Data Communication

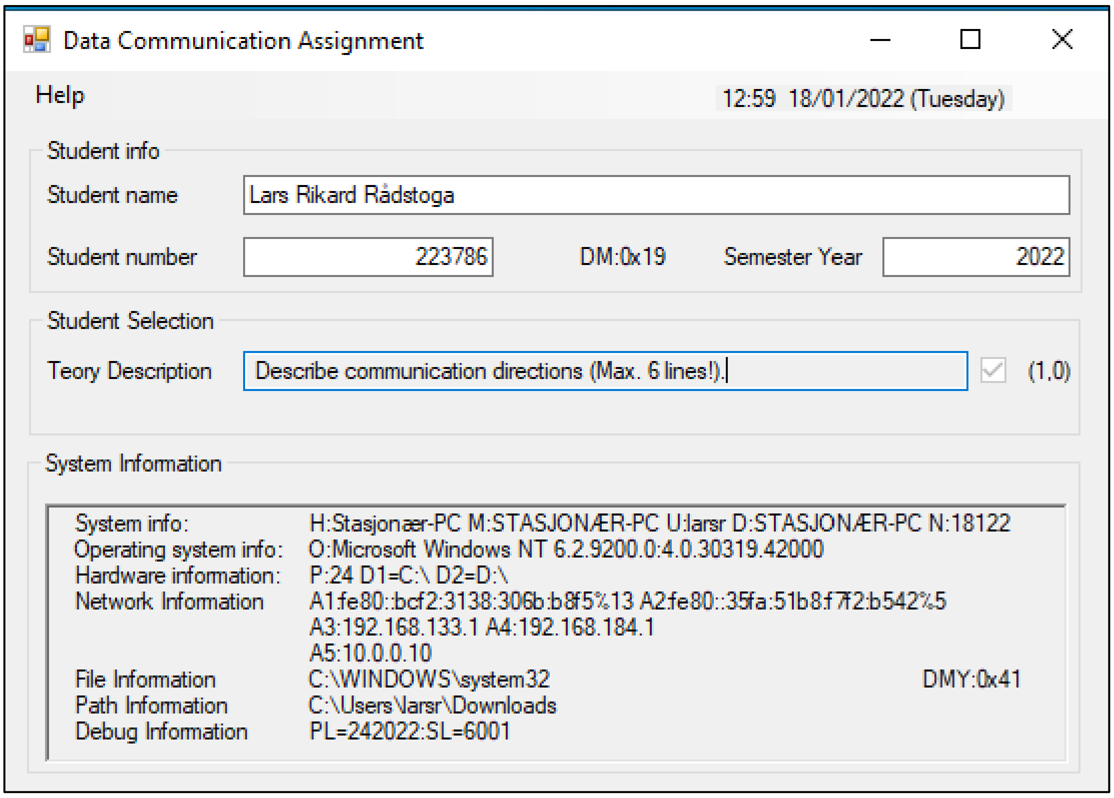
Lars Rikard Rådstoga | 223786

2022-01-18

1 Introduction

This document contains answers to the assignment “Data Communication” given as a compulsory assignment in the course “Industrial Information Technology” at the University of South-Eastern Norway – spring 2022.

Screenshot in Figure 1-1: Figure 1-1 shows a seemingly deterministic program which assigned the task “Describe communication directions”. The program scanned my computer for information and took my student information as input to generate the task. Repeated tests of the program with the same inputs resulted in the same task. Thus, the program is inductively postulated to be deterministic.



Data Communication Assignment

Help — 12:59 18/01/2022 (Tuesday)

Student info

Student name	Lars Rikard Rådstoga		
Student number	223786	DM:0x19	Semester Year 2022

Student Selection

Teory Description: Describe communication directions (Max. 6 lines!). ✓ (1,0)

System Information

```
System info:             H:Stasjonær-PC M:STASJONÆR-PC U:larsr D:STASJONÆR-PC N:18122
Operating system info:   O:Microsoft Windows NT 6.2.9200.0:4.0.30319.42000
Hardware information:    P:24 D1=C:\ D2=D:\
Network Information      A1:fe80::bcf2:3138:306b:b8f5%13 A2:fe80::35fa:51b8:f7f2:b542%5
                         A3:192.168.133.1 A4:192.168.184.1
                         A5:10.0.0.10
File Information         C:\WINDOWS\system32                              DMY:0x41
Path Information         C:\Users\larsr\Downloads
Debug Information        PL=242022:SL=6001
```

Figure 1-1: Student and computer information

2 System information

This chapter contains answers to tasks in chapter 1.3 from the assignment document.

2.1 Interface protocols between motherboard and storage devices

My computer has two disk drives as seen in Figure 2-1. They are sized roughly 1TB and 2TB as seen in Figure 2-2 and Figure 2-3. The disks are of types: SSD and SSHD (some sort of hybrid drive) respectively. The SSD is connected using the NVMe protocol as seen in Figure 2-4 and the SSHD uses the SATA protocol seen in Figure 2-5.

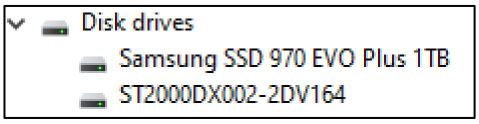
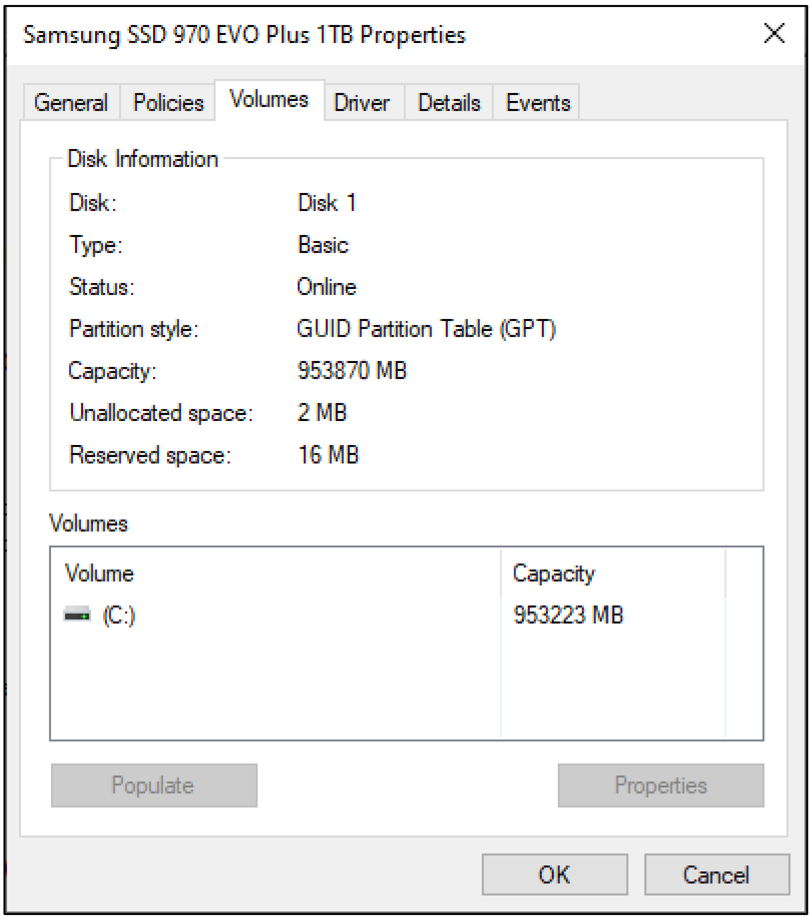

Disk drives
Samsung SSD 970 EVO Plus 1TB
ST2000DX002-2DV164

Figure 2-1 Disk drives on my computer.


Samsung SSD 970 EVO Plus 1TB Properties

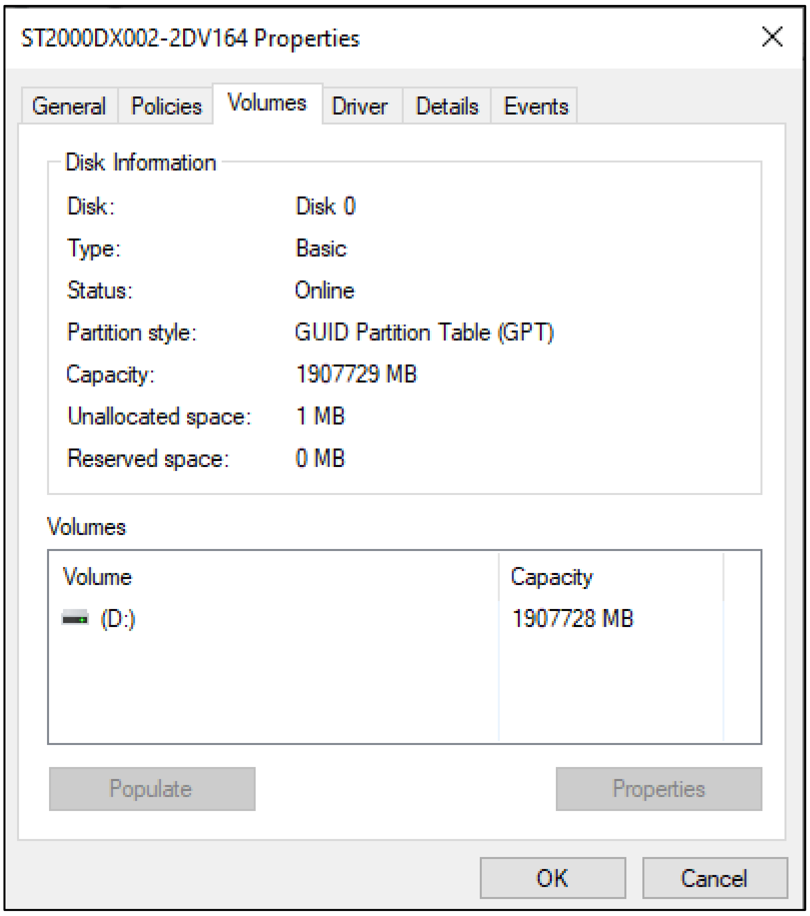
Disk Information

Disk:	Disk 1
Type:	Basic
Status:	Online
Partition style:	GUID Partition Table (GPT)
Capacity:	953870 MB
Unallocated space:	2 MB
Reserved space:	16 MB

Volumes

Volume	Capacity
(C:)	953223 MB

Figure 2-2 Details of my M.2 SSD.


ST2000DX002-2DV164 Properties

General | Policies | Volumes | Driver | Details | Events

Disk Information

Disk:	Disk 0
Type:	Basic
Status:	Online
Partition style:	GUID Partition Table (GPT)
Capacity:	1907729 MB
Unallocated space:	1 MB
Reserved space:	0 MB

Volumes

Volume	Capacity
(D:)	1907728 MB

Figure 2-3 Details of my HDD.

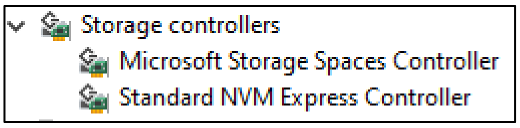

Storage controllers
- Microsoft Storage Spaces Controller
- Standard NVM Express Controller

Figure 2-4 Storage controllers

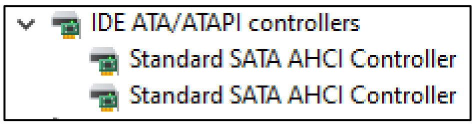
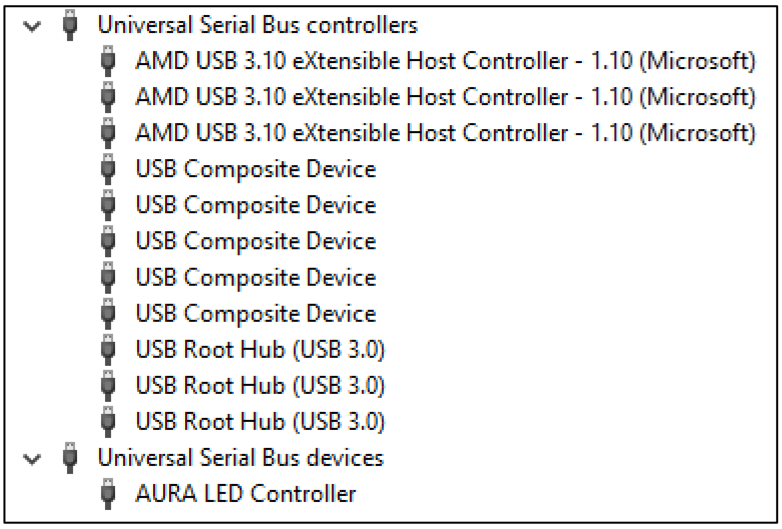

IDE ATA/ATAPI controllers
- Standard SATA AHCI Controller
- Standard SATA AHCI Controller

Figure 2-5 Standard interface for SATA communication

2.2 Serial ports

Figure 2-6 should reflect the amount of USB ports on the computer, but the list contains 11 ports, but the computer case only has 9 ports connected to the motherboard. Perhaps this means it could support more ports if the case supported it.

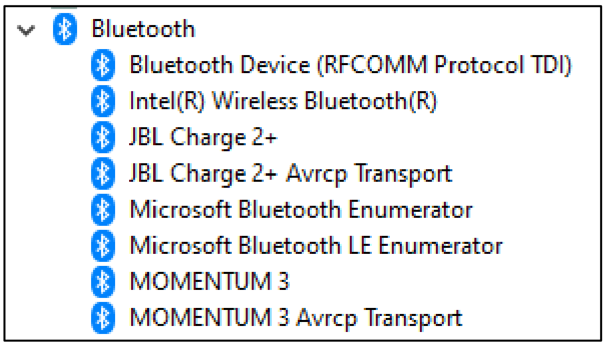


Universal Serial Bus controllers
AMD USB 3.10 eXtensible Host Controller - 1.10 (Microsoft)
AMD USB 3.10 eXtensible Host Controller - 1.10 (Microsoft)
AMD USB 3.10 eXtensible Host Controller - 1.10 (Microsoft)
USB Composite Device
USB Composite Device
USB Composite Device
USB Composite Device
USB Composite Device
USB Root Hub (USB 3.0)
USB Root Hub (USB 3.0)
USB Root Hub (USB 3.0)
Universal Serial Bus devices
AURA LED Controller

Figure 2-6 An array of USB drivers.

2.3 Bluetooth devices

Figure 2-7 shows the installed Bluetooth profiles [1] . Interestingly the connected audio devices have separate profiles for the audio streaming and the remote-control features of the device. E.g., the Momentum 3 headset uses the AVCRP profile to pause, skip and adjust volume, etc.



Bluetooth
Bluetooth Device (RFCOMM Protocol TDI)
Intel(R) Wireless Bluetooth(R)
JBL Charge 2+
JBL Charge 2+ Avrcp Transport
Microsoft Bluetooth Enumerator
Microsoft Bluetooth LE Enumerator
MOMENTUM 3
MOMENTUM 3 Avrcp Transport

Figure 2-7 Bluetooth drivers.

3 Network information

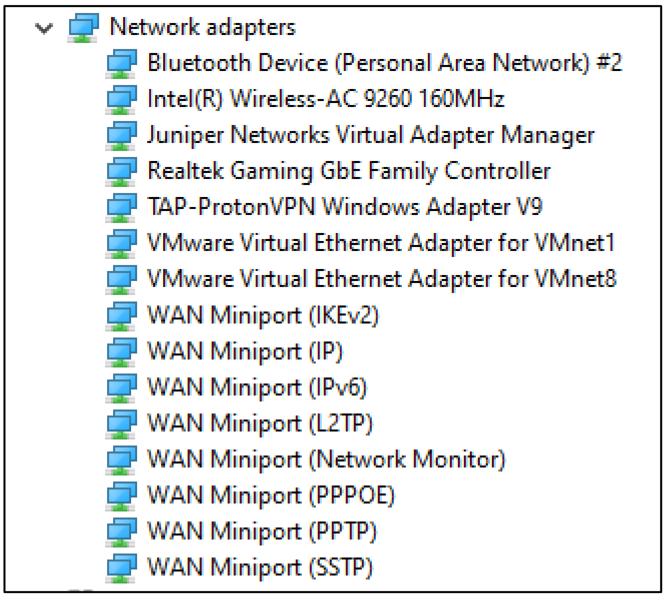
This chapter contains answers to tasks in chapter 1.4 from the assignment document.

3.1 Describe communication directions

In a physical cable, communication can only occur in one direction at a time. A setup with communication set to always be in the same direction is called simplex. A setup which requires two-way communication will either require half-duplex or duplex. Duplex communication requires two cables or channels. Half-duplex allows for two-way communication with only one cable, using some sort of Primary/Replica protocol to avoid signals from interfering with each other.

3.2 Number of network devices

Figure 3-1 shows network drivers of the computer. These do not reflect the number of network devices.

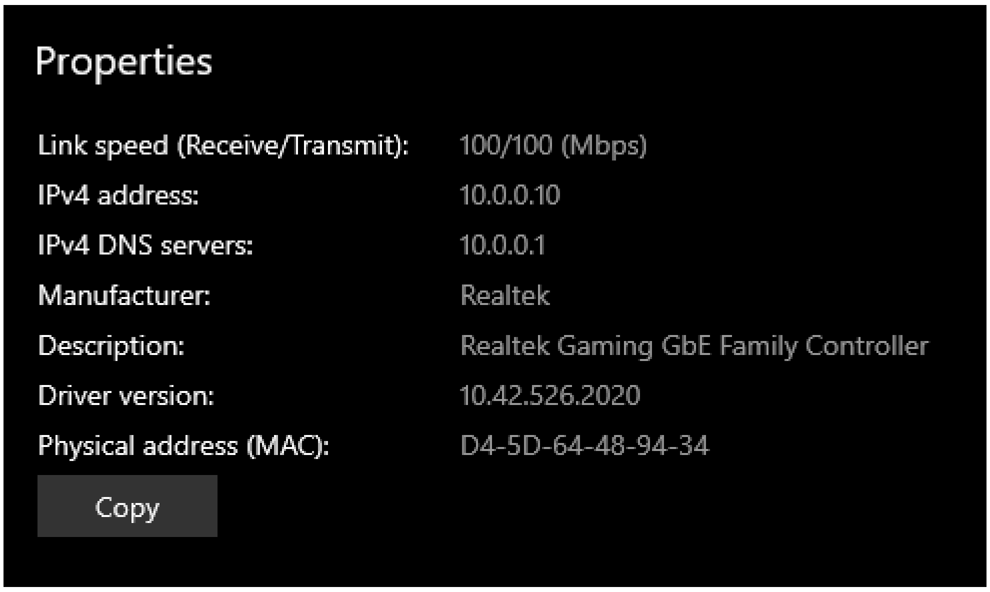


```
Network adapters
    Bluetooth Device (Personal Area Network) #2
    Intel(R) Wireless-AC 9260 160MHz
    Juniper Networks Virtual Adapter Manager
    Realtek Gaming GbE Family Controller
    TAP-ProtonVPN Windows Adapter V9
    VMware Virtual Ethernet Adapter for VMnet1
    VMware Virtual Ethernet 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)
```

Figure 3-1 Network drivers.

3.3 Physical address

Figure 3-2 shows that the physical address (MAC) of the network adapter is D4-5D-64-48-94-34.



Properties

Link speed (Receive/Transmit):	100/100 (Mbps)
IPv4 address:	10.0.0.10
IPv4 DNS servers:	10.0.0.1
Manufacturer:	Realtek
Description:	Realtek Gaming GbE Family Controller
Driver version:	10.42.526.2020
Physical address (MAC):	D4-5D-64-48-94-34

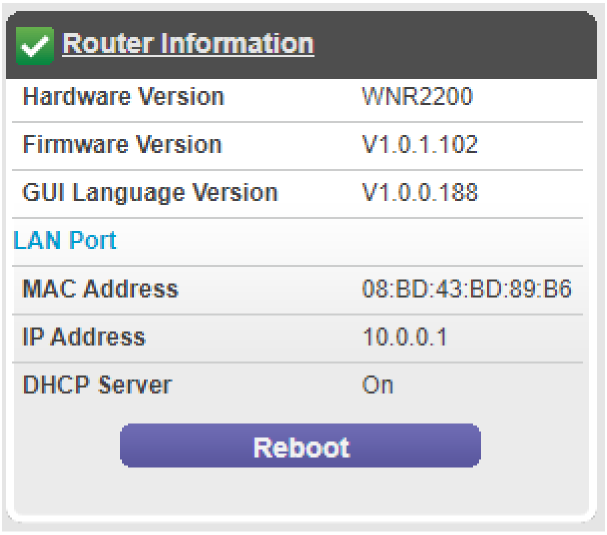
Figure 3-2 Network details

3.4 Logical address

Figure 3-2 shows that the logical address of the computer on the network is 10.0.0.10. The logical address of the network is 10.0.0.0 [2].

3.5 Why TCP/IP

It is likely to be a TCP/IP address because it is characteristic of a class A network class. 10.0.0.0 is the network address and 0.0.0.10 is the host address. The network class is of type A because the first octet of the network address is between 0 and 127 [2]. This network class uses the default subnet mask of 255.0.0.0. It is a dynamic address because the DHCP server is turned in the router settings shown in Figure 3-3.



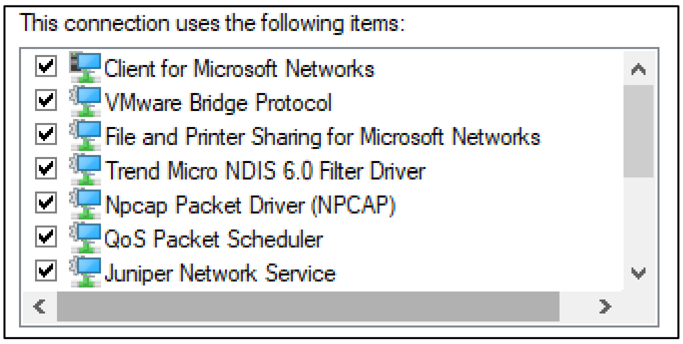
Router Information

Hardware Version	WNR2200
Firmware Version	V1.0.1.102
GUI Language Version	V1.0.0.188
LAN Port	
MAC Address	08:BD:43:BD:89:B6
IP Address	10.0.0.1
DHCP Server	On

Figure 3-3 Router settings

3.6 Other network protocols installed

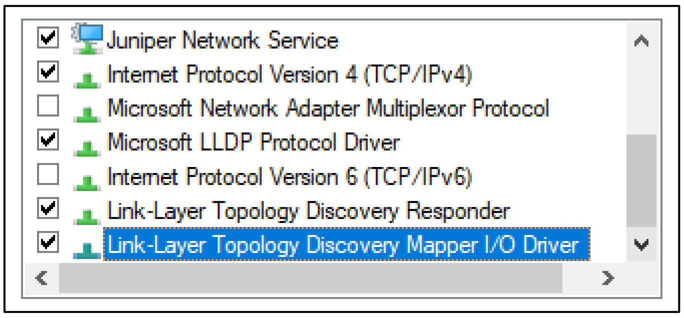
Figure 3-4 and Figure 3-5 show other installed network protocols.



This connection uses the following items:

- ✓ Client for Microsoft Networks
- ✓ VMware Bridge Protocol
- ✓ File and Printer Sharing for Microsoft Networks
- ✓ Trend Micro NDIS 6.0 Filter Driver
- ✓ Npcap Packet Driver (NPCAP)
- ✓ QoS Packet Scheduler
- ✓ Juniper Network Service

Figure 3-4 Installed network protocols (1/2)



☑ Juniper Network Service
☑ Internet Protocol Version 4 (TCP/IPv4)
☐ Microsoft Network Adapter Multiplexor Protocol
☑ Microsoft LLDP Protocol Driver
☐ Internet Protocol Version 6 (TCP/IPv6)
☑ Link-Layer Topology Discovery Responder
☑ Link-Layer Topology Discovery Mapper I/O Driver

Figure 3-5 Installed network protocols (2/2)

3.7 IPv6 support?

Yes, seemingly IPv6 is supported, but not activated/used. see Figure 3-5.

3.8 Wireless

This chapter contains answers to tasks in chapter 1.4.1 from the assignment document.

3.8.1 Which network will you use?

The network that is first from the top is the one I would connect to as it has the best signal strength.

3.8.2 Does it require a password?

It does not require a password as it is unencrypted.

3.8.3 Why do we not see any other networks?

Wireless networks do not have infinite practical reach, this can be one reason we don't see other networks. But it is also possible host networks using hidden SSIDs, these won't appear in the network list either.

3.8.4 Assumptions about network owners

I assume that the computer that is scanning for the networks are in the same building as the "CM" network and the AB networks belong to a neighboring office or similar. I assume that the encryptions are weak (WEP) because I would honestly expect WPA2 in the windows XP era, but one of the networks explicitly states it uses WPA so I assume WPA2 would also show up in the interface if it was the case.

3.8.5 My available networks

Networks available from my location shown in Figure 3-6.

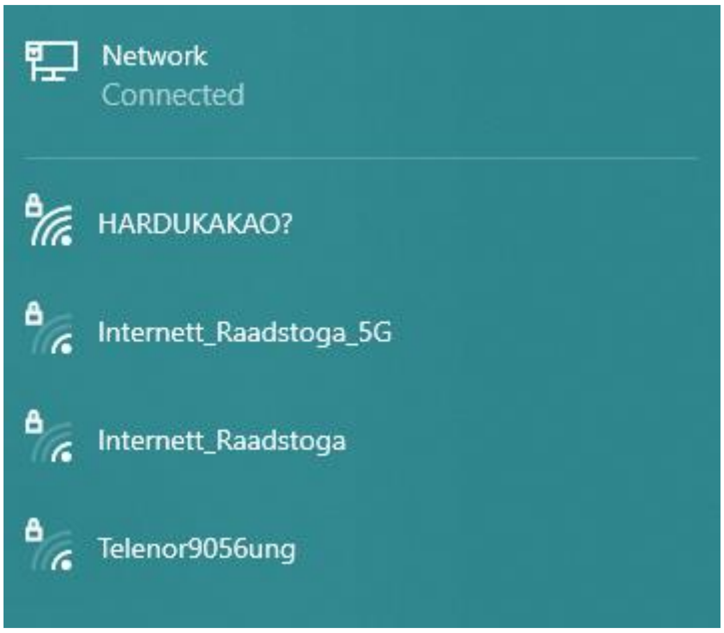

Network
Connected
HARDUKAKAO?
Internett_Raadstoga_5G
Internett_Raadstoga
Telenor9056ung

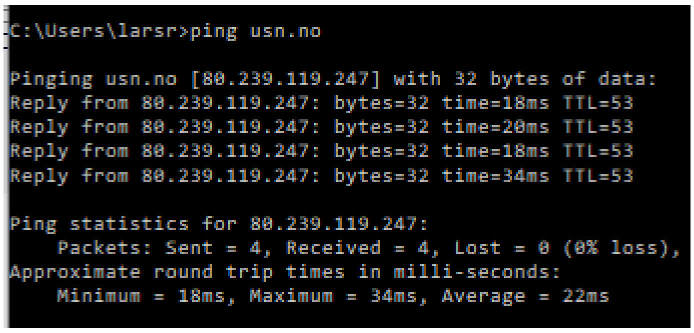
Figure 3-6 My network list.

4 Network testing

This chapter contains answers to tasks in chapter 1.5 from the assignment document.

4.1 Pinging usn.no

We can see in Figure 4-1 that the average ping response time to usn.no is 22ms.

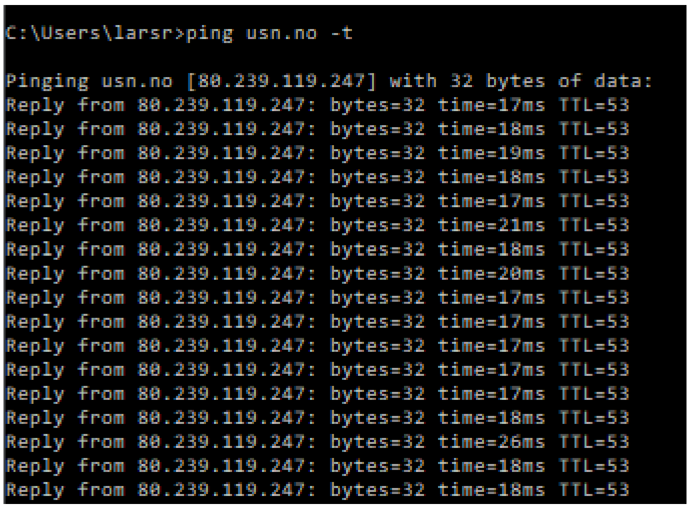


```
C:\Users\larsr>ping usn.no

Pinging usn.no [80.239.119.247] with 32 bytes of data:
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=20ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=34ms TTL=53

Ping statistics for 80.239.119.247:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 18ms, Maximum = 34ms, Average = 22ms
```

Figure 4-1 A single ping with cmd.

4.2 Continuous pinging

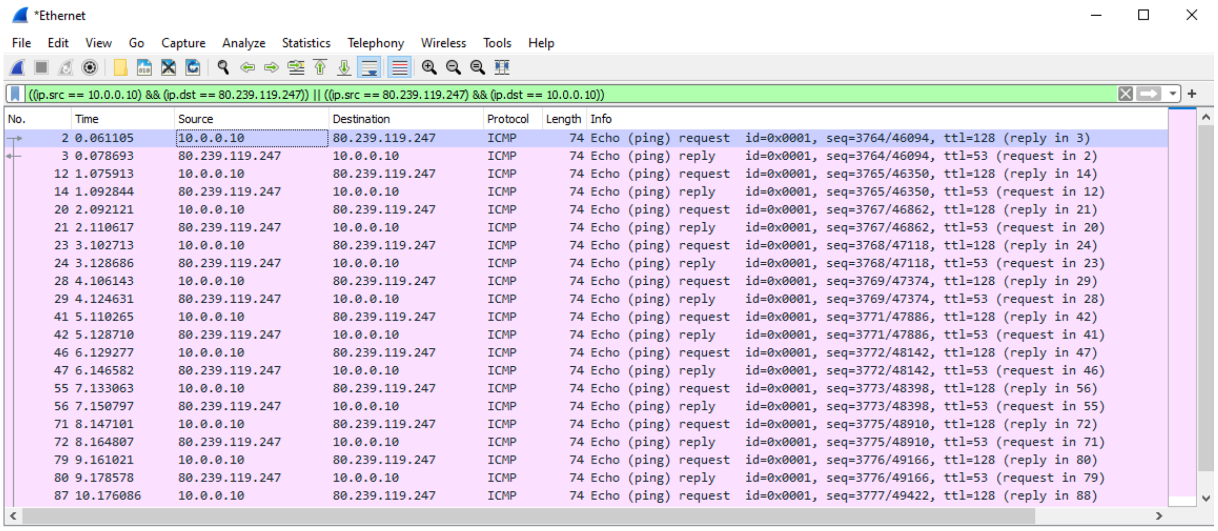
Continuous pinging seen in Figure 4-2.



```
C:\Users\larsr>ping usn.no -t

Pinging usn.no [80.239.119.247] with 32 bytes of data:
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=19ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=21ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=20ms TTL=53
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=17ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=26ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
Reply from 80.239.119.247: bytes=32 time=18ms TTL=53
```

Figure 4-2 Continuous pinging with cmd.

The TCP/IP address of the remote node is 80.239.119.247 as seen in



*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

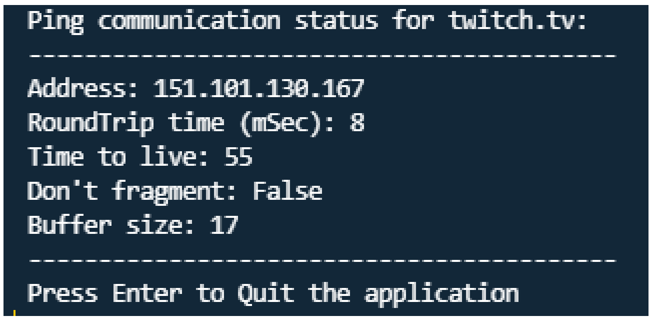
```
((ip.src == 10.0.0.10) && (ip.dst == 80.239.119.247)) || ((ip.src == 80.239.119.247) && (ip.dst == 10.0.0.10))
```

```
No.  Time      Source          Destination     Protocol Length Info
   2 0.061105  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3764/46094, ttl=128 (reply in 3)
   3 0.078693  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3764/46094, ttl=53 (request in 2)
  12 1.075913  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3765/46350, ttl=128 (reply in 14)
  14 1.092844  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3765/46350, ttl=53 (request in 12)
  20 2.092121  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3767/46862, ttl=128 (reply in 21)
  21 2.110617  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3767/46862, ttl=53 (request in 20)
  23 3.102713  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3768/47118, ttl=128 (reply in 24)
  24 3.128686  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3768/47118, ttl=53 (request in 23)
  28 4.106143  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3769/47374, ttl=128 (reply in 29)
  29 4.124631  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3769/47374, ttl=53 (request in 28)
  41 5.110265  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3771/47886, ttl=128 (reply in 42)
  42 5.128710  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3771/47886, ttl=53 (request in 41)
  46 6.129277  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3772/48142, ttl=128 (reply in 47)
  47 6.146582  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3772/48142, ttl=53 (request in 46)
  55 7.133063  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3773/48398, ttl=128 (reply in 56)
  56 7.150797  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3773/48398, ttl=53 (request in 55)
  71 8.147101  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3775/48910, ttl=128 (reply in 72)
  72 8.164807  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3775/48910, ttl=53 (request in 71)
  79 9.161021  10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3776/49166, ttl=128 (reply in 80)
  80 9.178578  80.239.119.247  10.0.0.10       ICMP     74 Echo (ping) reply    id=0x0001, seq=3776/49166, ttl=53 (request in 79)
  87 10.176086 10.0.0.10       80.239.119.247  ICMP     74 Echo (ping) request  id=0x0001, seq=3777/49422, ttl=128 (reply in 88)
```

Figure 4-3 Wireshark: filtered to see the ping messages.

4.3 Testing the ping application

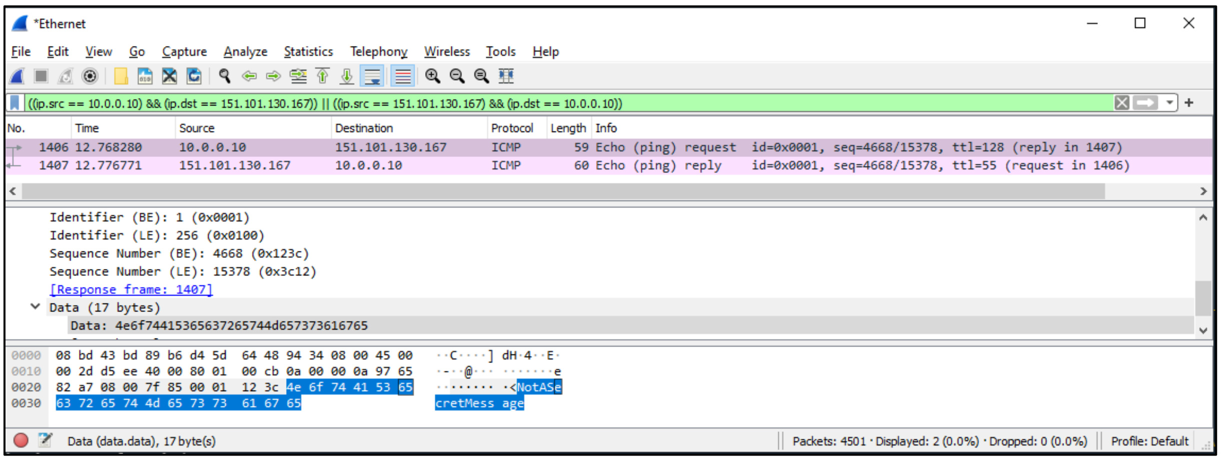
Test shows that the ping from my pc to twitch.tv is 8ms as seen in Figure 4-4. See Appendix A: Testing ping to a connected node for the source code.



```
Ping communication status for twitch.tv:
-------------------------------------------
Address: 151.101.130.167
RoundTrip time (mSec): 8
Time to live: 55
Don't fragment: False
Buffer size: 17
-------------------------------------------
Press Enter to Quit the application
```

Figure 4-4 Ping response for twitch.tv.

It is clearly possible to sniff the message sent with the ping as shown in Figure 4-5.



*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
((ip.src == 10.0.0.10) && (ip.dst == 151.101.130.167)) || ((ip.src == 151.101.130.167) && (ip.dst == 10.0.0.10))

No.  Time       Source           Destination      Protocol Length Info
1406 12.768280  10.0.0.10        151.101.130.167  ICMP     59 Echo (ping) request  id=0x0001, seq=4668/15378, ttl=128 (reply in 1407)
1407 12.776771  151.101.130.167  10.0.0.10        ICMP     60 Echo (ping) reply    id=0x0001, seq=4668/15378, ttl=55 (request in 1406)

    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence Number (BE): 4668 (0x123c)
    Sequence Number (LE): 15378 (0x3c12)
    [Response frame: 1407]
  Data (17 bytes)
      Data: 4e6f744153656372657444d657373616765

0000  08 bd 43 bd 89 b6 d4 5d  64 48 94 34 08 00 45 00   ··C····] dH·4··E·
0010  00 2d d5 ee 40 00 80 01  00 cb 0a 00 00 0a 97 65   ·-··@··· ·······e
0020  82 a7 08 00 7f 85 00 01  12 3c 4e 6f 74 41 53 65   ········ ·<NotASe
0030  63 72 65 74 4d 65 73 73  61 67 65                  cretMess age

Data (data.data), 17 byte(s)    Packets: 4501 · Displayed: 2 (0.0%) · Dropped: 0 (0.0%)    Profile: Default
```

Figure 4-5 Sniffing the message in the ping with WireShark.

4.4 Pseudo code continuous ping application

A ping application that takes 5 recent ping values and averages them:

```
Recent pings = Array of size 5
int i = 0;
while (true){
        Recent pings[i%recent_pings.length] = ping(website);
        Print(average(recent pings));
        i++;
        sleep(a bit);
}
```

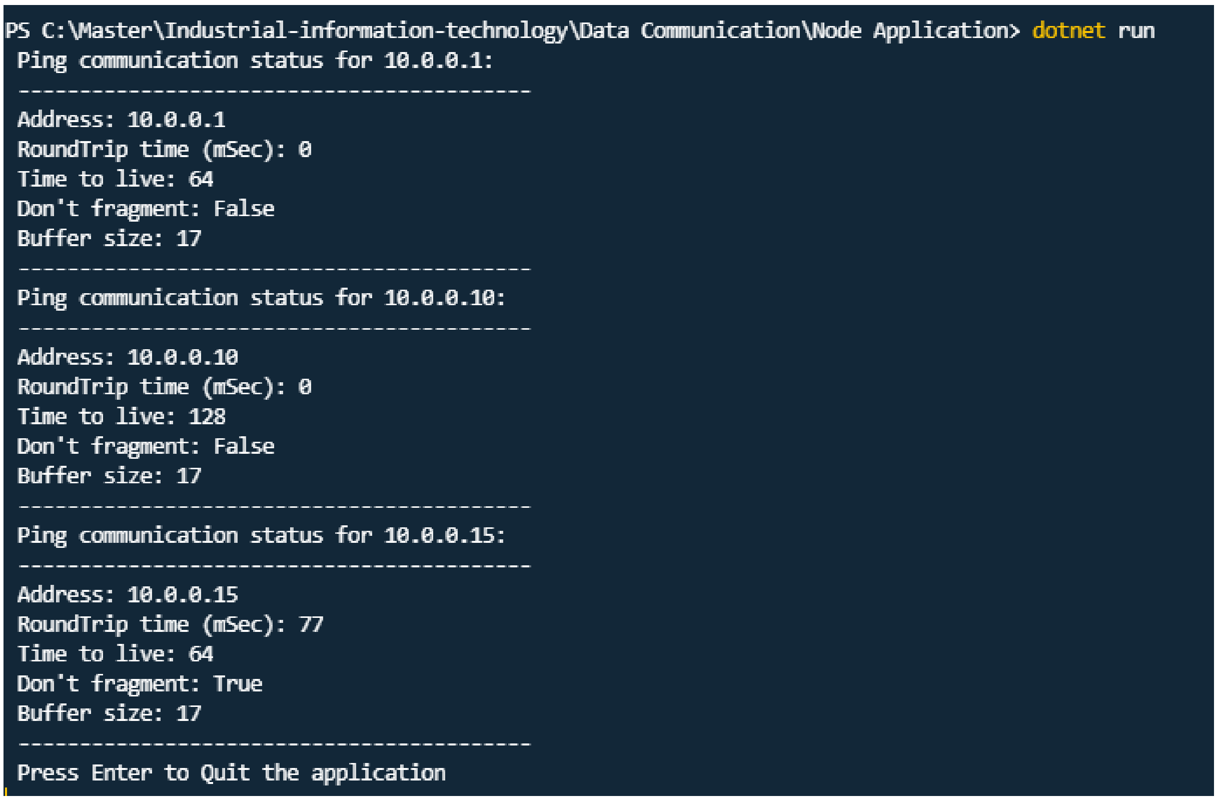
4.5 Pseudo code of node checking ping application

A program that iterates local network nodes, stores and prints their response times:

```
Nodes = array of size 256
for(i = 0; i < nodes.length, i++){
        pingObject = ping(10.0.0.i)
        nodes[i] = pingObject
        print(pingObject)
}
```

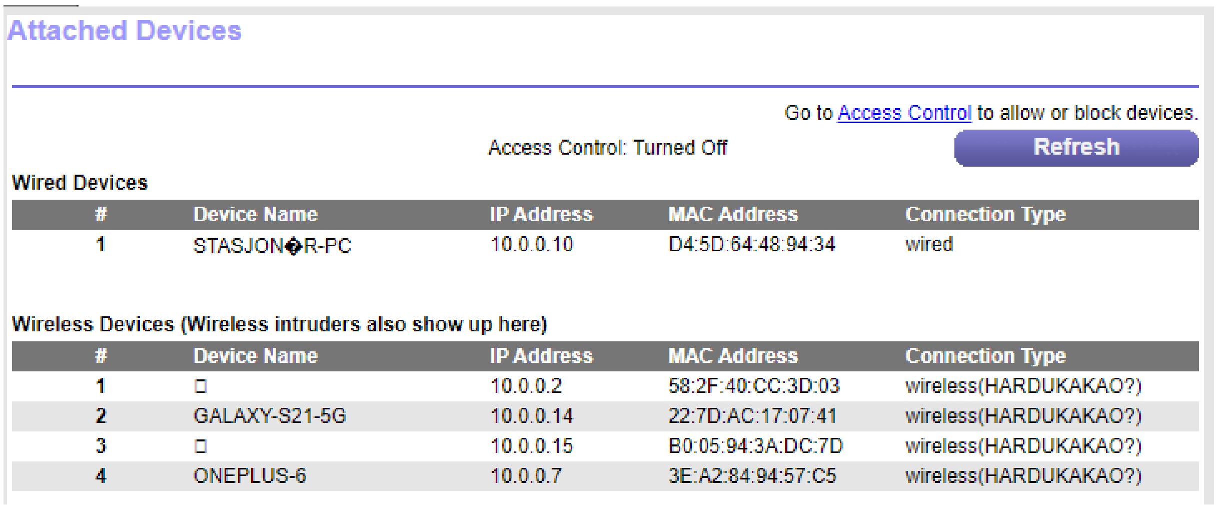
4.6 Implementation of the node checker ping application

A program was created to ping all the local nodes on the network. The source code can be found in Appendix B: testing ping to all local nodes. A screenshot of the console outputs can be seen in Figure 4-6. Though it does not seem to be able to get a response from all nodes. Figure 4-7 shows devices visible for the router and Figure 4-8 shows the ARP table of the network.



```
PS C:\Master\Industrial-information-technology\Data Communication\Node Application> dotnet run
 Ping communication status for 10.0.0.1:
 -----------------------------------------
 Address: 10.0.0.1
 RoundTrip time (mSec): 0
 Time to live: 64
 Don't fragment: False
 Buffer size: 17
 -----------------------------------------
 Ping communication status for 10.0.0.10:
 -----------------------------------------
 Address: 10.0.0.10
 RoundTrip time (mSec): 0
 Time to live: 128
 Don't fragment: False
 Buffer size: 17
 -----------------------------------------
 Ping communication status for 10.0.0.15:
 -----------------------------------------
 Address: 10.0.0.15
 RoundTrip time (mSec): 77
 Time to live: 64
 Don't fragment: True
 Buffer size: 17
 -----------------------------------------
 Press Enter to Quit the application
```

Figure 4-6 Outputs of the node checker application



Attached Devices

Go to Access Control to allow or block devices.

Access Control: Turned Off

Refresh

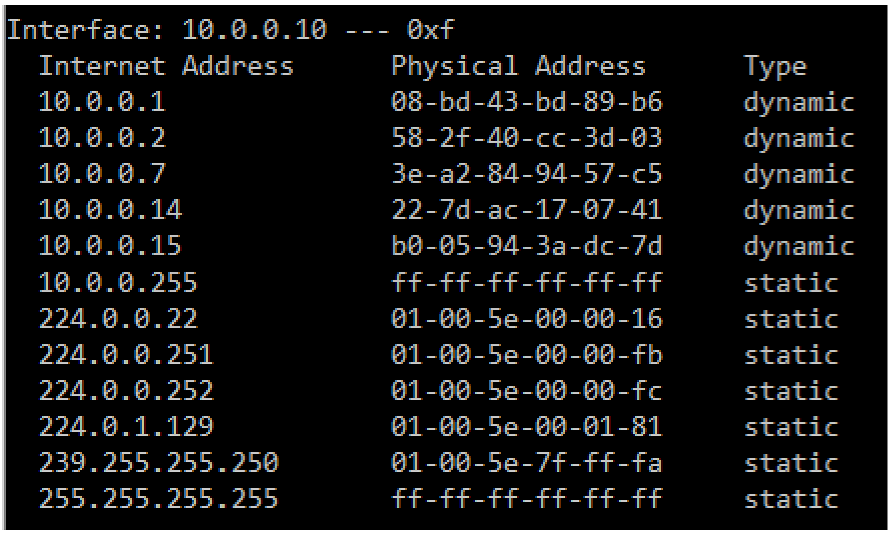
Wired Devices

#	Device Name	IP Address	MAC Address	Connection Type
1	STASJON�R-PC	10.0.0.10	D4:5D:64:48:94:34	wired

Wireless Devices (Wireless intruders also show up here)

#	Device Name	IP Address	MAC Address	Connection Type
1	▢	10.0.0.2	58:2F:40:CC:3D:03	wireless(HARDUKAKAO?)
2	GALAXY-S21-5G	10.0.0.14	22:7D:AC:17:07:41	wireless(HARDUKAKAO?)
3	▢	10.0.0.15	B0:05:94:3A:DC:7D	wireless(HARDUKAKAO?)
4	ONEPLUS-6	10.0.0.7	3E:A2:84:94:57:C5	wireless(HARDUKAKAO?)

Figure 4-7 All devices connected to the Router



```
Interface: 10.0.0.10 --- 0xf
  Internet Address      Physical Address      Type
  10.0.0.1              08-bd-43-bd-89-b6     dynamic
  10.0.0.2              58-2f-40-cc-3d-03     dynamic
  10.0.0.7              3e-a2-84-94-57-c5     dynamic
  10.0.0.14             22-7d-ac-17-07-41     dynamic
  10.0.0.15             b0-05-94-3a-dc-7d     dynamic
  10.0.0.255            ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.1.129           01-00-5e-00-01-81     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Figure 4-8 Outputs of the "arp -a" cmd command

5 Conclusion

We can gather information about our computers and our networks, but features in Windows change such as the device manager, it being because of a lack of maintenance or change of features. It seems information that the assignment asks for is not presented there in the way it might once was. We also try to ping all nodes on the local network, but not all are visible to the computer, which was perhaps not as expected.

6 References

[1]	"List of Bluetooth profiles," [Online]. Available: https://en.wikipedia.org/wiki/List_of_Bluetooth_profiles#Audio/Video_Remote_Control_Profile_(AVI [Accessed 29 01 2022].
[2]	"Master/slave (technology)," 23 January 2022. [Online]. Available: https://en.wikipedia.org/wiki/Master/slave_(technology). [Accessed 28 January 2022].
[3]	"Understand TCP/IP addressing and subnetting basics," Microsoft, 23 09 2021. [Online]. Available: https://docs.microsoft.com/en-us/troubleshoot/windows-client/networking/tcpip-addressing-and-subnetting. [Accessed 28 01 2022].

7 Appendices

Appendix A: Testing ping to a connected node
Appendix B: testing ping to all local nodes

8 Appendix A: Testing ping to a connected node

```
////////////////////////////////////////////////////////////////////////////
////////////
///
/// PingAppConsole: application testing the ping() connection to a node
///
/// Based on code form Microsoft MSDN about the ping() command
///
/// Version: 1.1: 28-JAN-22: LRR
///
////////////////////////////////////////////////////////////////////////////
////////////
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 PingAppConsole
{
    class Program
    {
        /// <summary>
        ///
////////////////////////////////////////////////////////////////////////////
        static void Main(string[] args)
        ///
        /// Purpose: the main function in the application handling the
ping()communication
        ///
        /// Version: 1.1: 28-JAN-22: LRR
        /// </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 = "UniversityCollegeofSouthEastNorw";
            buffer = Encoding.ASCII.GetBytes(data);
            timeout = 120;
            // Name or address of node to access
            host = "www.twitch.tv";
            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();
        }
    }
}
///
/////////////////////////////////////////////////////////////////////////////
//////////
```

9 Appendix B: testing ping to all local nodes

```
////////////////////////////////////////////////////////////////////////////////
/////////////
///
/// PingNodes: application testing the ping() connection with all nodes
///
/// Based on code form Microsoft MSDN about the ping() command
///
/// Version: 1.1: 28-JAN-22: LRR
///
////////////////////////////////////////////////////////////////////////////////
/////////////
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 PingAppConsole
{
    class Program
    {
        /// <summary>
        ///
////////////////////////////////////////////////////////////////////////////////
        static void Main(string[] args)
        ///
        /// Purpose: the main function in the application handling the ping()
communication
        ///
        /// Version: 1.1: 28-JAN-22: LRR
        /// </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 = "NotASecretMessage";
            buffer = Encoding.ASCII.GetBytes(data);
            timeout = 120;

            var replies = new List<PingReply> { };
            for (int i = 0; i < 256; i++)
            {
                // Name or address of node to access
                host = $"10.0.0.{i}";
                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(" --------------------------------------
----");
                }
            }
            Console.WriteLine(" Press Enter to Quit the application");
            Console.ReadLine();
        }
    }
}
///
//////////////////////////////////////////////////////////////////////////////
//////////
```