconfig /release *Con*        ... release all matching connections,
                                         eg. "Wired Ethernet Connection 1" or
                                             "Wired Ethernet Connection 2"
    > ipconfig /allcompartments      ... Show information about all
                                         compartments
    > ipconfig /allcompartments /all ... Show detailed information about all
                                         compartments
PS C:\Users\isak.skeie> ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : OSL-0687
   Primary Dns Suffix  . . . . . . . : goodtech.local
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : goodtech.local

Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : goodtech.local
   Description . . . . . . . . . . . : Intel(R) Ethernet Connection (11) I219-V
   Physical Address. . . . . . . . . : 38-F3-AB-31-30-CF
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter Ethernet 3:

   Connection-specific DNS Suffix  . : goodtech.local
   Description . . . . . . . . . . . : Lenovo USB Ethernet
   Physical Address. . . . . . . . . : 08-3A-88-5C-D8-2A
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::c4e5:e05f:7479:e7e2%21(Preferred)
   IPv4 Address. . . . . . . . . . . : 172.19.10.222(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : tirsdag 25. januar 2022 09:45:09
   Lease Expires . . . . . . . . . . : tirsdag 25. januar 2022 21:45:10
   Default Gateway . . . . . . . . . : 172.19.10.1
   DHCP Server . . . . . . . . . . . : 172.19.10.1
   DHCPv6 IAID . . . . . . . . . . . : 638073480
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-27-EE-7C-3B-38-F3-AB-31-30-CF
   DNS Servers . . . . . . . . . . . : 172.19.18.10
                                       172.30.1.5
   NetBIOS over Tcpip. . . . . . . . : Enabled
   Connection-specific DNS Suffix Search List :
                                       goodtech.local
                                       velocloud.net

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter
   Physical Address. . . . . . . . . : E8-F4-08-C0-47-03
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter #2
   Physical Address. . . . . . . . . : EA-F4-08-C0-47-02
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VMware Virtual Ethernet Adapter for VMnet1
   Physical Address. . . . . . . . . : 00-50-56-C0-00-01
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::4cb3:40e6:b490:a0be%12(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.132.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 872435798
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-27-EE-7C-3B-38-F3-AB-31-30-CF
   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                       fec0:0:0:ffff::2%1
```

Figure 6: Local network information in powershell

2.3 Wireless

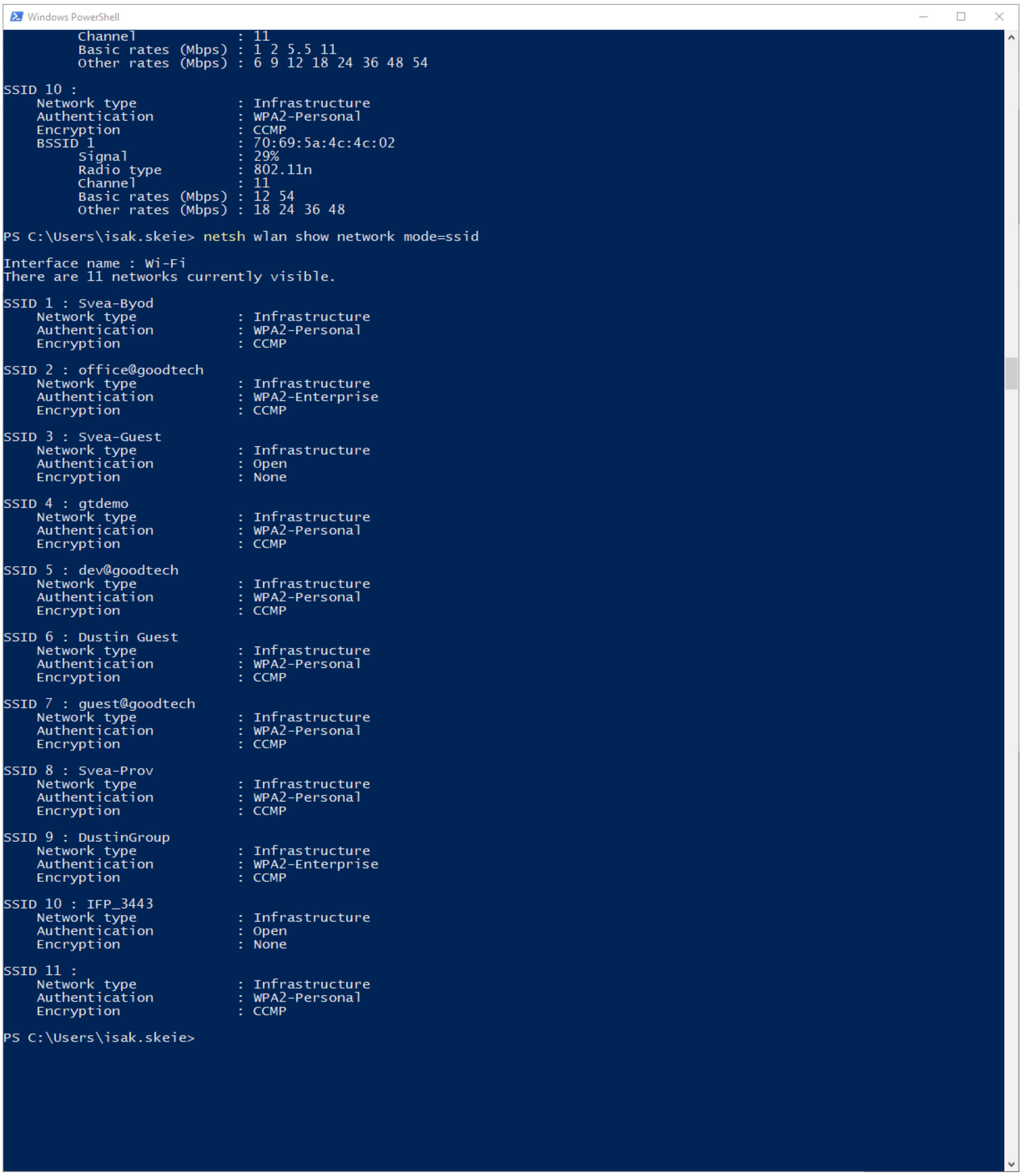
In figure 1.2 from the assignment paper. The first option seems to be the ideal one, with the wireless signal being the strongest, and having the fewest connections.

From Figure 1.2, its unclear if a password is necessary. This is due to the Encryption being unknown for the network.

In figure 1.2 from the assignment paper, there are no other networks shown because the network module being used by the computer does not extend past the adjacent layer of the OSI-Model. With the network-module being layer 2 (Data link layer), and the access points being layer 3 (Network layer).

Assume owners, location, security. In the list in Figure 1.3 in the assignment paper. We can assume Thamm is the name of the owner of the network, as it is a computer-to-computer connection. Due to the bad signal strength, we can assume there’s a significant distance between the nodes, we can also see that the connection is unsecured which is unwise, especially to a computer. The network CM have security, but not specified which type. The signal strength is very good, indicating the access point is close. AB Stargate1 have the same properties as CM, with a lower signal strength. AB ext. and AB Stargate have the same properties as CM and AB Stargate1 but with a bad signal quality, indicating they are further away. AB Skywalker have a medium to low signal strength, and WPA security. WPA is better than the security of the other options, something like WPA2 should have been employed. [5]

In figure 7 the command *netsh wlan show network mode-ssid* is used to list the available wireless networks in PowerShell.

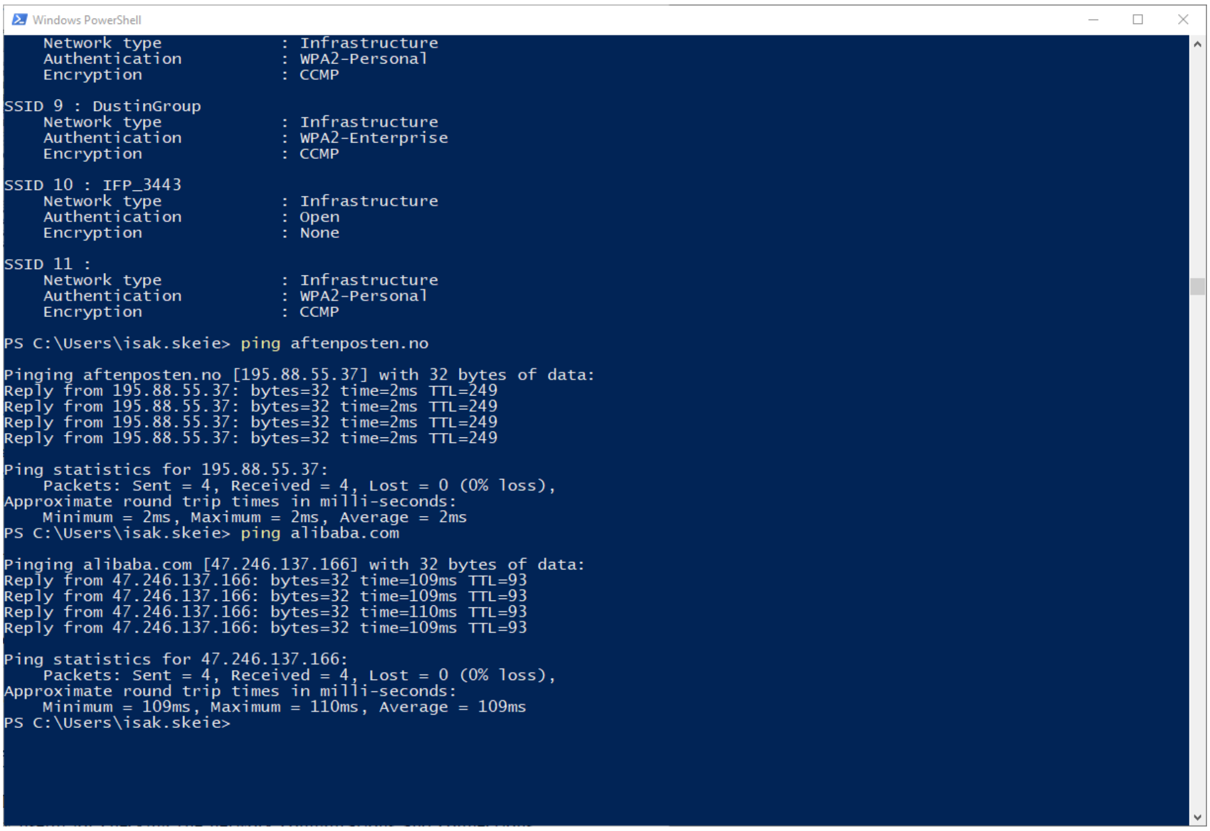


```
         Channel            : 11
         Basic rates (Mbps) : 1 2 5.5 11
         Other rates (Mbps) : 6 9 12 18 24 36 48 54

SSID 10 :
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP
    BSSID 1                 : 70:69:5a:4c:4c:02
         Signal             : 29%
         Radio type         : 802.11n
         Channel            : 11
         Basic rates (Mbps) : 12 54
         Other rates (Mbps) : 18 24 36 48

PS C:\Users\isak.skeie> netsh wlan show network mode=ssid

Interface name : Wi-Fi
There are 11 networks currently visible.

SSID 1 : Svea-Byod
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 2 : office@goodtech
    Network type            : Infrastructure
    Authentication          : WPA2-Enterprise
    Encryption              : CCMP

SSID 3 : Svea-Guest
    Network type            : Infrastructure
    Authentication          : Open
    Encryption              : None

SSID 4 : gtdemo
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 5 : dev@goodtech
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 6 : Dustin Guest
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 7 : guest@goodtech
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 8 : Svea-Prov
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 9 : DustinGroup
    Network type            : Infrastructure
    Authentication          : WPA2-Enterprise
    Encryption              : CCMP

SSID 10 : IFP_3443
    Network type            : Infrastructure
    Authentication          : Open
    Encryption              : None

SSID 11 :
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

PS C:\Users\isak.skeie>
```

Figure 7: List of available wireless connections

3 Network testing

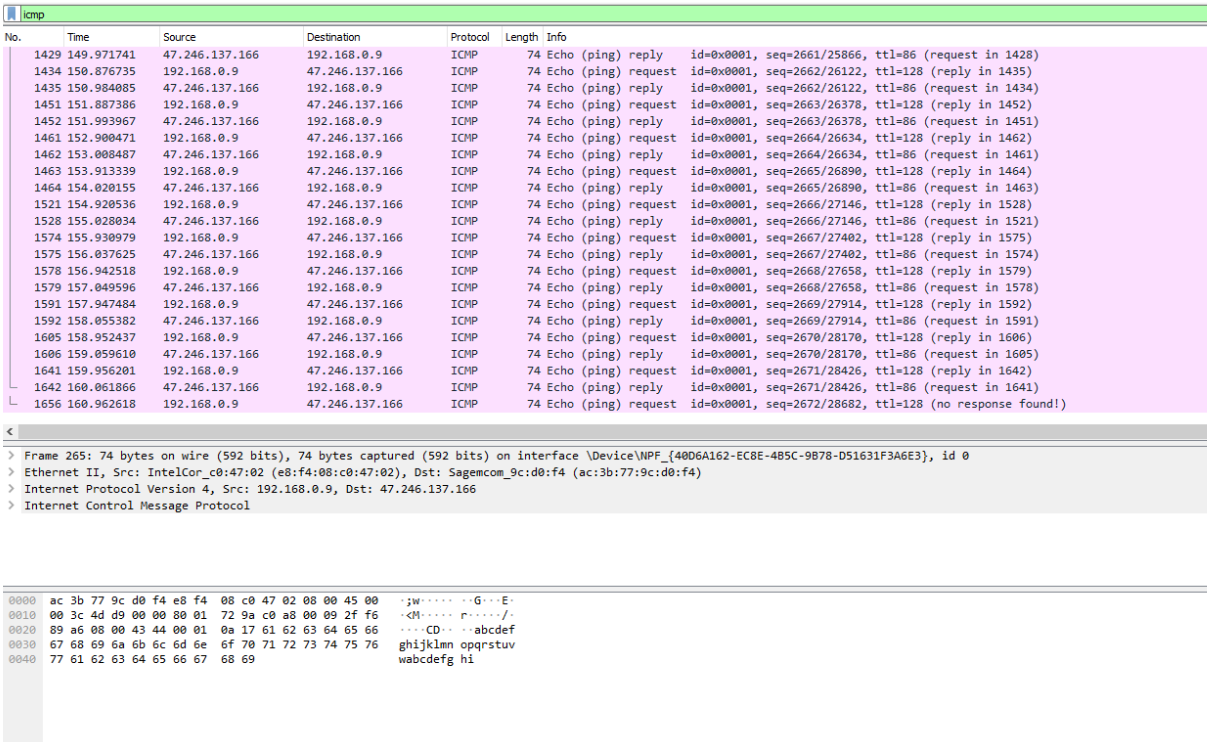
By first pinging Aftenposten.no, we it has a good connection with a short response time of 2ms. Trying Alibaba.com (assuming the server is further away from this computers location). There's also a good connection, but with a significant higher response time. The results can be seen in figure 8.



```
    Network type             : Infrastructure
    Authentication           : WPA2-Personal
    Encryption               : CCMP

SSID 9 : DustinGroup
    Network type             : Infrastructure
    Authentication           : WPA2-Enterprise
    Encryption               : CCMP

SSID 10 : IFP_3443
    Network type             : Infrastructure
    Authentication           : Open
    Encryption               : None

SSID 11 :
    Network type             : Infrastructure
    Authentication           : WPA2-Personal
    Encryption               : CCMP

PS C:\Users\isak.skeie> ping aftenposten.no

Pinging aftenposten.no [195.88.55.37] with 32 bytes of data:
Reply from 195.88.55.37: bytes=32 time=2ms TTL=249
Reply from 195.88.55.37: bytes=32 time=2ms TTL=249
Reply from 195.88.55.37: bytes=32 time=2ms TTL=249
Reply from 195.88.55.37: bytes=32 time=2ms TTL=249

Ping statistics for 195.88.55.37:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 2ms, Average = 2ms
PS C:\Users\isak.skeie> ping alibaba.com

Pinging alibaba.com [47.246.137.166] with 32 bytes of data:
Reply from 47.246.137.166: bytes=32 time=109ms TTL=93
Reply from 47.246.137.166: bytes=32 time=109ms TTL=93
Reply from 47.246.137.166: bytes=32 time=110ms TTL=93
Reply from 47.246.137.166: bytes=32 time=109ms TTL=93

Ping statistics for 47.246.137.166:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 109ms, Maximum = 110ms, Average = 109ms
PS C:\Users\isak.skeie>
```

Figure 8: Ping test applied to two different websites

By using *ping Alibaba.com -t*, the ping command is send continuously until stopped. When opening WireShark and filter for ICMP, its possible to view the ping requests. In the header of each packet, the destination IP-address for Alibaba.com is visible, being 47.246.137.166. When clicking on a packet its possible to view details for that specific package.

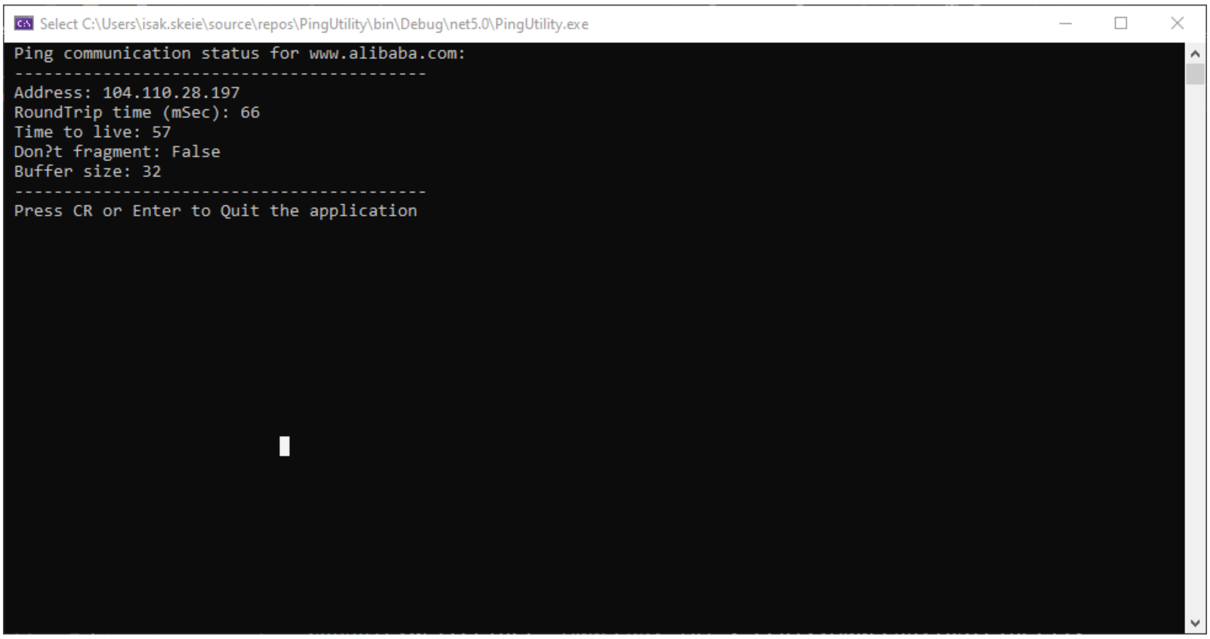


```
icmp
No.   Time       Source          Destination     Protocol Length Info
 1429 149.971741 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2661/25866, ttl=86 (request in 1428)
 1434 150.876735 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2662/26122, ttl=128 (reply in 1435)
 1435 150.984085 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2662/26122, ttl=86 (request in 1434)
 1451 151.887386 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2663/26378, ttl=128 (reply in 1452)
 1452 151.993967 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2663/26378, ttl=86 (request in 1451)
 1461 152.900471 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2664/26634, ttl=128 (reply in 1462)
 1462 153.008487 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2664/26634, ttl=86 (request in 1461)
 1463 153.913339 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2665/26890, ttl=128 (reply in 1464)
 1464 154.020155 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2665/26890, ttl=86 (request in 1463)
 1521 154.920536 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2666/27146, ttl=128 (reply in 1528)
 1528 155.028034 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2666/27146, ttl=86 (request in 1521)
 1574 155.930979 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2667/27402, ttl=128 (reply in 1575)
 1575 156.037625 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2667/27402, ttl=86 (request in 1574)
 1578 156.942518 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2668/27658, ttl=128 (reply in 1579)
 1579 157.049596 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2668/27658, ttl=86 (request in 1578)
 1591 157.947484 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2669/27914, ttl=128 (reply in 1592)
 1592 158.055382 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2669/27914, ttl=86 (request in 1591)
 1605 158.952437 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2670/28170, ttl=128 (reply in 1606)
 1606 159.059610 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2670/28170, ttl=86 (request in 1605)
 1641 159.956201 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2671/28426, ttl=128 (reply in 1642)
 1642 160.061866 47.246.137.166  192.168.0.9     ICMP     74 Echo (ping) reply    id=0x0001, seq=2671/28426, ttl=86 (request in 1641)
 1656 160.962618 192.168.0.9     47.246.137.166  ICMP     74 Echo (ping) request  id=0x0001, seq=2672/28682, ttl=128 (no response found!)

> Frame 265: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{40D6A162-EC8E-4B5C-9B78-D51631F3A6E3}, id 0
> Ethernet II, Src: IntelCor_c0:47:02 (e8:f4:08:c0:47:02), Dst: Sagemcom_9c:d0:f4 (ac:3b:77:9c:d0:f4)
> Internet Protocol Version 4, Src: 192.168.0.9, Dst: 47.246.137.166
> Internet Control Message Protocol

0000  ac 3b 77 9c d0 f4 e8 f4  08 c0 47 02 08 00 45 00   ·;w·····  ··G···E·
0010  00 3c 4d d9 00 00 80 01  72 9a c0 a8 00 09 2f f6   ·<M·····  r·····/·
0020  89 a6 08 00 43 44 00 01  0a 17 61 62 63 64 65 66   ····CD··  ··abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn  opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg  hi
```

Figure 9: Monitoring Ping with WireShark

3.1 Ping Application

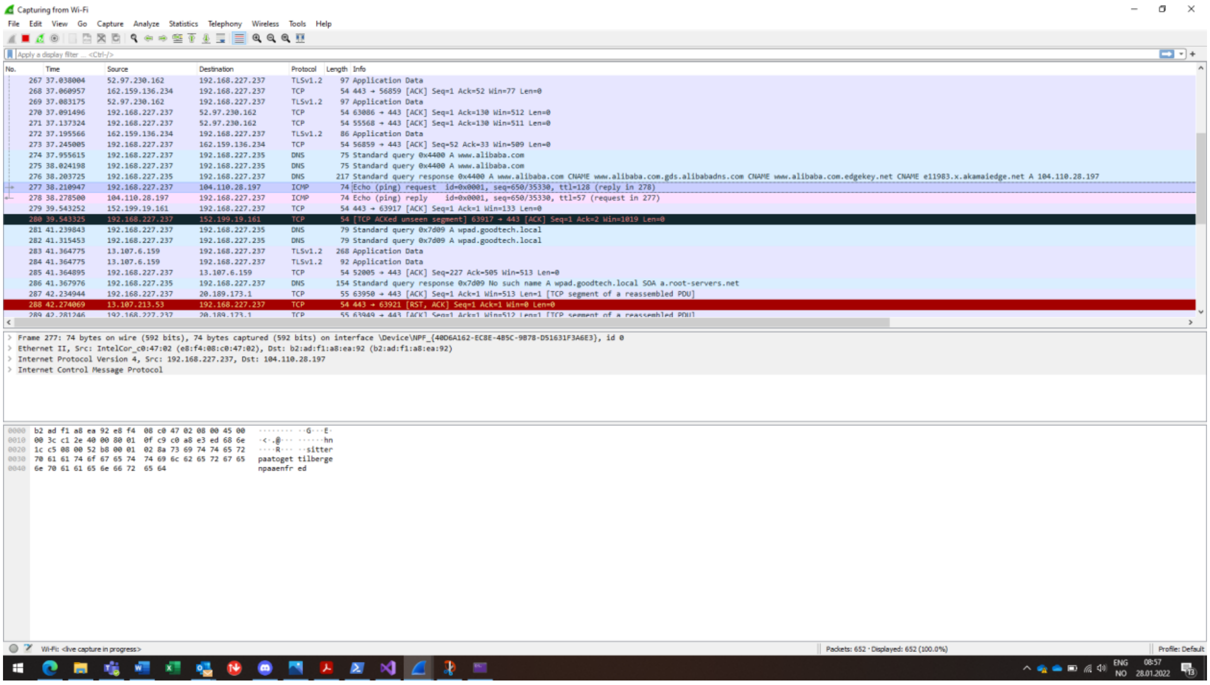
By adding a website and a personal data-string, its possible to apply the ping application and use it for testing, the website used for this program was Alibaba.com, and the personal data-string being ”sitterpaatogettilbergenpaaenfred”. The results gets displayed in a console, as seen in figure 10. By monitoring the network at the same time with WireShark, its possible to pick up the ping package and view its content. As seen in figure 11.



Select C:\Users\isak.skeie\source\repos\PingUtility\bin\Debug\net5.0\PingUtility.exe

```
Ping communication status for www.alibaba.com:
------------------------------------------
Address: 104.110.28.197
RoundTrip time (mSec): 66
Time to live: 57
Don?t fragment: False
Buffer size: 32
------------------------------------------
Press CR or Enter to Quit the application
```

Figure 10: Result from the ping application with a induvidual host, and datapacket



Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time       Source           Destination      Protocol Length Info
 267 37.038004  52.97.230.162    192.168.227.237  TLSv1.2   97 Application Data
 268 37.060957  162.159.136.234  192.168.227.237  TCP       54 443 → 56859 [ACK] Seq=1 Ack=52 Win=77 Len=0
 269 37.083175  52.97.230.162    192.168.227.237  TLSv1.2   97 Application Data
 270 37.091496  192.168.227.237  52.97.230.162    TCP       54 63086 → 443 [ACK] Seq=1 Ack=130 Win=512 Len=0
 271 37.137324  192.168.227.237  52.97.230.162    TCP       54 55568 → 443 [ACK] Seq=1 Ack=130 Win=511 Len=0
 272 37.195566  162.159.136.234  192.168.227.237  TLSv1.2   86 Application Data
 273 37.245005  192.168.227.237  162.159.136.234  TCP       54 56859 → 443 [ACK] Seq=52 Ack=33 Win=509 Len=0
 274 37.955615  192.168.227.237  192.168.227.235  DNS       75 Standard query 0x4400 A www.alibaba.com
 275 38.024198  192.168.227.237  192.168.227.235  DNS       75 Standard query 0x4400 A www.alibaba.com
 276 38.203725  192.168.227.235  192.168.227.237  DNS      217 Standard query response 0x4400 A www.alibaba.com CNAME www.alibaba.com.gds.alibabadns.com CNAME www.alibaba.com.edgekey.net CNAME e11983.x.akamaiedge.net A 104.110.28.197
 277 38.210947  192.168.227.237  104.110.28.197   ICMP      74 Echo (ping) request  id=0x0001, seq=650/35330, ttl=128 (reply in 278)
 278 38.278500  104.110.28.197   192.168.227.237  ICMP      74 Echo (ping) reply    id=0x0001, seq=650/35330, ttl=57 (request in 277)
 279 39.543252  152.199.19.161   192.168.227.237  TCP       54 443 → 63917 [ACK] Seq=1 Ack=1 Win=133 Len=0
 280 39.543325  192.168.227.237  152.199.19.161   TCP       54 [TCP ACKed unseen segment] 63917 → 443 [ACK] Seq=1 Ack=2 Win=1019 Len=0
 281 41.239843  192.168.227.237  192.168.227.235  DNS       79 Standard query 0x7d09 A wpad.goodtech.local
 282 41.315453  192.168.227.237  192.168.227.235  DNS       79 Standard query 0x7d09 A wpad.goodtech.local
 283 41.364775  13.107.6.159     192.168.227.237  TLSv1.2  268 Application Data
 284 41.364775  13.107.6.159     192.168.227.237  TLSv1.2   92 Application Data
 285 41.364895  192.168.227.237  13.107.6.159     TCP       54 52005 → 443 [ACK] Seq=227 Ack=505 Win=513 Len=0
 286 41.367976  192.168.227.235  192.168.227.237  DNS      154 Standard query response 0x7d09 No such name A wpad.goodtech.local SOA a.root-servers.net
 287 42.234944  192.168.227.237  20.189.173.1     TCP       55 63950 → 443 [ACK] Seq=1 Ack=1 Win=513 Len=1 [TCP segment of a reassembled PDU]
 288 42.274069  13.107.213.53    192.168.227.237  TCP       54 443 → 63921 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 289 42.281246  192.168.227.237  20.189.173.1     TCP       55 63949 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1 [TCP segment of a reassembled PDU]
```

```
> Frame 277: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{40D6A162-EC8E-4B5C-9B78-D51631F3A6E3}, id 0
> Ethernet II, Src: IntelCor_c0:47:02 (e8:f4:08:c0:47:02), Dst: b2:ad:f1:a8:ea:92 (b2:ad:f1:a8:ea:92)
> Internet Protocol Version 4, Src: 192.168.227.237, Dst: 104.110.28.197
> Internet Control Message Protocol
```

```
0000  b2 ad f1 a8 ea 92 e8 f4  08 c0 47 02 08 00 45 00   ········ ··G···E·
0010  00 3c c1 2e 40 00 80 01  0f c9 c0 a8 e3 ed 68 6e   ·<·.@··· ······hn
0020  1c c5 08 00 52 b8 00 01  02 8a 73 69 74 74 65 72   ····R··· ··sitter
0030  70 61 61 74 6f 67 65 74  74 69 6c 62 65 72 67 65   paatoget tilberge
0040  6e 70 61 61 65 6e 66 72  65 64                     npaaenfr ed
```

Wi-Fi: <live capture in progress>
Packets: 652 · Displayed: 652 (100.0%)
Profile: Default
ENG
NO
08:57
28.01.2022

Figure 11: Packet information from ping viewed in WireShark

3.2 Pseudocode for average ping response

The pseudocode below shows how you could calculate an average of several responses. This is done using the source code for the Ping Test Utility as a class.

```
using System;

namespace PingAverageResponse
{
    class Program
    {
        static void Main(string[] args)
        {

            Initialize host-string;
            Initialize data-string;
            Initialize sum-int;
            Initialize average-double;
            Initialize responses-int;
            Declares array of classes with PingPackage;

            for( Iterates through the number of responses)
            {
               Initializes classes for pingRespons;
                Adds the response from class PingResponse to sum;

            }
             Calculates average: sum divided by (number of responses minus packages
not received);


            Write average to console;

        }
    }

}
```

3.3 Pseudocode for checking all nodes in network

As with the pseudocode for the Average ping response, the source code for Ping Response utility code is used to check whether it’s a valid IP-address or not. The source code for this console application can be found under Appendix B.

```
using System;
using System.Net.NetworkInformation;
using System.Net;
using System.Collections.Generic;
using PingUtility;

namespace NodeOverView
{

    public class Program
    {

        static void Main(string[] args)
        {
            Initilize data-string;
            Initialize ip-address string;
            Initialized empty ip search string;
            Create bool for ping test result;
            Create Lost of strings for storing the available nodes;

            Create new instance of ping testing class;

            Write header for console, declaring the found nodes;

            for(Goes through all the available network address within the subnet)
            {
                Creates ip from ip-adress and iterative number;
                Send Data and ip to ping test instance;

                if(Ping Test is succesful)
                {
                    Add ip to List;
                    Write ip to console;
                }
            }
    }
}
```

4 Conclusion

The tasks performed throughout this assignment have highlighted the communication capabilities of a regular laptop, as well as going into the fundamentals of networks and aspects of the OSI-model. By gathering network information and testing networks, you get insight into the different aspects of layer 2, and 3 of the OSI-Model. This is knowledge that's fundamental for an automation engineer to grasp. The importance and use of this knowledge is shown in the tasks with the ping applications. Exploring relevant custom tools an engineer might need, as well as the occurrence of vulnerabilities in communication systems.

Network tools like the pingAppConsole, WireShark, ping and ipconfig are valuable and crucial for an engineer. For someone to be able to use these tools, its important to understand the fundamental aspect of communication and networking, as well networking theory surrounding them.

5 References

[1] Cisco, «Understanding the Ping and Traceroute commands,» 03 02 2022. [Internett]. Available: https://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-software-releases-121-mainline/12778-ping-traceroute.html.

[2] A. Zola, «Ping,» 3 2 2022. [Internett]. Available: https://www.techtarget.com/searchnetworking/definition/ping.

[3] w3schools, «Types of network protocols and their use,» 2 2 2022. [Internett]. Available: https://www.w3schools.in/types-of-network-protocols-and-their-uses/.

[4] P. Predamkar, «Types of networking protocols,» 2 2 2022. [Internett]. Available: https://www.educba.com/types-of-networking-protocols/.

[5] pandasecurity, «wpa-vs-wpa2,» pandasecurity, 20 1 2022. [Internett]. Available: https://www.pandasecurity.com/en/mediacenter/security/wpa-vs-wpa2/. [Funnet 3 2 2022].

Appendices

Appendix A <Ping source code based on
Microsoft MSDN1 information.>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Threading;
using System.Net;
using System.Net.NetworkInformation;
//
namespace PingUtility
{
    class Program
    {
        /// <summary>
        ///
///////////////////////////////////////////////////////////////////////////////
        static void Main(string[] args)
        ///
        /// Purpose: the main function in the application handling the
ping()communication
        ///
        /// Version: 1.0: 8-JAN-17: NOS
        /// </summary>
        {
            string host, data;
            byte[] buffer;
            int timeout;
            Ping pingSender = new Ping();
            PingOptions options = new PingOptions();


            // Use the default Ttl value which is 128,
            // but change the fragmentation behavior.
            options.DontFragment = true;
            // Create a buffer of 32 bytes of data to be transmitted.
            data = "sitterpaatogettilbergenpaaenfred";
            buffer = Encoding.ASCII.GetBytes(data);
            timeout = 120;
            // Name or address of node to access
            host = "www.alibaba.com";
            PingReply reply = pingSender.Send(host, timeout, buffer, options);
            if (reply.Status == IPStatus.Success)
            {
                Console.WriteLine(" Ping communication status for {0}:", host);
                Console.WriteLine(" ------------------------------------------");
                Console.WriteLine(" Address: {0}", reply.Address.ToString());
                Console.WriteLine(" RoundTrip time (mSec): {0}",
reply.RoundtripTime);
                Console.WriteLine(" Time to live: {0}", reply.Options.Ttl);
                Console.WriteLine(" Don□t fragment: {0}",
reply.Options.DontFragment);
                Console.WriteLine(" Buffer size: {0}", reply.Buffer.Length);
                Console.WriteLine(" ------------------------------------------");
            }
            else
            {
                Console.WriteLine(" Error connecting to network address/name {0}",
host);
            }
            Console.WriteLine(" Press CR or Enter to Quit the application");
            Console.ReadLine();
        }
    }
```

```
}
```

Appendix B <Code for Node finder >

```
using System;
using System.Net.NetworkInformation;
using System.Net;
using System.Collections.Generic;
using PingUtility;

namespace NodeOverView
{

    public class Program
    {


        static void Main(string[] args)
        {

            /* GetIp used to recieve ip-adress of the wifi-interface
            GetIp();
            */
            string data = "sitterpaatogettilbergenpaaenfred";
            string ip = "10.0.0.";
            string ipSearch = "";
            bool PingTest;
            List<string> Nodes = new List<string>();

            PingPackage ping = new PingPackage();

            Console.WriteLine("The nodes at the local network are:");
            for(int i = 0; i<254; i++)
            {
                ipSearch = ip + i.ToString();
                PingTest = ping.Ping(data, ipSearch);

                if(PingTest)
                {
                    Nodes.Add(ipSearch);
                    Console.WriteLine(ipSearch);
                }
            }
        }

        static public string GetIp()
        {
            string hostName = Dns.GetHostName(); // Retrive the Name of HOST
            string myIP = Dns.GetHostByName(hostName).AddressList[6].ToString();

            Console.WriteLine("My IP Address is :" + myIP);

            Console.ReadKey();

            return myIP;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Threading;
using System.Net;
using System.Net.NetworkInformation;


namespace PingUtility
{
    public class PingPackage
    {
        byte[] buffer;
        int timeout;
        private string data;
        private string host;
        public int response;
        public static int conErrors = 0;

        Ping pingSender = new Ping();
        PingOptions options = new PingOptions();

        //options.DontFragment = true;

        public bool Ping(string dataSet, string hostSet)
        {
            data = dataSet;
            host = hostSet;


            buffer = Encoding.ASCII.GetBytes(data);
            timeout = 120;
            PingReply reply = pingSender.Send(host, timeout, buffer, options);
            if (reply.Status == IPStatus.Success)
            {


                return true;
            }
            else
            {
                return false;

            }
        }
    }
}
```