Global Information Assurance Certification Paper

Using SNORT® for intrusion detection in MODBUS TCP/IP communications

GIAC (GCIA) Gold Certification

Author: Javier Jiménez Díaz, javier.jimenez@coit.es
Advisor: Robert Vandenbrink
Accepted: December 7th, 2011

Abstract

The regular IP traffic analysis has been well studied from an intrusion detection point of view in the field of the Information Security. Nevertheless the convergence process among conventional IT (networks and services) and industrial communication technologies is creating new environments with purpose built networks and new security requirements. On this scenario MODBUS TCP/IP comes up as a 'de facto' communication standard. For those networks there are commercial products that can analyze traffic, detect intrusions and even take actions. However most of them have their own hardware and software platforms and are not always as transparent and flexible as could be expected. Additionally their cost can even made them not suitable for all deployments. This paper proposes a method to approach the problem in a cost effective manner, based on the use of well known open source tools and a methodology to develop the rules to detect intrusions. As a result the IT resources of an organization (employees, hardware and software) can also take care of the company industrial network security without high additional cost in equipment or training time.

1. Introduction

Not long ago, analog and purpose built communications systems use to be prevalent technologies on industrial plants. It wasn't common to find either interoperability or compatibility among them. In the 70s communication Networking began to be used in Direct Digital Control (Berge Jonas, 2004).

Different technologies were developed during the following period and was in 1979 when MODBUS were first introduced in the market. Modicon developed it with the idea of a protocol to communicate register's content from a PLC (Programmable Logic Controller)[1] to another station, which could be another PLC or a computer. It was first designed to use serial communications which was the common practice on industrial environments (Caro Richard, 2003).

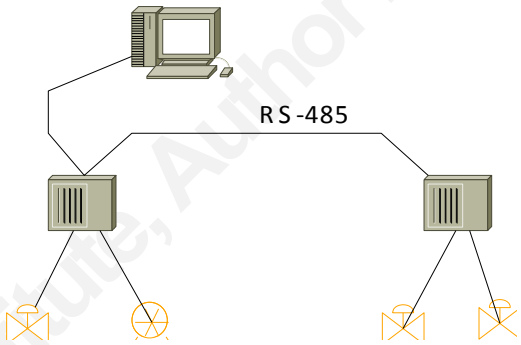


Figure 1: Serial MODBUS Schema

From that time on a lot of other industrial communication systems have been developed and deployed. So today we can find Hart instruments communicated with handhelds communicators or PC (with specific Software), Profibus DP networks or Foundation Fieldbus architectures for control (Thompson Lawrence, 2008).

During the same period especially from mid 90's on, the use of IT (Information Technology) has grown all around the globe and also inside the companies. Thus two major processes have affected the industrial communication environment: the standardization and the convergence.

Even though we can find many different options and products to communicate devices like computers, PDAs (personal digital assistant) or purpose build devices, most of them are used

[1] (*Programmable logic controller*, 2011)

on small and specific fields. Only a subset of communication systems is widely adopted, with Ethernet and the TCP/IP model as the big ones. On general purpose computer systems we can also reduce the Operating System spectrum to Microsoft Windows and Unix / Linux based systems. On this scenario with official and de facto standards, commercial hardware and software becomes significantly cheaper than dedicated systems. Because of that a vast majority of vendors tend to use COTS (Commercial of the shelf) systems which are the same as the regular IT infrastructure. That makes possible to feed the Enterprise Data Servers with process data (real time and consolidated ones) with computers included on the control of critical infrastructures. That is a convergence process among IT systems and industrial information Systems.

Because of how suitable it is and in order to communicate easily with the corporate networks, the MODBUS Organization released an open MODBUS TCP/IP specification in 1999 ("Modbus messaging on," 2006). That keeps MODBUS as the most common protocol for use in SCADA systems directly with PLCs (Caro Richard, 2003). And MODBUS Organization claims it to be the most widely used network protocol in the industrial manufacturing environment ("Modbus faq," 2011).

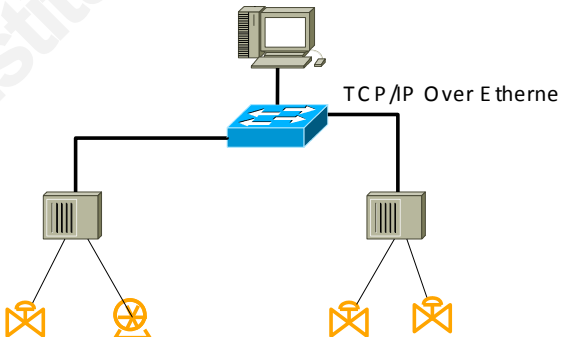


Figure 2

Today MODBUS protocols are used in different industrial system widely, according to a study MODBUS was the world leader protocol in terms on units shipped in 2004 ("Press release: Modbus," 2005). MODBUS protocol is involved on the control of lots of different kind of industries, including critical infrastructures. Thus we can find it on high tech material production (Lutovsky Adrienne & Bramhall Danetta, 2011), beverage brewing ("Case study modbus,"), Petrochemical industry ("Petrochemical plant replaces,") or wastewater ("Matrikon's opc

server,"). It makes it a potential vector of attack for somebody trying to compromise industrial systems. So it makes sense to focus on monitor such a common protocol that is commonly implemented over TCP/IP.

2. MODBUS Protocol Analysis

MODBUS was originally designed to work over a serial link like RS-232 or RS-485. However it has become an application layer messaging protocol for client/server communication. Because of its request/reply architecture is a natural step to embed it on IP packets, doing it available to different devices connected to different networks.

2.1. MODBUS Application Protocol

MODBUS is an application layer protocol, layer 7 of the OSI model ("Modbus application protocol," 2006). Thus it can relay over a set of different protocols on the layer below. The description is as represented on Figure 3.

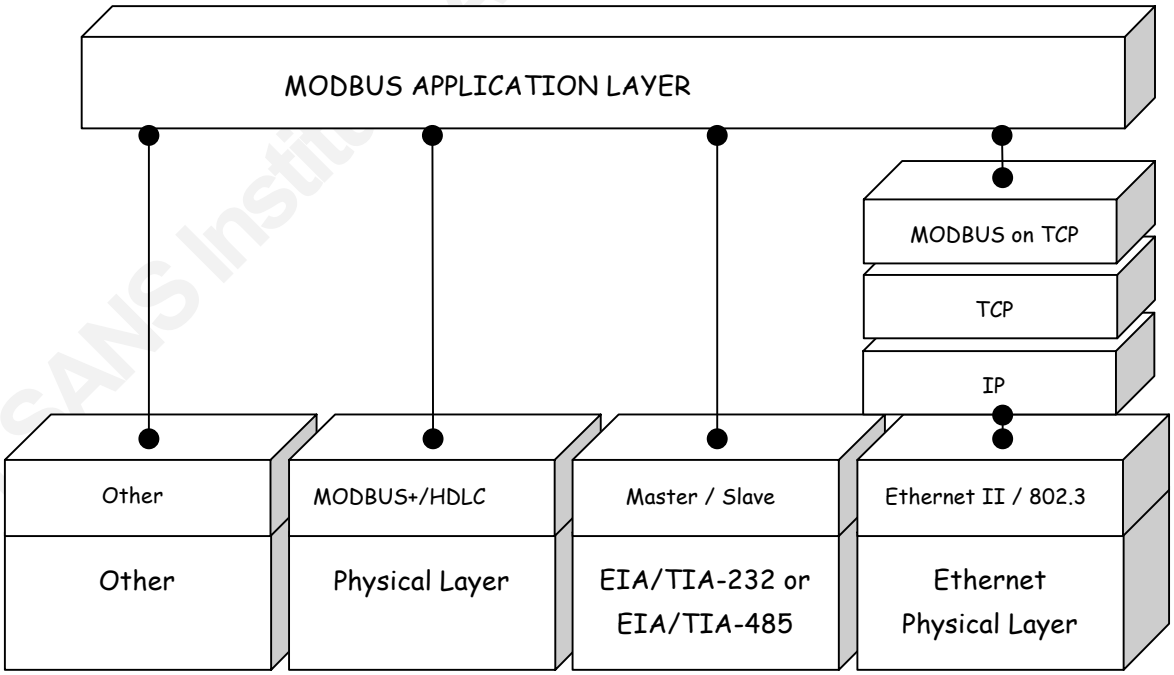


Figure 3: MODBUS Communication Stack as in protocol specification ("Modbus messaging on," 2006)

The MODBUS defines a simple PDU that might include additional ADUs depending on the layer that relies on.

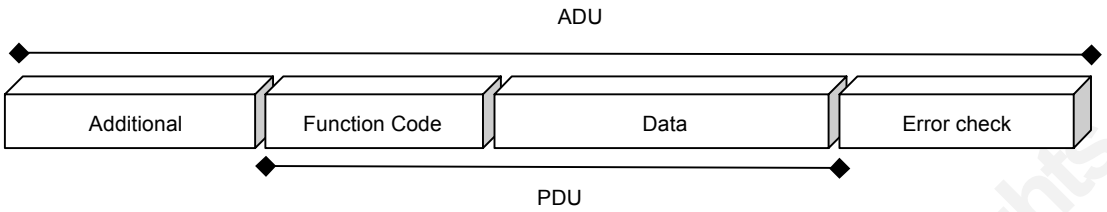


Figure 4: MODBUS Frame (General)

As we see on Figure 4 there always will be a function code and a data field. The function code is a one byte valid values are from 1 to 255. Function code 0 is not valid.

The data field length depends on the function code and could be nonexistent (zero length).

If no error occurs the server response contains the function code (same as in the request) and de data requested. If an error related to the MODBUS function requested occurs, the function code field contains an exception function code. In that case data field contains an exception code (function code with most significant bit set to one).

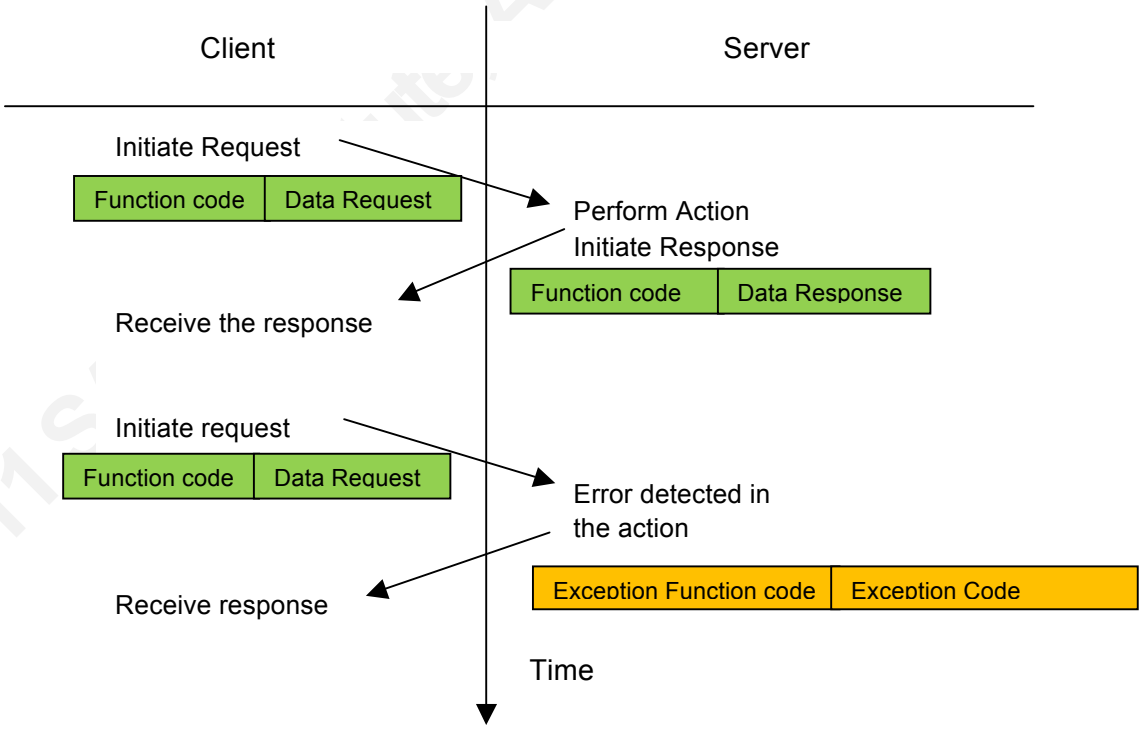


Figure 5: MODBUS Transactions

Since, as was explained on the introduction, the protocol was designed to lay over serial communication protocols the maximum PDU length is a legacy constraint related to that (RS485 maximum ADU is 256 bytes).

Serial communications include on the ADU the server address (1byte) plus a CRC (2 bytes) at the end. Then the maximum PDU is 253 bytes.

2.2. MODBUS Data Model

The MODBUS data model is a quite simple set of serial tables all of them with different characteristics. The primary tables are:

Table	Object type	Type of	Characteristics / Descriptions
Discrete Input	Single bit	Read-Only	Type of data retrieved from an I/O system. Typically represents digital (voltage or contact) inputs.
Coils	Single Bit	Read-Write	Type of data that can be alterable by an application program (inputs, outputs or intermediate variables)
Input Registers	16-bit word	Read-Only	It can be provide by an I/O system. Typically analog inputs.
Holding Registers	16-bit word	Read-Write	This type of data can be alterable by an application program.

Table 1: MODBUS data

It is accepted and very common to refer all four tables as overlaying one another.

For the primary tables the protocol allows individual selection of 65536 data items. However the data is regarded as a real physical memory structure (that is because old systems use to have that kind of structure) it does not mean that the data reference match the memory address. The only requirement on the MODBUS server is to link the data reference to the physical address.[2]

Unsigned integer indices (starting at zero) are used in MODBUS Functions.

[2] The way this link is done depends on the vendor. Two different examples can be consulted at the MODBUS Application Protocol Specification ("Modbus messaging on," 2006).

2.3. MODBUS addressing model

The MODBUS Data Model was developed with the original Modicon PLCs in mind. Then only four tables of 1000 each of them where allocated. As a result even today we can find software and references that allows us just to address that set of address instead of the 0-65535 standard defined.

On that old addressing model the thing looks as follow:

Coil/Register Numbers	Data Addresses	Type	Table Name
1-9999	0000 to 270E	Read-Write	Discrete Output Coils
10001-19999	0000 to 270E	Read-Only	Discrete Input Contacts
30001-39999	0000 to 270E	Read-Only	Analog Input Registers
40001-49999	0000 to 270E	Read-Write	Analog Output Holding

Table 2: MODBUS Address, old style

The table includes the numbers that are actually referenced on well known software or design documents with the real addresses that, as stated before, are artificially limited to 1000 for each table.

2.4. MODBUS over TCP/IP

The MODBUS general definition fits perfectly on the TCP/IP communication model. The communication becomes a client-server communication with four types of messages: Request, Confirmation, Indication and Response.

Where:

- MODBUS Request: Message sent on the network by the client to initiate a transaction.
- MODBUS Indication: Request message received on the Server side.
- MODBUS Response: Response message sent by the Server.
- MODBUS Confirmation: Response message received on the client side.

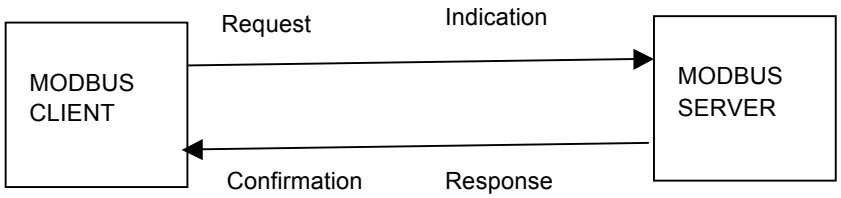


Figure 6: Client-Server communication model

The MODBUS TCP/IP specifications define what actually can be considered an application protocols that relays over the layer 4 of the OSI model. Thus the layer diagram looks like Table 3

7	APPLICATION	MODBUS TCP/IP
6	PRESENTATION	
5	SESSION	
4	TRANSPORT	TCP
3	NETWORK	IP
2	DATA LINK	Ethernet (IEEE 802.3)
1	PHYSICAL	Wi-Fi (IEEE 802.11a/b/g) Others

Table 3: MODBUS TCP/IP on the layer stack

The general frame described on 2.1 are now encapsulated on the TCP/IP model removing the error checking field (mainly oriented to serial protocols that does not have error checking integrated).

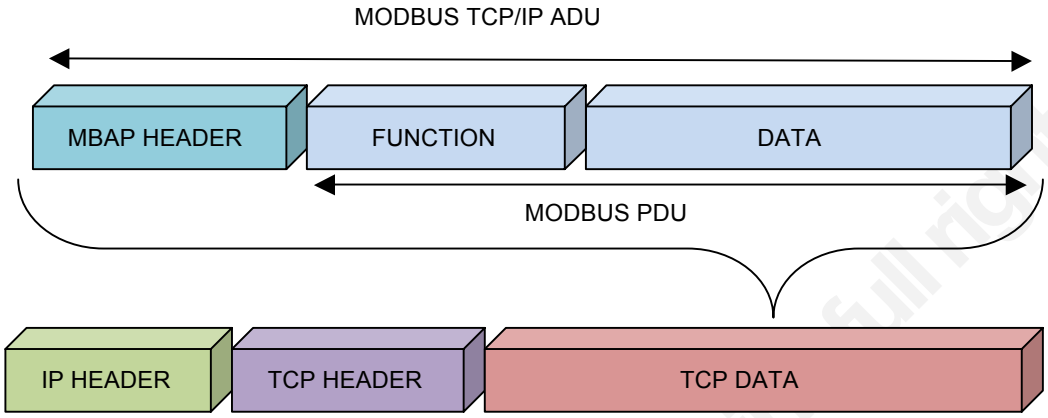


Figure 7: MODBUS ADU and PDU over TCP/IP stack

The MBAP header is the MODBUS Application protocol header that is used to identify the MODBUS Application Data Unit. Provides some differences compared to the MODBUS RTU application as that MBAP header carries information to allow the recipient to identify message limits.

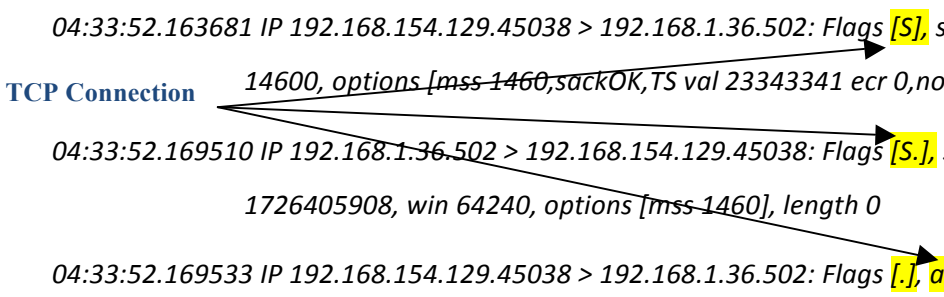
Fields	Length	Description -	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)
Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

Table 4: MBAP Header fields ("Modbus messaging on," 2006

All this structure can be seen on the following capture of a read input register request and response:

Connection establishment:

As a regular TCP connection, it is done with a SYN → SYN ACK → ACK Schema.



Request:

Once the connection has been established the client asks for the data.

04:33:52.178858 IP 192.168.154.129.45038 > 192.168.1.36.502: Flags [P.], seq 1:13, ack 1, win 14600, length 12

Response (and acknowledge):

The server serves the data and the client acknowledges it.

04:33:52.186899 IP 192.168.1.36.502 > 192.168.154.129.45038: Flags [.], ack 13, win 64240, length 0

04:33:52.186953 IP 192.168.1.36.502 > 192.168.154.129.45038: Flags [P.], seq 1:12, ack 13, win 64240, length 11

04:33:52.186977 IP 192.168.154.129.45038 > 192.168.1.36.502: Flags [.], ack 12, win 14589, length 0

Communication finish:

Once the data has been delivered the server finish the communication with the client. The software used for the test ends the connection with a RESET from the client side after a FIN from the server which also tried to FINISH connections on other two ports preceding the connection one (it seems to be to avoid left connections hanging).

04:33:52.187540 IP 192.168.154.129.45038 > 192.168.1.36.502: Flags [F.], seq 13, ack 12, win 14589, length 0

04:33:52.187732 IP 192.168.1.36.502 > 192.168.154.129.45038: Flags [.], ack 14, win 64239, length 0

04:34:02.854993 IP 192.168.1.36.502 > 192.168.154.129.45036: Flags [FP.], seq 684277905, ack 2063228082, win 64239, length 0

04:34:02.855087 IP 192.168.154.129.45036 > 192.168.1.36.502: Flags [R], seq 2063228082, win 0, length 0

04:34:23.197425 IP 192.168.1.36.502 > 192.168.154.129.45037: Flags [FP.], seq 14618640, ack 3946956188, win 64239, length 0

04:34:23.197521 IP 192.168.154.129.45037 > 192.168.1.36.502: Flags [R], seq 3946956188, win 0, length 0

3. Security Problems

3.1. MODBUS security profile

The MODBUS protocol was not initially designed with cybersecurity in mind hence it lacks the mechanism to avoid the classical information security threats. The protocol does not include a way of ciphering the traffic, check the integrity of messages, and authenticate client and server[3].

Therefore an attacker could compromise:

- Confidentiality since the traffic goes clear over the network.
- Integrity because there is no way of know if the original message has been changed throughout the communication links
- Availability due to de possibility of use crafted packets or reusing legitimated ones to consume either network or client or server resources (e.g. resetting tcp connections).

The theory can then be taken to the field and state that once an attacker can inject traffic to the network it could send commands to servers or false information to clients as well as require information or (depending on the attacker position) sniff it.

3.2. Attack Scenarios

There are different types of networks depending on the control system and the user's information necessities. However two different situations can represent most of them with regards to cybersecurity.

[3] The Access Control Module checks only IP addresses ("Modbus messaging on," 2006).

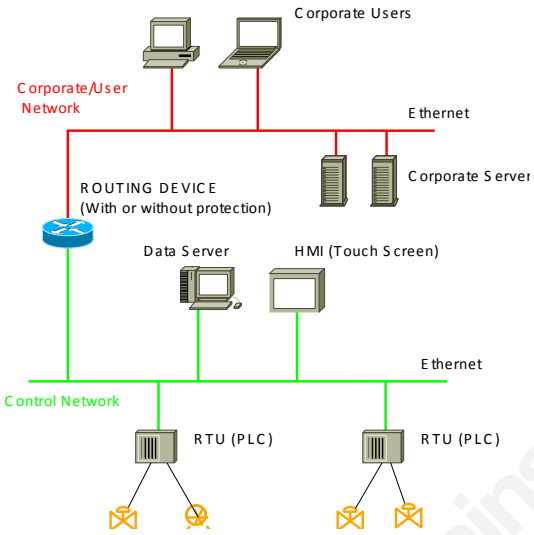


Figure 8: Single Network scenario

On Figure 8 is shown a plain network scenario. On that scenario an attacker that could send packets to the control network either from inside or outside could reset connection, send commands to the slaves (RTUs) or cheat masters (HMI) with fake data pretending to be the PLCs. Depending on the network and device configuration could also sniff traffic and retrieve information about memory addresses or common operations performed on the system.

By contrast the other common way of organizing the network is having different network segments. On that case there are two different control related networks[4].

PCN - Process Control Network: Contains the Servers, SCADA and HMI components of the control System. Operators and plant personnel usually interact directly with that network. Additionally control data used on the corporate network is retrieved from this network as well.

CSN - Control Systems Network: Contains the automation equipment and also the DCS or SCADA servers are connected. It carries the traffic from the field automation devices (RTUs) to the servers that put those data on the PCN.

On the scenario described above the data are served to the different SCADA/DCS machine not using MODBUS but a different protocol (e.g. OPC). Thus injecting traffic from outside becomes more difficult since usually the devices do not route traffic. Then in order to

[4] Based on the definition described in reference ("Security concept pcs 7.", 2004).

exploit the protocols weakness an attacker needs to take control of one of those machines or physically access the network devices.

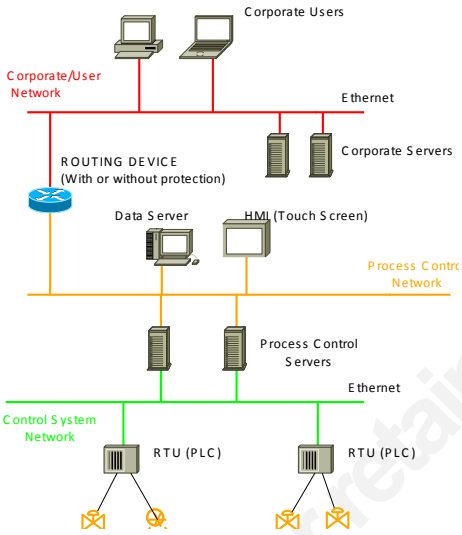


Figure 9: Separate networks scenario

Obviously the scenarios can be more complex including other kind of routing protocols more than two different levels or hybrid configurations (for examples PLCs on both the CPN and the PCN). But most of the attack scenarios can be represented on those two.

4. Remediation

The first step to be protected against the problems related with the absence of cybersecurity capabilities on MODBUS TCP IP, is the use of General IT knowledge and best practices to secure the network. Those recommendations came from vendors, industry, standard and regulations[5].

In addition to a good policy that might include regular assessments and audits, use of authorization and authentication, the most useful thing to be done is have a good network design.

The network design recommended by most vendors, standards and regulations use the defense in depth concept. Such approach reflects the idea of designing complementary and highly redundant security controls that establish multiple layers of protection to safeguard assets.

[5] E.g.: (DesRuisseaux Dan, 2009)

With such strategy the failure of a single control or barrier should not result in the compromise of any of the systems to protect ("Cyber security programs," 2010).

Hence a good network design would include different layers separated by firewalls of different vendors with redundant rules to avoid undesired traffic to achieve our control systems. To be useful and complete a good and restrictive traffic and access rule policy is the necessary. In the MODBUS TCP/IP case not allowing traffic to TCP 502 port (standard MODBUS TCP/IP port) must be forbidden to anybody but the legitimate clients. Depending on the products that are integrated on a system or set of interconnected systems the final user will have enough information to do a fine tuning of those rules.

Another common recommendation is put in place compensating controls when no digital protection is available (or even if they are). They include physical access limitation which are quite common on industrial plants (e.g. if you cannot put a RADIUS system keep everybody physically away from the switches).

Isolating the systems is also a solution to be considered. Modern control systems have the capabilities of communicating with regular IT systems allowing the company to access them for other purposes beyond control (e.g.: Near real time analysis and supervision of production indicators by the managers). Unfortunately accepting the isolation of the system is, in fact, comparable to a defeat on the information availability battlefield. So it does make no difference at all whether the system is isolated on purpose or by an attacker (obviously assuming that no other aspect of the system is disrupted). Although isolating the system actually mitigates some threats, it is not really possible to isolate any modern system completely. Somehow most of them need to load data (e.g.: software updates of operating system, control software, control program, etc.), and data will be in any case needed to be downloaded (e.g.: analysis of malfunction, program modifications, etc.). So the system is clearly more protected from the world but cannot be considered completely secure. As a famous example of that, the Stuxnet Attack was designed to attack devices never directly connected to any insecure network (PLCs). It uses Field PGs[6] devices that were just temporary connected to LAN networks and never to internet or public access networks (Falliere Nicolas, Murchu Liam & Chien Eric, 2011) as an attack vector. Even

[6] Rugged Windows Laptops used to program PLCs.

though it was not an attack to any MODBUS system, nevertheless it shows how impossible is to keep an isolating system as a security measure.

4.1. Specialized products

It is clear then that there is a field for improvement with regard to MODBUS security. Because of that there are specialized products on the market that allow the final user to protect and monitor the MODBUS TCP/IP communications.

The MODBUS TCP/IP Specialized IPS Tofino™ allows the user to define rules to just allow the legitimate traffic to go over the inline device. It has a centralized console to define rules and apply them for different devices and analyze the logs and alerts ("Loadable security modules," 2011).

Other vendors offer a solution based on traffic flow control, which means allow only unidirectional traffic. This is a concept included security regulations as (Regulatory Guide 5.71, 2010). Following this concept a MODBUS replication or a one-way sending solution allow the data to be available on a server out of the control Network and easily accessible to users on the corporate network ("Modbus - waterfall," 2010).

As a complement to that solutions there are other ways to approach the system that could try to secure or improve the protocol itself. Unfortunately the standards products on the market and the systems that are already installed (including legacy systems) neither have that capabilities nor the option to implement them easily. A good example of that is on (Nai Fovino Igor, Carcano Andrea, Masera Marcelo & Trombetta Alberto, 2009).

5. Intrusion Detection on the control network

Whereas the solutions related above mitigate the risk that any control network faces, none of them can secure 100% the assets on the network and thus the control itself. Intrusions and abnormal behaviors on the network must be detected and monitored.

As a core tool for that task, SNORT® is a good option since it is the open source 'de facto' standard. However is an IT world tool which means that most control engineers are not quite familiar with that. By contrast it is very adaptable to the user necessities whatever they are and there are a wide range of already published rules to protect against well known vulnerabilities. Additionally rules for protecting MODBUS TCP/IP are also disclosed by

organizations like Digital Bond's and can be included on a customized configuration ("Quickdraw scada ids," 2011) . Furthermore also companies and IT security Specialists that deliver services for SNORT® are easily available on the market.

5.1. Traffic characteristics

Because of the industrial control system network's nature, the traffic is mostly static. This means that the network and the regular communication don't change frequently. If a PLC is controlling on a stable situation, changes on the operations that are allowed does not vary overnight. That includes the communication parameters like operations, addresses, transmitters and receivers.

In addition to that, the traffic on a MODBUS TCP/IP segment (e.g. the link to the slave) is well known since it should match the standard. The standard is also well documented and accessible.

A general statement that can be assume as well is that when an RTU or other device is monitored or controlled using MODBUS TCP/IP on such kind of link there is almost no other traffic. If a PLC is controlled or monitored using MODBUS, it is not often also running an ftp or a NetBIOS communication as well.

By contrast to those advantages (from a security point of view), an interruption on control traffic can be catastrophic, so then protection systems on the boundaries prompt to be more permissive than restrictive to avoid drop traffic by mistake which could lead to an undesirable situation.

5.2. How to design the rules

The first step to define the rules is to make an approach of what is consider being normal traffic. The parameters to define it (on a first approach) are:

- Network and transport layer (TCP/IP)
  - Origin
  - Destination
  - Protocol
  - Port

- Application layer (MODBUS)
  - MODBUS Header (MBAP Header).
  - Function code and subcodes
  - Data

That way a first rule version based on only one or two parameters can be set and then complete adding options to the same rule or with a set of rules schema.

For instance if only discrete input reading operations are allowed on a network (that is monitoring only) a rule like follows would be set for start up:

```
alert tcp $CLIENT_MODBUS_NET any -> $SERVER_MODBUS_NET 502\
(content:!"|02|";offset:7;depth:1; flow:established, to_server;\
msg:"MODBUS- not allowed funct attempt"; sid:1000001;rev:0;priority:5;)
```

Function code

Byte in eighth position (offset 7) is the byte that contents the function code, so the rule will check from the modbus networks of clients to the modbus networks of servers at Modbus port the function code. It will alerts of any non read discrete input request detected. On the following section more parameters will be include on one single rule.

5.3. Designing rules

Following is the capture of a TCP Connection establish and a Read Coils request (flags market on the packets).

```
12:33:12.282228 IP 192.168.154.128.54919 > 192.168.1.36.502: Flags [S], seq 1466713865, win
    14600, options [mss 1460,sackOK,TS val 21116036 ecr 0,nop,wscale 4], length 0
12:33:12.283951 IP 192.168.1.36.502 > 192.168.154.128.54919: Flags [S.], seq 1551594437, ack
    1466713866, win 64240, options [mss 1460], length 0
12:33:12.284014 IP 192.168.154.128.54919 > 192.168.1.36.502: Flags [.], ack 1, win 14600, length 0
12:33:12.284584 IP 192.168.154.128.54919 > 192.168.1.36.502: Flags [P.], seq 1:13, ack 1, win 14600,
    length 12
```

The last packet with the request from the client is showed below.

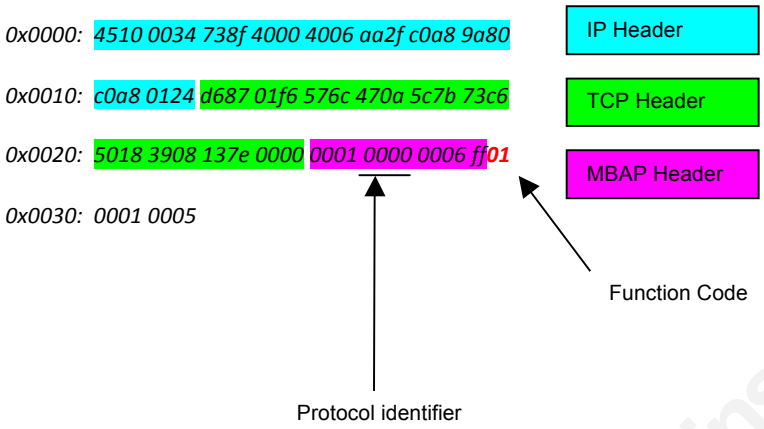


Figure 10: Request coils packet analysis

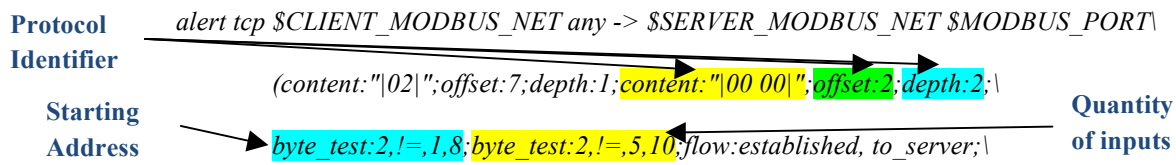
The previous rule will alert every time the 8th byte (offset 7) of the TCP payload is different of 0x02 which is the function code. Then it would alert even with not MODBUS TCP packets to port 502.

That is because the rule is focus on the function code (marked red on Figure 10). The function code of the MBAP Header will always be 0000 in case a MODBUS communication is going on so it can be include on the rule to avoid false positives.

Additionally the unit identifier (previous byte to the function) can be used to characterize the traffic more precisely, even though is only used for intra-system routing (the field is always included on the header).

When it comes to tune the rule, the parameters to be taken in account are the parameters related to the functions. The parameters to be controlled on the writing and reading functions are the addresses. They are the next 4 bytes on the most common reading and writing operations.

This kind of operations is done using an offset count schema. Depending on the software used the client asks for all the memory addresses or for the different burst. The rules should be done on a request basis. As an example the following rule:



Javier Jiménez Díaz, javier.jimenez@coit.es

msg:"MODBUS- not expected address requested";sid:1000001;rev:0;priority:5;)

The rule will alert on an event of a coil reading request with an offset and count different of the expected (1 and 5).

Depending on the situation of the network it can be use a "pass all" approach or an "alert all" approach. Since the traffic of control systems is usually quite static (meaning than once characterized does not change frequently) an "alert all" approach allows to monitor the traffic easier and with less rules. On the other hand is more difficult to tune and false positives are likely to happened. Coming back to RTU devices, usually they don't have other traffic than the monitor or control traffic so then an "alert all" schema is preferable to be used.

Below there is an example of a set of two rules that will alert in case of an event of a MODBUS request to the server network, on port 502, different to a read coils starting in 1 and asking for 5 coils.

```
var CLIENT_MODBUS_NET 192.168.154.0/24
var SERVER_MODBUS_NET 192.168.1.0/24
var MODBUS_PORT 502
output log_tcpdump: tcpdump.log

preprocessor stream5_global: track_tcp yes
preprocessor stream5_tcp
preprocessor stream5_udp: ignore_any_rules


pass tcp $CLIENT_MODBUS_NET any -> $SERVER_MODBUS_NET $MODBUS_PORT\
    (content:"|01|";offset:7;depth:1;byte_test:2,=,1,8; content:"|00 00|";\
    offset:2;depth:2;byte_test:2,=,5,10;\
    flow:established, to_server;sid:1000001;rev:0;priority:5;)
```

```
alert tcp any any -> $SERVER_MODBUS_NET $MODBUS_PORT (content:"|00 00|";\
    offset:2;depth:2;flow:established, to_server;\
    msg:"MODBUS- not allowed funct attempt"; sid:1000002;)
```

Figure 11 shows the fields that are check on a binary packet capture.

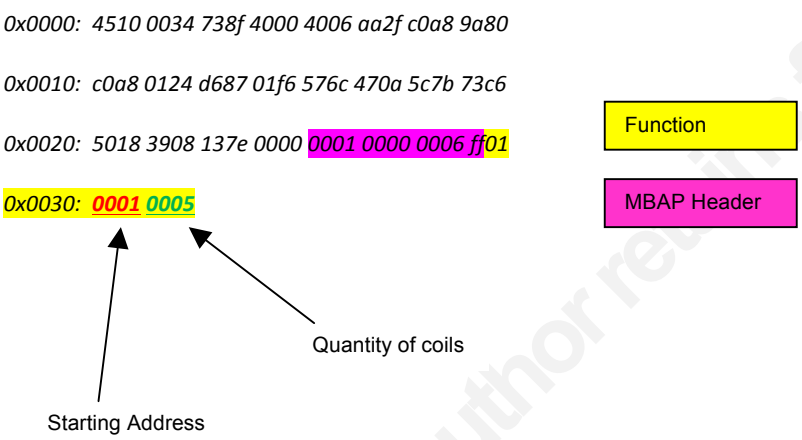


Figure 11: Request coils packet analysis (cont)

The other side of the communication is from the server to the client. The client application can be cheated by an attacker by sending responses to non requested operations. That kind of rule would monitor the client on a similar way as done with the server. On a correct response the function code is the same on request and response. Hence similar rules apply for both. The following rule would alert in case of a writing coil operation response.

```
alert tcp $SERVER_MODBUS_NET 502 -> $CLIENT_MODBUS_NET any  \
    (content:"|05|";offset:7;depth:1; content:"|00 00|";offset:2;depth:2;\
    flow:established,from_server;\
    msg:"MODBUS- not allowed funct response"; sid:1000001;rev:0;priority:5;)
```

Since the error code in case of an exception response is the function code with the most significant bit set to one those can also be monitored to avoid not allowed responses. To create a rule for that it can be use a higher function that 43 (the higher one according to protocol specification) or the most significant bit to one, that is value higher than decimal 128 (hex 0x80)

```
alert tcp $SERVER_MODBUS_NET 502 -> $CLIENT_MODBUS_NET any \
    (content:!"|82|";offset:7;depth:1; content:"|00 00|";offset:2;depth:2;\
    byte_test:1,>,128,7;flow:established, from_server;\
    msg:"MODBUS- suspicious error code"\
    sid:1000001;rev:0;priority:5;)
```

The rule above alerts in case of an exception response (function code higher than 127) different from the one allowed (0x82 for read discrete inputs function).

There are also codes and subcodes that are reserved on the protocol specification, thus checking for them would also be a good practice since they should not be present on a regular communication. As an example:

```
alert tcp $CLIENT_MODBUS_NET any -> $SERVER_MODBUS_NET 502\
    (content:!"|0a|";offset:7;depth:1; content:"|00 00|";offset:2;depth:2;\
    flow:established, to_server;\
    msg:"MODBUS- reserved function code used."\
    sid:1000001;rev:0;priority:5;)
```

6. Conclusions

The use of regular IT networks for industrial control protocols brings new security challenges to be addressed because of the different uses, requirements and restrictions of that kind of systems. In such Scenario MODBUS TCP/IP is a wide spread protocol that came from legacy serial communication systems. With no build in security capabilities protection, monitoring and analysis are necessary to achieve a reasonable security level.

MODBUS TCP/IP is no other than an application protocol relying over regular IP network so then regular tools can be used to analyze and monitor it. Moreover different specialized tools on the market can help to protect the systems using that kind of traffic. However SNORT® as a widely used open source IDS solution is ideal to a customized of the MODBUS TCP/IP traffic.

Using SNORT® almost any MODBUS parameter can be check whereas using only some of them is easy to do a general characterization of the traffic. According to specification MODBUS has enough parameters to be properly characterized. Depending on the systems to

monitor and the traffic to analyze an "alert all" or "pass all" default approach will suit better, although an "alert all" schema is likely to be used on simple links like PLCs connections. Because of the simple and stable nature of the traffic a small set of rules can check for most general deviations on expected traffic.

7. References

Berge Jonas. (2004). *Fieldbuses for process control: Engineering, operation, and maintenance.* (1 ed.). Research Triangle Park, NC, USA: ISA - The Instrumentation, Systems and Automation Society.

Caro Richard, H. (2003*). Automation network selection.* (1 ed.). Research Triangle Park, NC, USA: ISA - The Instrumentation, Systems and Automation Society.

Thompson Lawrence, M. (2008). *Industrial data communications.* (4 ed.). Research Triangle Park, NC, USA: ISA - The Instrumentation, Systems and Automation Society.

(2010). *Cyber security programs for nuclear facilities (Regulatory Guide 5.71).* Retrieved from U.S. Nuclear Regulatory Commission website: http://www.nrc.gov/reading-rm/adams.html under Accession No. ML090340159

DesRuisseaux Dan. (2009).*Designing a security policy to protect your automation solution.* Retrieved from Schneider Electric website: http://www2.schneider-electric.com/documents/support/white-papers/Security-Policy-20091020.pdf

Tanenbaum, A. S. (2003).*Redes de computadoras*. (4 ed.). Mexico: Pearson Educación.

Roesch Martin. , Green Chris , , & Snort Team, (2009).*Snort users manual 2.8.5*. Retrieved from Sourcefire, Inc. website: http://www.snort.org/assets/125/snort_manual-2_8_5_1.pdf

Modbus faq. (2011, November). Retrieved from http://www.modbus.org/faq.php

(2006).*Modbus application protocol specification v1.1b.* Retrieved from Modbus Organization, Inc. website: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf

(2006).*Modbus messaging on tcp/ ip implementat ion guide v1.0b*. Retrieved from Modbus Organization, Inc. website: http://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf

Simply modbus - frequently asked questions. (2011). Retrieved from http://www.simplymodbus.ca/faq.htm

Programmable logic controller. In (2011). *Wikipedia The Free encyclopedia* Retrieved from http://en.wikipedia.org/wiki/Programmable_logic_controller

Modbus - waterfall security solutions. (2010). Retrieved from http://www.waterfallsecurity.com/MODBUS/

Nai Fovino Igor. , Carcano Andrea, , Masera Marcelo, , & Trombetta Alberto, (2009).*Design and implementation of a secure modbus protocol*. In Charles Palmer, Sujeet Shenoi (Eds.),*Crticial Infrastructure Protection III - Third Annual IFIP WG 11.10 International Conference on Critical Infrastructure Protection*(pp. 83-96). Retrieved from http://books.google.es/books?id=J3xDGltyvlEC&lpg=PA83&dq=Design%20and%20Implementation%20of%20a%20Secure%20Modbus%20Protocol&hl=es&pg=PA83#v=onepage&q=Design%20and%20Implementation%20of%20a%20Secure%20Modbus%20Protocol&f=false

Falliere Nicolas. , Murchu Liam, O., & Chien Eric, (2011).*W32.stuxnet dossier*. Retrieved from website: http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

(2004).*Security concept pcs 7 and wincc - basic document*(A5E02128732-01). Retrieved from Siemens AG Automation and Drives website: http://www.industry.siemens.nl/automation/nl/nl/industriele-automatisering/PCS7/Documents/Whitepaper_IT_security_PCS7.pdf

Press release: Modbus tcp is world leader in new arc study. (2005, May 12). Retrieved from http://www.modbus.org/docs/Modbus_ARC_studyMay2005.pdf

Lutovsky Adrienne. , & Bramhall Danetta, (2011). Prosoft technology gives nasa a boost.*Modbus Organization Newsletter*, (Spring 2011), 3-4. Retrieved from http://www.modbus.org/docs/MBNewsletter_Spring2011.pdf

(n.d.).*Case study modbus based scada*. Retrieved from Comtrol GmbH website: http://www.comtrol.co.uk/applications/CaseStudy-Modbus_Based_SCADA.pdf

(n.d.).*Petrochemical plant replaces legacy equipment and integrates matrikonopc to connect honeywell tdc3000 with tricon and modbus plc*. Retrieved from Matrikon Inc. website: http://www.matrikonopc.com/downloads/131/casestudies/index.aspx

(n.d.).*Matrikon's opc server for modbus makes connection to 300 rtus possible*. Retrieved from Matrikon Inc. website: http://www.matrikonopc.com/downloads/85/casestudies/index.aspx

Loadable security modules | tofino industrial security solutions. (2011, October 9). Retrieved from http://www.tofinosecurity.com/products/loadable-security-modules

Quickdraw scada ids. (2011, October 12). Retrieved from http://www.digitalbond.com/tools/quickdraw/

MODBUS software

MOD_RSSIM. MODBUS RTU and TCP/IP PLC Simulator (Version 8.20) [Software]. Available from: http://www.plcsimulator.org.

(2011) proconX Pty Ltd: Modpoll MODBUS Master Simulator. [Software]. Available from: http://www.modbusdriver.com/modpoll.html

Raimbault, Stéphane., Doerffel, Tobias., & Forster, Florian (2011) libMODBUS. A MODBUS library for Linux, Mac OS X, FreeBSD, QNX and Win32. [Software]. Available from: http://libmodbus.org/download/ . Jul 2011.