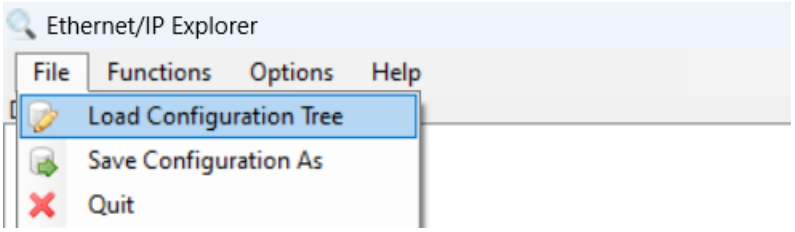
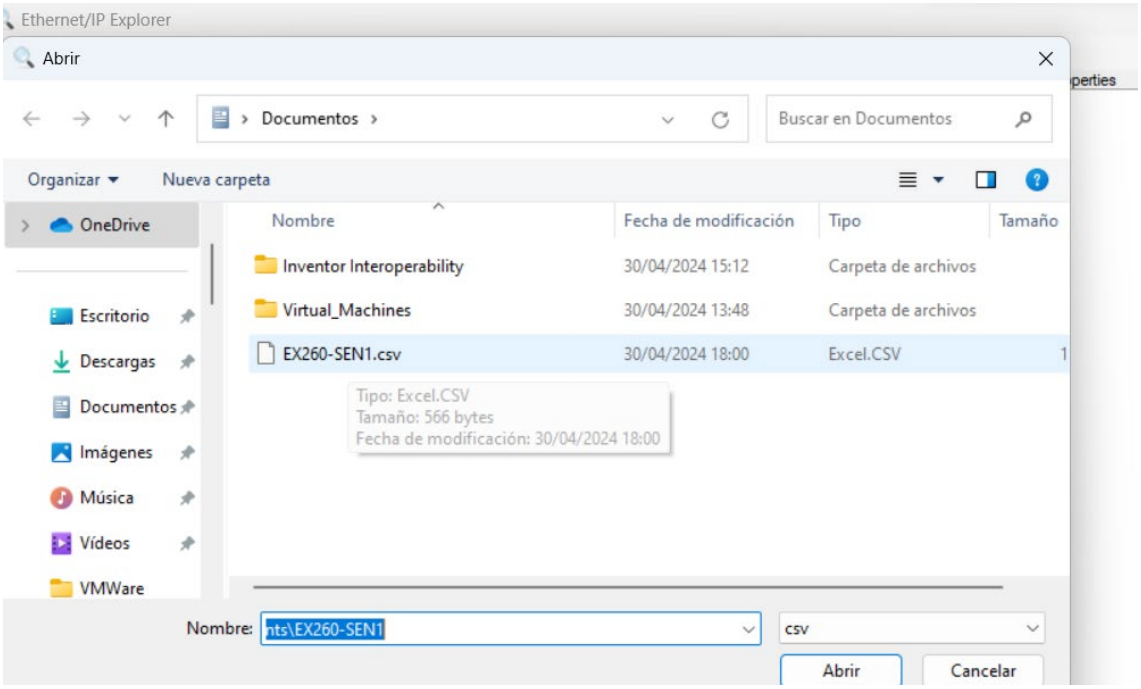
Una vez establecida la IP, hacemos ping para comprobar su conexión.

```
C:\Program Files (x86)\VMware\VMware Workstation\bin>ping 192.168.0.202

Haciendo ping a 192.168.0.202 con 32 bytes de datos:
Respuesta desde 192.168.0.202: bytes=32 tiempo<1m TTL=60
Respuesta desde 192.168.0.202: bytes=32 tiempo=1ms TTL=60
Respuesta desde 192.168.0.202: bytes=32 tiempo=1ms TTL=60
Respuesta desde 192.168.0.202: bytes=32 tiempo=1ms TTL=60

Estadísticas de ping para 192.168.0.202:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 0ms, Máximo = 1ms, Media = 0ms
```

Ejecutamos EnIPExplorer:

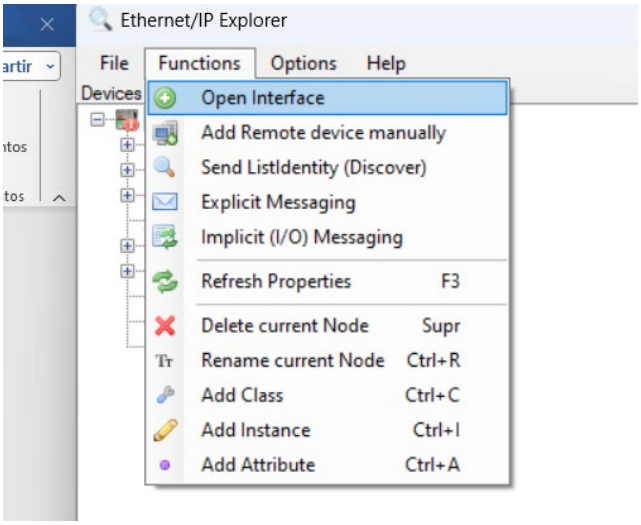
Cargamos el archivo correspondiente, pero también podemos crearlo de cero:



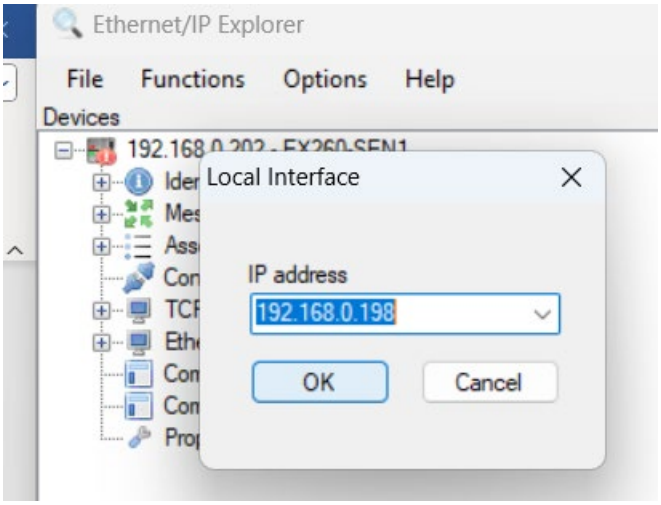


Electroválvula SMC EX260-SEN1

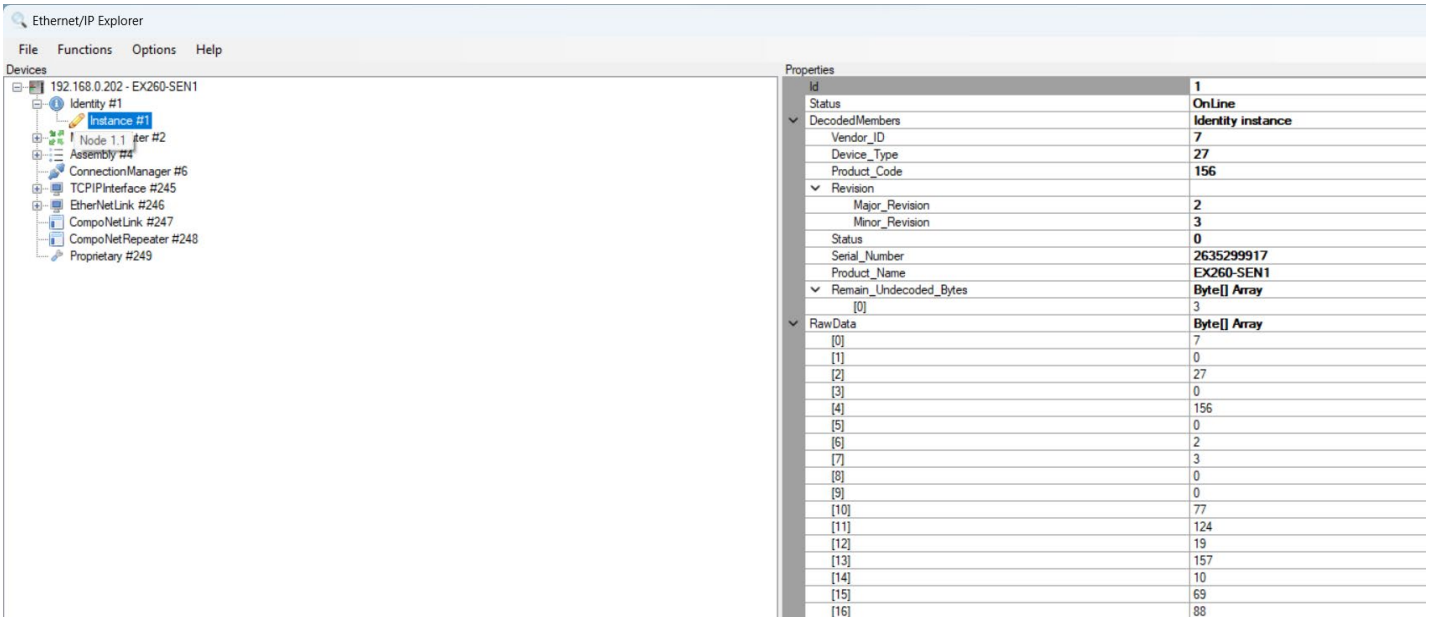
Una vez cargado, abrimos la conexión:



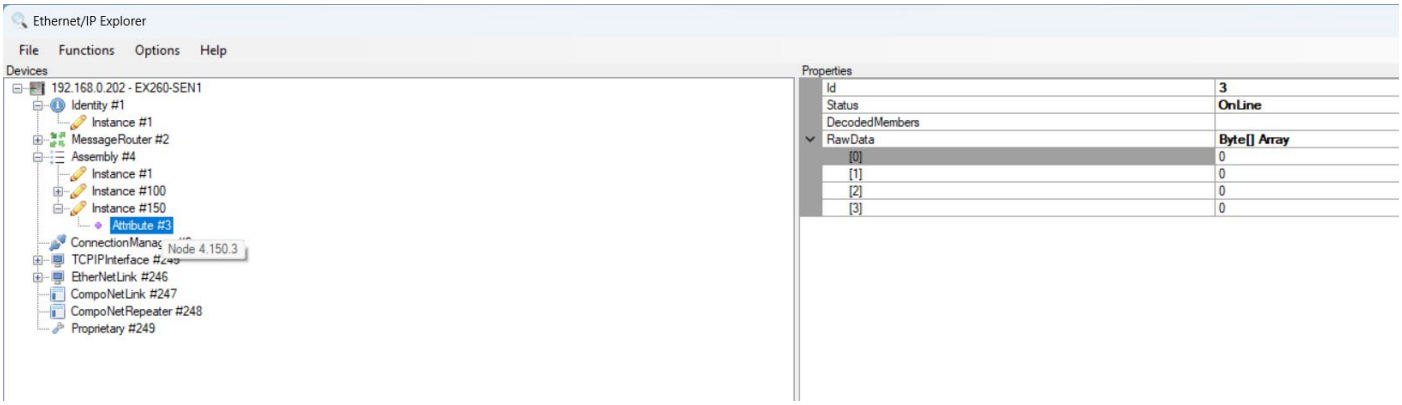
Nos pedirá la IP de nuestro Host:



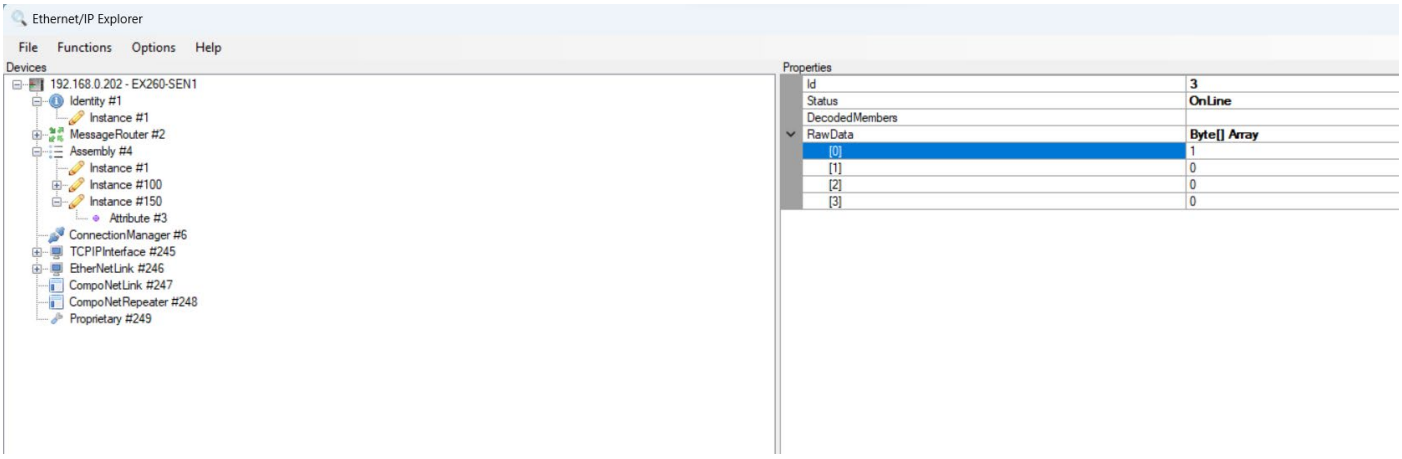
Una vez abierta la conexión, podemos ver la identidad y los atributos, que también están representados en el archivo de configuración EDS.

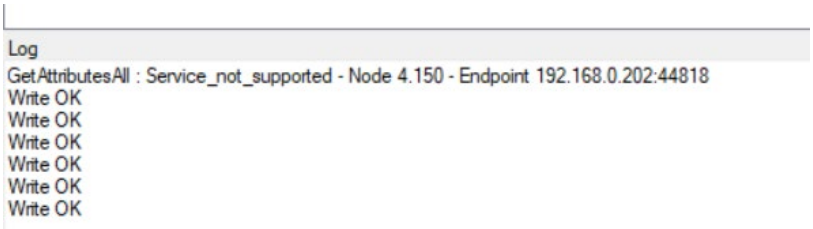


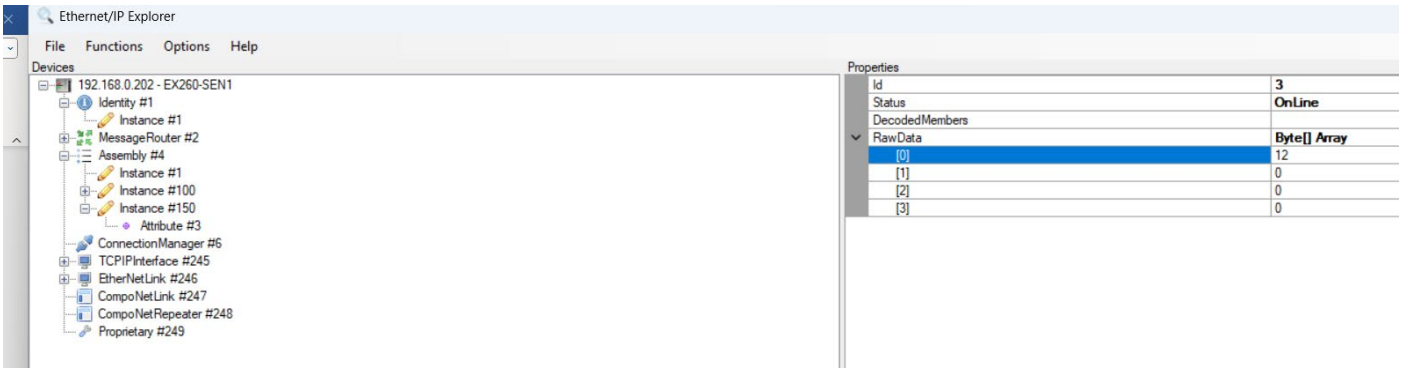
Para escribir necesitamos ir al área de salida, para ello debemos ir a la Assembly #4, Interface #150 y Attribute #3, para escribir debemos utilizar la primera posición del Byte[] Array que esta asignada al Byte[0].



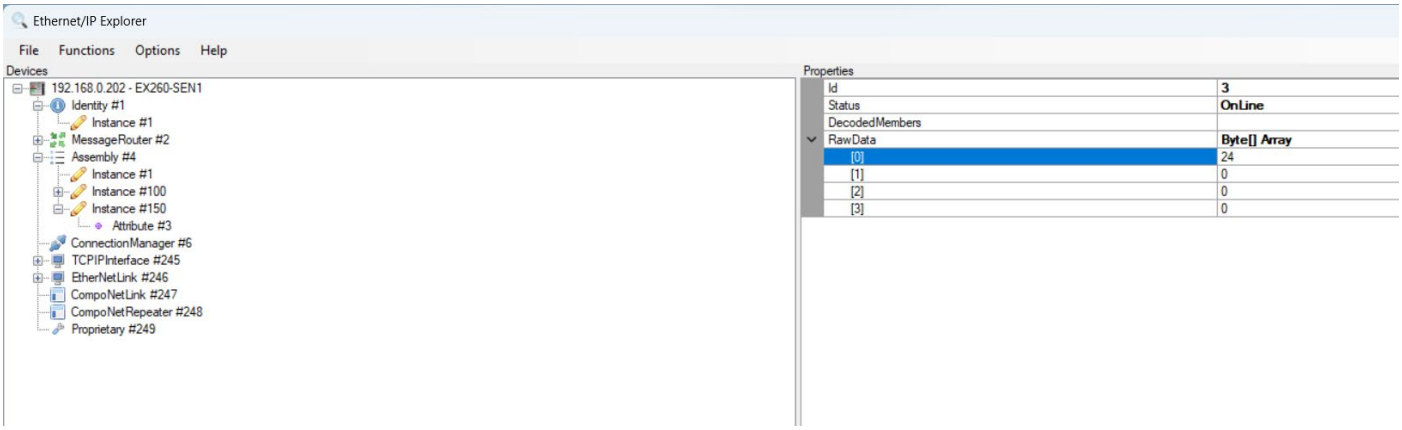
Ejemplos de escritura:



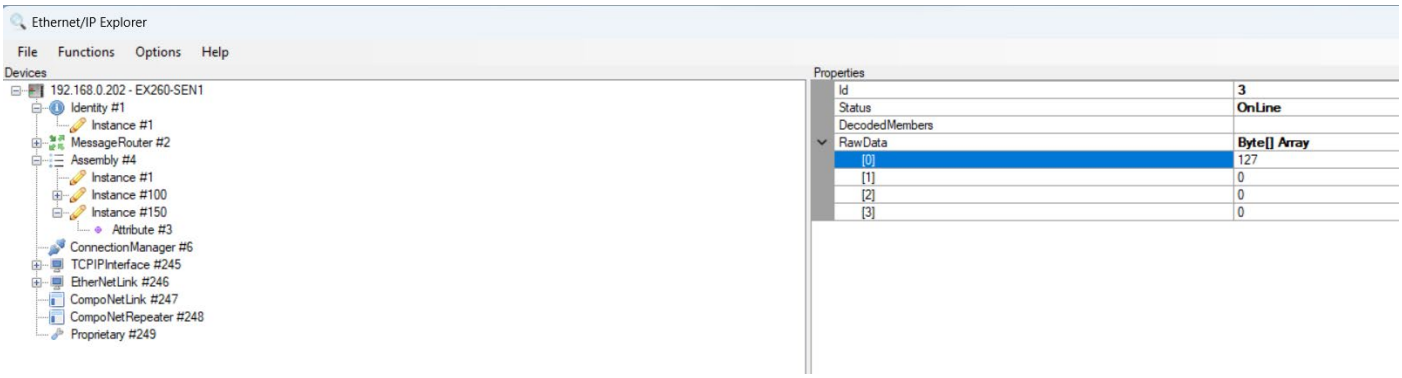




Electroválvula SMC EX260-SEN1



Para escribir en el estado de todas, utilizamos el entero 127.



Electroválvula SMC EX260-SEN1

Para poder manipularlas hay varias librerías en diferentes lenguajes, disponibles en GitHub, ejemplo de la librería cpppo de Python:

En el nos conectamos a la electroválvula y especificamos en @4/150/3, el Assembly/Instance y Attribute.

import cpppo.py > ...

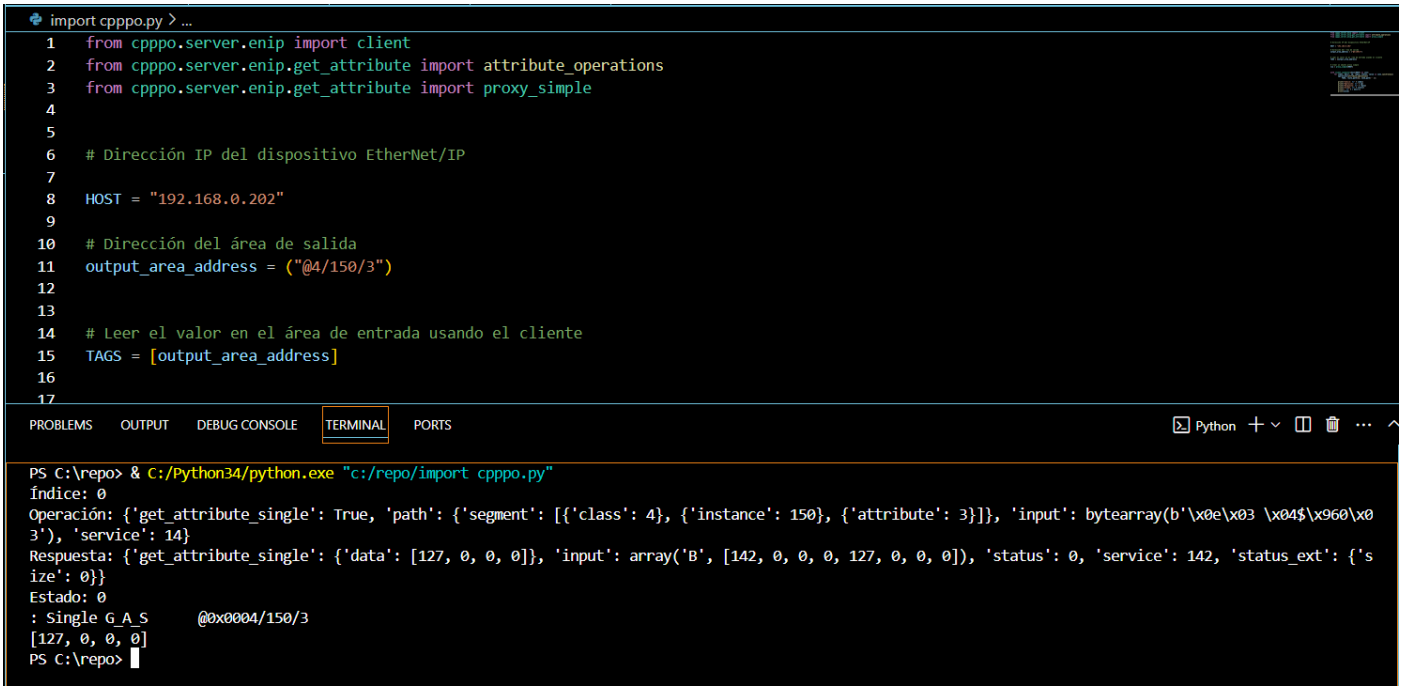
```python
from cpppo.server.enip import client
from cpppo.server.enip.get_attribute import attribute_operations
from cpppo.server.enip.get_attribute import proxy_simple


# Dirección IP del dispositivo EtherNet/IP

HOST = "192.168.0.202"

# Dirección del área de salida
output_area_address = ("@4/150/3")


# Leer el valor en el área de entrada usando el cliente
TAGS = [output_area_address]


# Crear un objeto proxy simple
via = proxy_simple(HOST)


with client.connector(host=HOST) as conn:
    for index, descr, op, reply, status, value in conn.synchronous(
        operations=attribute_operations(
            TAGS, route_path=[], send_path='' )):

        print("Índice: %s" % index)
        print("Operación: %s" % op)
        print("Respuesta: %s" % reply)
        print("Estado: %s" % status)
        print(": %s" % ( descr))
        print(value)
```

Salida del programa con las electroválvulas en 0:

```
PS C:\repo> & C:/Python34/python.exe "c:/repo/import cpppo.py"
Índice: 0

Operación: {'service': 14, 'input': bytearray(b'\x0e\x03 \x04$\x960\x03'), 'get_attribute_single': True, 'path': {'segment': [{'class': 4}, {'instance': 150}, {'attribute': 3}]}}

Respuesta: {'input': array('B', [142, 0, 0, 0, 0, 0, 0, 0]), 'service': 142, 'status': 0, 'get_attribute_single': {'data': [0, 0, 0, 0]}, 'status_ext': {'size': 0}}

Estado: 0

: Single G_A_S      @0x0004/150/3

[0, 0, 0, 0]
```

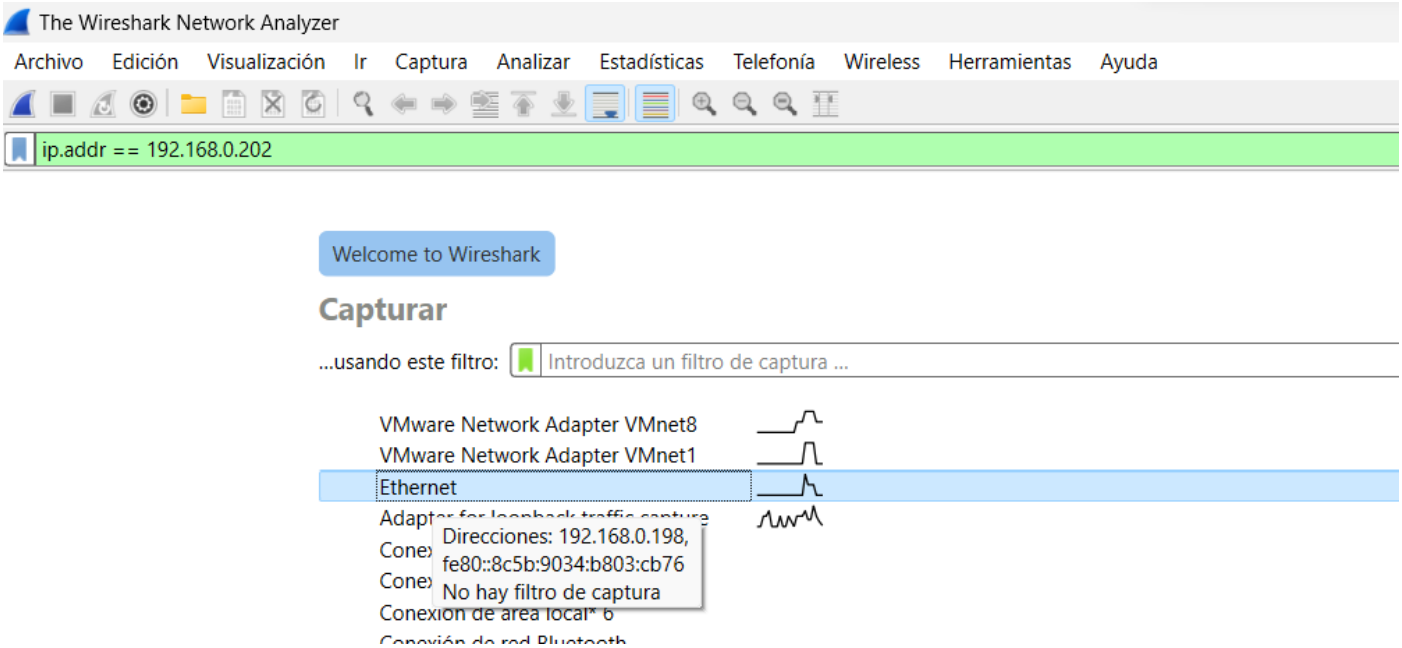
Electroválvula SMC EX260-SEN1

Salida del programa con las electroválvulas en 127:



Capturar tramas para utilizar en diferentes lenguajes con Wireshark

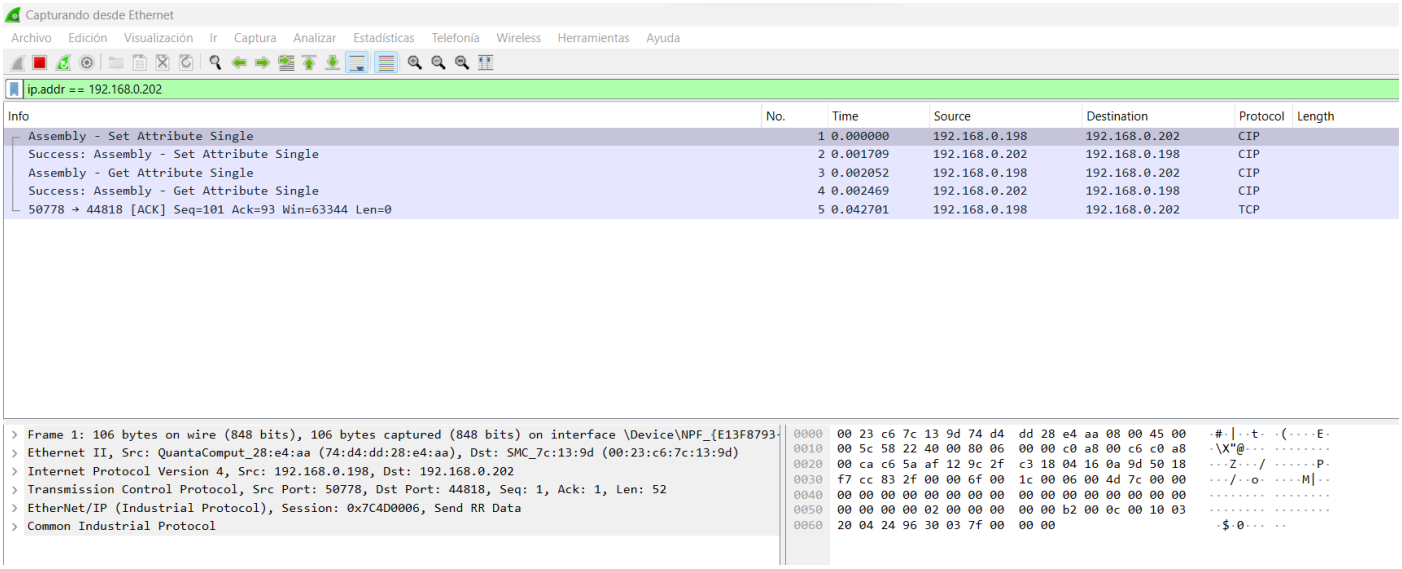
Colocamos de filtro el ip.addr == IP de la electroválvula.



Electroválvula SMC EX260-SEN1

En esta primera trama podemos capturar cuando las valvulas se activan todas a la vez, en el Set Attribute Single, una vez activada nos devolverá el estado actualizado de la valvula mediante Get.

En la línea 0060 podemos ver la ruta especificada en el archivo EDS que es 20 04 24 96 30 03 y la activación que corresponde a 7F 00 00 00



Ahora hacemos lo mismo pero desactivando todas las válvulas, entonces vemos en la línea 0060 la modificación en 20 04 24 96 30 03 y la activación que corresponde a 00 00 00 00