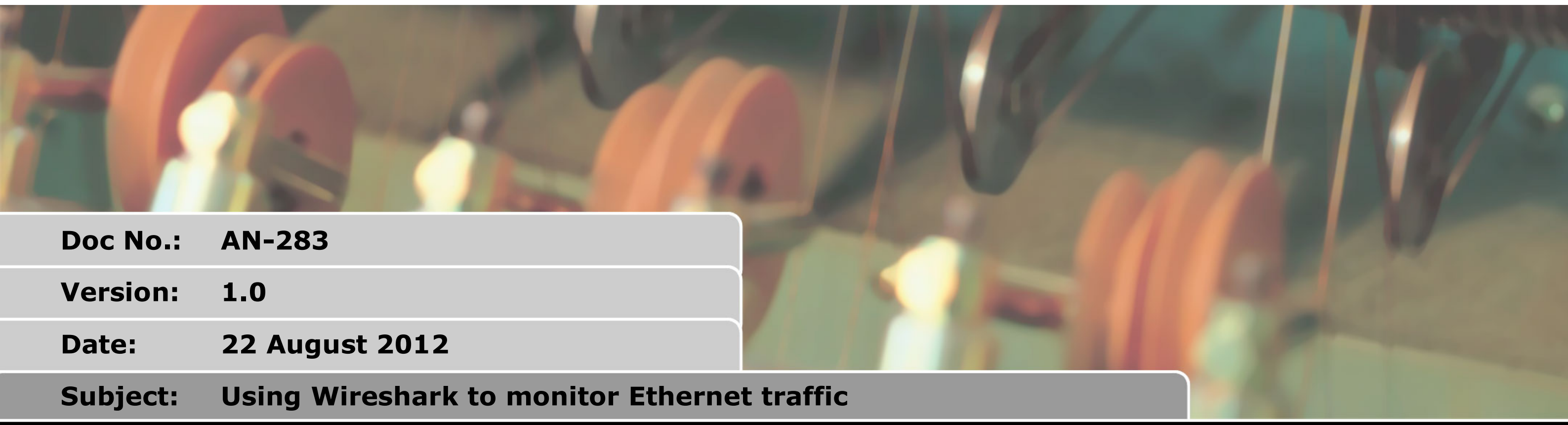
Shannon Way,
Tewkesbury,
Gloucestershire. GL20 8ND
United Kingdom
Tel: +44 (0)1684 292 333
Fax: +44 (0)1684 297 929

187 Northpointe Blvd,
Suite 105
Freeport, PA 16229
United States of America
Tel: +1 724-540-5018
Fax: +1 724-540-5098

Tomson Centre
118 Zhang Yang Rd., B1701
Pudong New Area, Shanghai,
Postal code: 200122
CHINA
Tel/Fax: +86 21 587 97659

SCMC House
16/6 Vishal Nagar
Pimpale Nilakh, Wakad, Pune
PIN 411027
INDIA
Tel: +91 206 811 4902



Doc No.:	**AN-283**
Version:	**1.0**
Date:	**22 August 2012**
Subject:	**Using Wireshark to monitor Ethernet traffic**

APPLICATION NOTE

1. Introduction

Ethernet can carry many types of telegram. When connecting a system together, it is often necessary to monitor the telegram traffic and see how the available bandwidth is being used. This is especially true when a real-time automation system is using Ethernet for part of its real-time functionality. Effects such as collisions, re-tries, acknowledge and handshake cycles all have an effect on the overall performance of the network.

This document describes how to set up an Ethernet Monitor using the free-to-download package "WireShark".

2. Equipment

The following equipment is required.

2.1. WireShark

Download Wireshark from http://www.wireshark.org/

Install the package on a suitable Windows PC. (The PC must have at least one Ethernet port)

2.2. Semi-managed Ethernet Switch

Modbus TCP uses TCP/IP telegrams which have a defined source and defined destination IP_Address. A standard Ethernet Switch will not pass these telegrams to the other ports on the switch. Therefore a managed or partly managed switch is needed which can set up one port to be a "mirror port" to re-transmit all telegrams received on the other ports.

An example of such a switch is the NetGear ProSafe Plus GS105E. This comes with software which allows the user to set up special features such as mirror ports.

An alternative to a managed switch is to use a plain Hub. A Hub is a dumb device which re-transmits all telegrams to all ports. Unfortunately Hubs are very rare and are not found on the market any more.

3. Set-up the switch

The set up shown here refers to the Netgear GS105E. Other managed and semi-managed switches will have a similar set up procedure.

3.1. Install the software

Install the switch management software on a PC. In the case of the Netgear GS105E, this is called NETGEAR UM+ Utility.

3.2. Set up the mirror port

Run the management software and follow the procedure in the operation manual for the switch.

In the case of the Netgear GS105E this is:

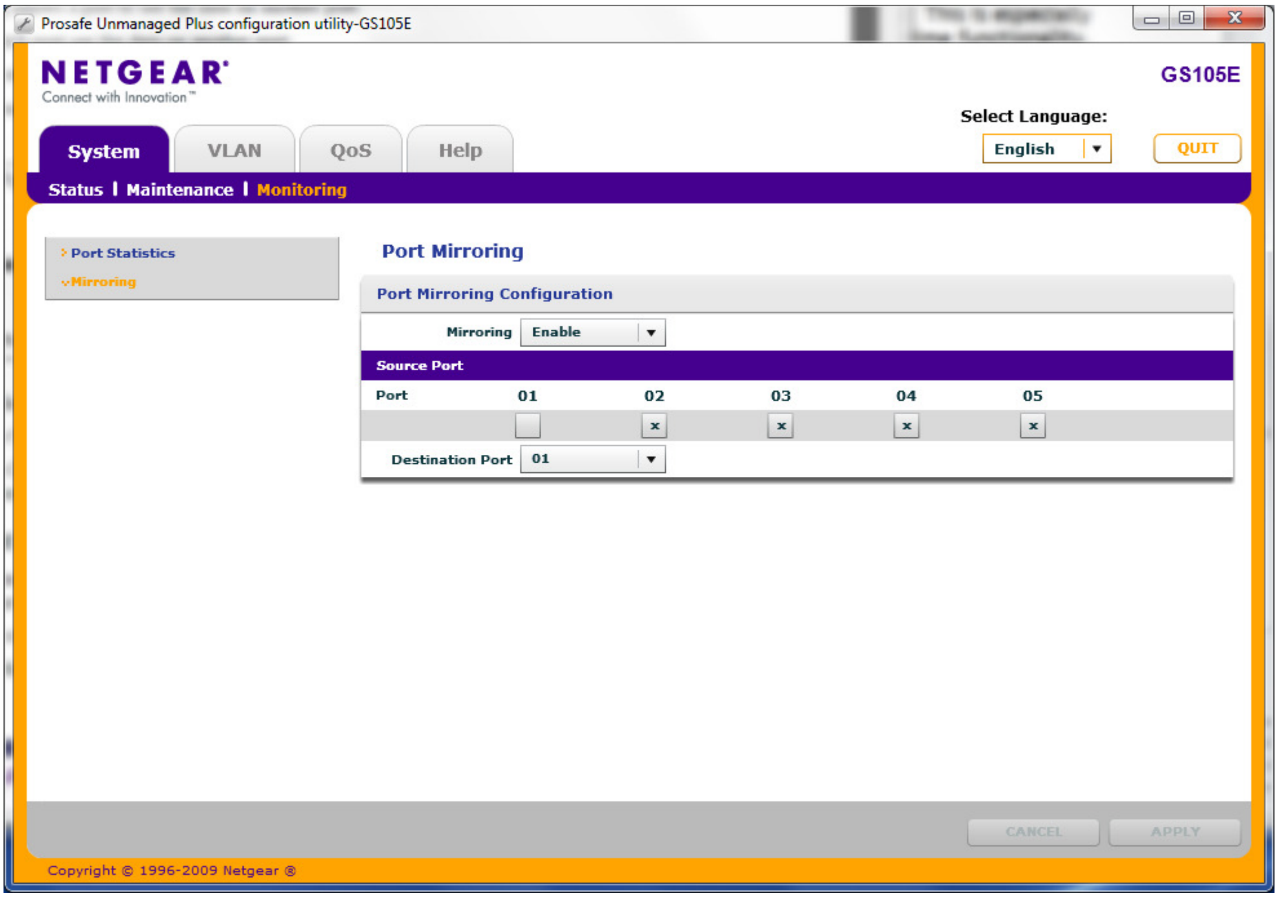
3.2.1. Port Mirroring

Port mirroring allows a port to see the data on another port.

To have a switch port see the data on another port:

1. Select System > Status, and select the switch.
2. Select Monitoring > Mirroring. The Mirroring page will display
3. Enable Mirroring.
4. Select the Source Port or ports.
5. Select the Destination Port from the pull down list and click Apply. Data on the source port will now also be routed to the destination port.

3.2.2. Screenshot



Select Language: English

Port Mirroring

Port Mirroring Configuration

Mirroring: Enable

Source Port

Port	01	02	03	04	05
		x	x	x	x

Destination Port: 01

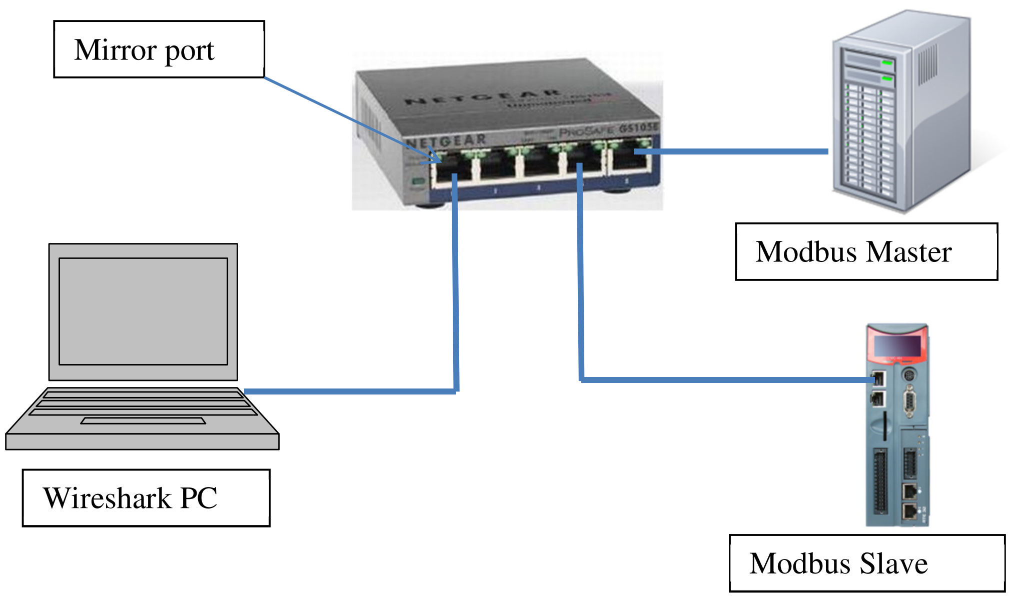
4. Run Wireshark

4.1. Connections

Connect the mirror port of the Switch to the PC which will run Wireshark.

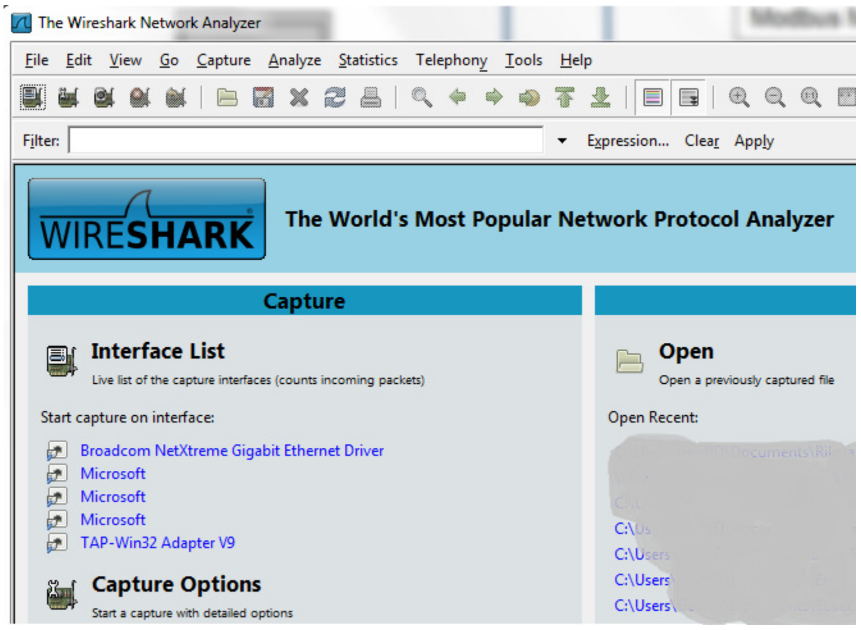
Connect one of the other ports to the Modbus master.

Connect one of the other ports to the Modbus slave.



4.2. Launch Wireshark

Launch Wireshark and select the Ethernet Port on the PC which is connected to the Switch.


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Capture
Open
Open a previously captured file
Open Recent:

Interface List

Live list of the capture interfaces (counts incoming packets)

Start capture on interface:

- Broadcom NetXtreme Gigabit Ethernet Driver
- Microsoft
- Microsoft
- Microsoft
- TAP-Win32 Adapter V9

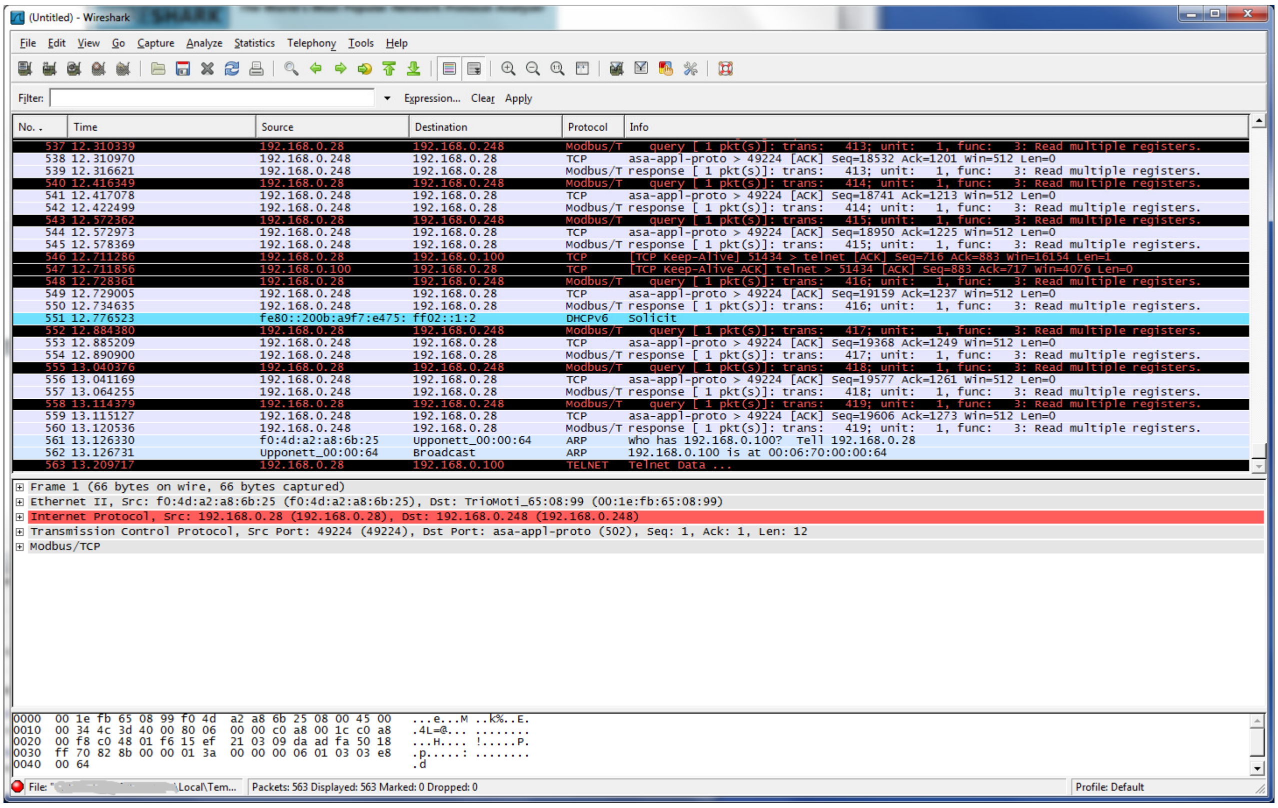
Capture Options

Start a capture with detailed options

In the above example, Broadcom NetXtreme is connected to the switch.

4.3. *Start monitoring*

As soon as the Ethernet port is selected, the Wireshark will begin to monitor the Ethernet data.


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
Profile: Default

```
No. .  Time       Source                   Destination          Protocol  Info
  537 12.310339  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   413; unit:   1, func:   3: Read multiple registers.
  538 12.310970  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=18532 Ack=1201 Win=512 Len=0
  539 12.316621  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   413; unit:   1, func:   3: Read multiple registers.
  540 12.416349  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   414; unit:   1, func:   3: Read multiple registers.
  541 12.417078  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=18741 Ack=1213 Win=512 Len=0
  542 12.422499  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   414; unit:   1, func:   3: Read multiple registers.
  543 12.572362  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   415; unit:   1, func:   3: Read multiple registers.
  544 12.572973  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=18950 Ack=1225 Win=512 Len=0
  545 12.578369  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   415; unit:   1, func:   3: Read multiple registers.
  546 12.711286  192.168.0.28             192.168.0.100        TCP       [TCP Keep-Alive] 51434 > telnet [ACK] Seq=716 Ack=883 Win=16154 Len=1
  547 12.711856  192.168.0.100            192.168.0.28         TCP       [TCP Keep-Alive ACK] telnet > 51434 [ACK] Seq=883 Ack=717 Win=4076 Len=0
  548 12.728361  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   416; unit:   1, func:   3: Read multiple registers.
  549 12.729005  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=19159 Ack=1237 Win=512 Len=0
  550 12.734635  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   416; unit:   1, func:   3: Read multiple registers.
  551 12.776523  fe80::200b:a9f7:e475: ff02::1:2            DHCPv6    Solicit
  552 12.884380  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   417; unit:   1, func:   3: Read multiple registers.
  553 12.885209  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=19368 Ack=1249 Win=512 Len=0
  554 12.890900  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   417; unit:   1, func:   3: Read multiple registers.
  555 13.040376  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   418; unit:   1, func:   3: Read multiple registers.
  556 13.041169  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=19577 Ack=1261 Win=512 Len=0
  557 13.064255  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   418; unit:   1, func:   3: Read multiple registers.
  558 13.114379  192.168.0.28             192.168.0.248        Modbus/T    query [ 1 pkt(s)]: trans:   419; unit:   1, func:   3: Read multiple registers.
  559 13.115127  192.168.0.248            192.168.0.28         TCP       asa-appl-proto > 49224 [ACK] Seq=19606 Ack=1273 Win=512 Len=0
  560 13.120536  192.168.0.248            192.168.0.28         Modbus/T response [ 1 pkt(s)]: trans:   419; unit:   1, func:   3: Read multiple registers.
  561 13.126330  f0:4d:a2:a8:6b:25        Upponett_00:00:64    ARP       Who has 192.168.0.100?  Tell 192.168.0.28
  562 13.126731  Upponett_00:00:64        Broadcast            ARP       192.168.0.100 is at 00:06:70:00:00:64
  563 13.209717  192.168.0.28             192.168.0.100        TELNET    Telnet Data ...
```

```
⊞ Frame 1 (66 bytes on wire, 66 bytes captured)
⊞ Ethernet II, Src: f0:4d:a2:a8:6b:25 (f0:4d:a2:a8:6b:25), Dst: TrioMoti_65:08:99 (00:1e:fb:65:08:99)
⊞ Internet Protocol, Src: 192.168.0.28 (192.168.0.28), Dst: 192.168.0.248 (192.168.0.248)
⊞ Transmission Control Protocol, Src Port: 49224 (49224), Dst Port: asa-appl-proto (502), Seq: 1, Ack: 1, Len: 12
⊞ Modbus/TCP
```

```
0000  00 1e fb 65 08 99 f0 4d  a2 a8 6b 25 08 00 45 00   ...e...M ..k%..E.
0010  00 34 4c 3d 40 00 80 06  00 00 c0 a8 00 1c c0 a8   .4L=@... ........
0020  00 f8 c0 48 01 f6 15 ef  21 03 09 da ad fa 50 18   ...H.... !.....P.
0030  ff 70 82 8b 00 00 01 3a  00 00 00 06 01 03 03 e8   .p.....: ........
0040  00 64                                              .d
```

File: "[redacted]\Local\Tem... Packets: 563 Displayed: 563 Marked: 0 Dropped: 0

The monitoring can be paused or stopped with the buttons at the top of the window.

4.4. *Save captured data*

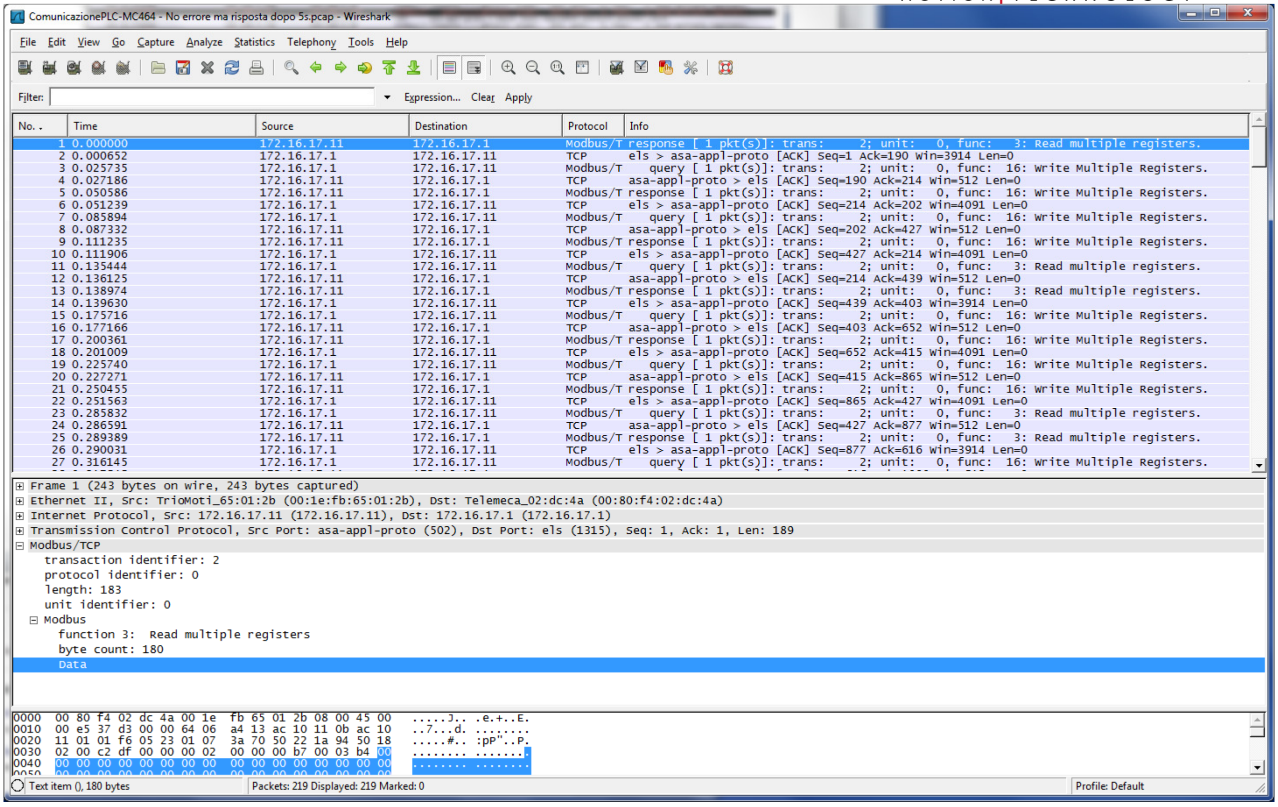
When enough data has been captured, stop the monitor and save the log to a Wireshark capture file.

This file can be sent to Trio for analysis if requested.

5. Loading previously saved files

A file saved from a previous monitoring session can be loaded into Wireshark for analysis.

Wireshark recognises the telegram types and provides a clear display of the telegram sequence. Any re-transmissions, timeouts and other potential errors are highlighted.


ComunicazionePLC-MC464 - No errore ma risposta dopo 5s.pcap - Wireshark

```
No. . Time       Source         Destination    Protocol   Info
    1 0.000000   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:   3: Read multiple registers.
    2 0.000652   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=1 Ack=190 Win=3914 Len=0
    3 0.025735   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
    4 0.027186   172.16.17.11   172.16.17.1    TCP      asa-appl-proto > els [ACK] Seq=190 Ack=214 Win=512 Len=0
    5 0.050586   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
    6 0.051239   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=214 Ack=202 Win=4091 Len=0
    7 0.085894   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
    8 0.087332   172.16.17.11   172.16.17.1    TCP      asa-appl-proto > els [ACK] Seq=202 Ack=427 Win=512 Len=0
    9 0.111235   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
   10 0.111906   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=427 Ack=214 Win=4091 Len=0
   11 0.135444   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:   3: Read multiple registers.
   12 0.136125   172.16.17.11   172.16.17.1    TCP      asa-appl-proto > els [ACK] Seq=214 Ack=439 Win=512 Len=0
   13 0.138974   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:   3: Read multiple registers.
   14 0.139630   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=439 Ack=403 Win=3914 Len=0
   15 0.175716   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
   16 0.177166   172.16.17.11   172.16.17.1    TCP      asa-appl-proto > els [ACK] Seq=403 Ack=652 Win=512 Len=0
   17 0.200361   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
   18 0.201009   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=652 Ack=415 Win=4091 Len=0
   19 0.225740   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
   20 0.227271   172.16.17.11   172.16.17.1    TCP      asa-appl-proto > els [ACK] Seq=415 Ack=865 Win=512 Len=0
   21 0.250455   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
   22 0.251563   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=865 Ack=427 Win=4091 Len=0
   23 0.285832   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:   3: Read multiple registers.
   24 0.286591   172.16.17.11   172.16.17.1    TCP      asa-appl-proto > els [ACK] Seq=427 Ack=877 Win=512 Len=0
   25 0.289389   172.16.17.11   172.16.17.1    Modbus/T response [ 1 pkt(s)]: trans:     2; unit:   0, func:   3: Read multiple registers.
   26 0.290031   172.16.17.1    172.16.17.11   TCP      els > asa-appl-proto [ACK] Seq=877 Ack=616 Win=3914 Len=0
   27 0.316145   172.16.17.1    172.16.17.11   Modbus/T    query [ 1 pkt(s)]: trans:     2; unit:   0, func:  16: Write Multiple Registers.
```

```
⊞ Frame 1 (243 bytes on wire, 243 bytes captured)
⊞ Ethernet II, Src: TrioMoti_65:01:2b (00:1e:fb:65:01:2b), Dst: Telemeca_02:dc:4a (00:80:f4:02:dc:4a)
⊞ Internet Protocol, Src: 172.16.17.11 (172.16.17.11), Dst: 172.16.17.1 (172.16.17.1)
⊞ Transmission Control Protocol, Src Port: asa-appl-proto (502), Dst Port: els (1315), Seq: 1, Ack: 1, Len: 189
⊟ Modbus/TCP
    transaction identifier: 2
    protocol identifier: 0
    length: 183
    unit identifier: 0
  ⊟ Modbus
      function 3:  Read multiple registers
      byte count: 180
      Data
```

```
0000  00 80 f4 02 dc 4a 00 1e  fb 65 01 2b 08 00 45 00   .....J.. .e.+..E.
0010  00 e5 37 d3 00 00 64 06  a4 13 ac 10 11 0b ac 10   ..7...d. ........
0020  11 01 01 f6 05 23 01 07  3a 70 50 22 1a 94 50 18   .....#.. :pP"..P.
0030  02 00 c2 df 00 00 00 02  00 00 00 b7 00 03 b4 00   ........ ........
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

Text item (), 180 bytes   Packets: 219 Displayed: 219 Marked: 0   Profile: Default

A typical Modbus communication sequence between a PLC and a MC464.