Metasploit Modbus Scanning

The **Metasploit** tool contains several Modbus modules. These modules include identifying devices running a Modbus server, extracting data from Modbus packet captures, and exploiting known vulnerabilities in common Program Logic Controlers (PLC).

- **auxiliary/ admin/scada/modicon_comman**: Schneider Modicon Remote START/STOP Command
- **auxiliary/ admin/scada/modicon_stux_transfer**: Schneider Modicon Ladder Logic Upload/Download
- **auxiliary/ analyze/modbus_zip**: Extract zip from Modbus communication
- **auxiliary/ scanner/scada/modbus_findunitid**: Modbus Unit ID and Station ID Enumerator
- **auxiliary/ scanner/scada/modbusclient**: Modbus Client Utility
- **auxiliary/ scanner/scada/modbusdetect**: Modbus Version Scanner

The following example is the **modbus_findunitid** module. This module will scan a subnet for devices with the Modbus service and enumerate the Unit ID field in the Modbus protocol.

```
control@ctp:~$ msfconsole
msf5 > use
auxiliary/scanner/scada/modbus_findunitid

msf5 auxiliary(scanner/scada/modbus_findunitid)
> set RHOSTS 127.0.0.1
RHOSTS => 127.0.0.1

msf5 auxiliary(scanner/scada/modbus_findunitid)
> set RPORT 10502
RPORT => 10502

msf5 auxiliary(scanner/scada/modbus_findunitid)
> set UNIT_ID_TO 2
UNIT_ID_TO => 2

msf5 auxiliary(scanner/scada/modbus_findunitid)
> run
[*] Running module against 127.0.0.1
[+] 127.0.0.1:10502 - Received: correct
MODBUS/TCP from stationID  1
[*] 127.0.0.1:10502 - Received: incorrect/none
data from stationID 2 (probably not in use)
```

Modbus Network Analysis

Wireshark and **Tshark** are excellent tools for conducting network analysis of Modbus communications.

Wireshark Display Filters

NOTE: Modbus TCP allows for nested Modbus response / request. Thus, filters may contain read and write requests in the same packet.

Display Modbus packets

```
Modbus
```

Display Modbus TCP packets

```
mbtcp
```

Display Modbus function code

```
modbus.func_code < 5
```

Display all write functions

```
(modbus.func_code == 5) || (modbus.func_code
== 6) || (modbus.func_code == 15) ||
(modbus.func_code == 16) || (modbus.func_code
== 23)
```

Tshark Filters

NOTE: Tshark filters will only display the first nested Modbus response / request when outputting packet summary.

IPv4 Conversations

```
tshark -n -q -z conv,ip -Y modbus -r <file.pcap>
```

Server count

```
tshark -Y "mbtcp && tcp.dstport == 502" -T fields -e ip.dst
-r <file.pcap> | sort | uniq | wc -l
```

Servers by IP and Hardware Address

```
tshark -Y "mbtcp && tcp.dstport == 502" -T fields -e ip.dst
-e eth.dst -r <file.pcap> | sort | uniq
```

Client by IP Address

```
tshark -n -Y "mbtcp && tcp.dstport == 502" -T fields -e
ip.src -r <file.pcap> | sort | uniq
```

Clients with function codes

```
tshark -n -Y "mbtcp && tcp.dstport == 502" -T fields -e
ip.src -e modbus.func_code -r <file.pcap> | sort | uniq
```

Modbus RTU / TCP Cheat Sheet v1.0

SANS ICS
ics.sans.org

By Don C. Weber
don@cutawaysecurity.com

This guide covers the basics of using Modbus Remote Terminal Unit (RTU) and Modbus Transmission Control Protocol (TCP). It provides an outline of the packet formatting for both protocols. A table of the read and write function calls is provided.

The current Modbus specification can be found on the Modbus Organization website at: https://www.modbus.org. Other Modbus protocol versions are available with slightly different characteristics. See the Modbus specification or https://en.wikipedia.org/wiki/Modbus.

How to Use This Sheet

This cheat sheet will outline the following Modbus interactions for protocol review, device interactions, and network analysis.

Modbus RTU / TCP Descriptions

Modbus is a client-server protocol used to monitor and program devices, communicate between intelligent devices and sensors and instruments, and to monitor field devices using controllers, servers, workstations, and human machine interfaces.
Modbus RTU is the protocol standard that defines the use of Modbus across a serial connection. The serial connection typically uses a TIA-232 / TIA-485 serial interface for communication. The data exchange is a simple Protocol Data Unit (PDU) that includes a unit identifier (ID), function code, data request / response, and a Cyclical Redundancy Check (CRC).
Modbus TCP is a conversion of Modbus RTU to operate within a TCP payload. The Modbus server's default service port is 502. The data exchange is a simple PDU that includes a transaction ID, protocol ID, length field, unit ID, function code, and data request / response. The CRC is not included as it is provided by lower-level communication layers.

Modbus Function Codes

Bit Operations

- Read Coils: 1
- Read-Only Discrete Inputs: 2
- Write Single Coil: 5
- Write Multiple Coils: 15

Byte Operations

- Read Holding Registers: 3
- Read-Only Input Registers: 4
- Write Single Holding Register: 6
- Write Multiple Holding Registers: 16
- Read / Write Multiple Registers: 23

File Operations

- Read File Record: 20
- Write File Record: 21

Modbus Communication Format

Formatting varies slightly depending on the Modbus Function Code. The following examples outline Modbus read requests and responses.

Modbus RTU

Request	
Name	Length
Unit ID	1 byte
Function	1 byte
Start Address	2 bytes
Count	2 bytes
CRC	2 bytes

Response	
Name	Length
Unit ID	1 byte
Function	1 byte
Count	1 byte or by Function
Data	(Count * 2 bytes)
CRC	2 bytes

Modbus TCP

Request	
Name	Length
Transaction ID	2 bytes
Protocol ID	2 bytes
Length	2 bytes
Unit ID	1 byte
Function	1 byte
Start Address	2 bytes
Data	2 bytes or (Length – 4)

Response	
Name	Length
Transaction ID	2 bytes
Protocol ID	2 bytes
Length	2 bytes
Unit ID	1 byte
Function	1 byte
Count	1 byte or by Function
Data	2 bytes or (Length – 4)

ModbusTCP Get (mbtget) Usage

The **mbtget** tool is maintained at: https://github.com/sourceperl/mbtget. This tool provides a simple client that interacts with a Modbus server.

Help mbtget

```
control@ctp:~$ mbtget -h
usage : mbtget [-hvdsf]
               [-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)
  -r7               : read exception status
  -f                : set floating point value
  -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)
```

Read Registers with mbtget

```
control@ctp:~$ mbtget -r3 -a 0 -n 5 -p 10502
127.0.0.1
values:
  1 (ad 00000):   117
  2 (ad 00001):   120
  3 (ad 00002):   110
  4 (ad 00003):   130
  5 (ad 00004):     0
```