Introduction

Schneider Electric's Protolyzer is a flexible, power tool for converting DNP, Modbus, Fisher ROC/ROC+, Series 5, SEL Smart Meter Protocol, Allen Bradly Control Interface Protocol (AB CIP), IEC 104, SuperFlo, TotalFlow and Vancomm protocol data into human-readable text. It will also do hex/ASCII dumps.

The Protolyzer has built-in support for parsing OASyS swana files, Schneider Electric RTU trap files, and PCAP files. However, by configuring custom delimiters, it can be configured to parse protocol data from a variety of sources.

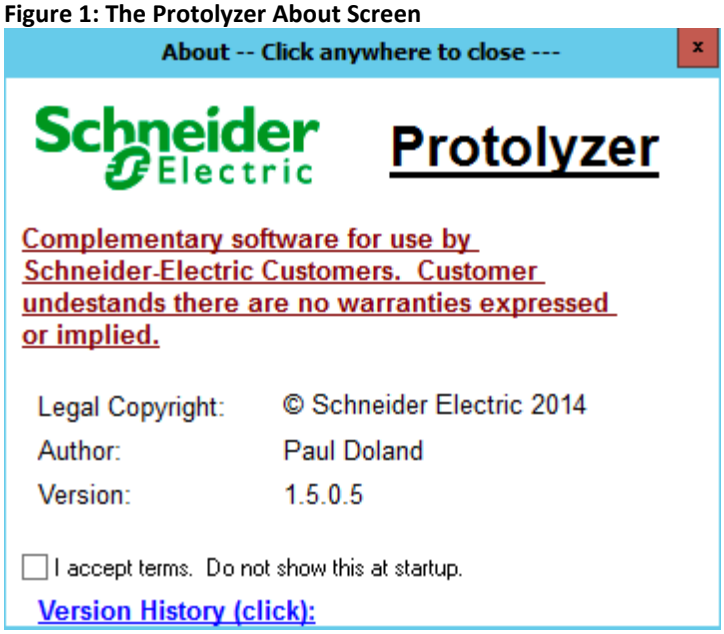
Installation

There is no particular installation procedure necessary, the Protolyzer is a self-contained application file.

You can copy the application file to any directory on your computer, create a shortcut to it on your desktop or in the **Start** menu, or run it from the command line. The file does not store anything in the Windows Registry. The Protolyzer will create a file called `Protolyzer.ini` to store user configuration settings.

Double-click on the **Protolyzer** application file, and click anywhere on the **About** screen to close it. The main screen of the program will open after the **About** screen closes.

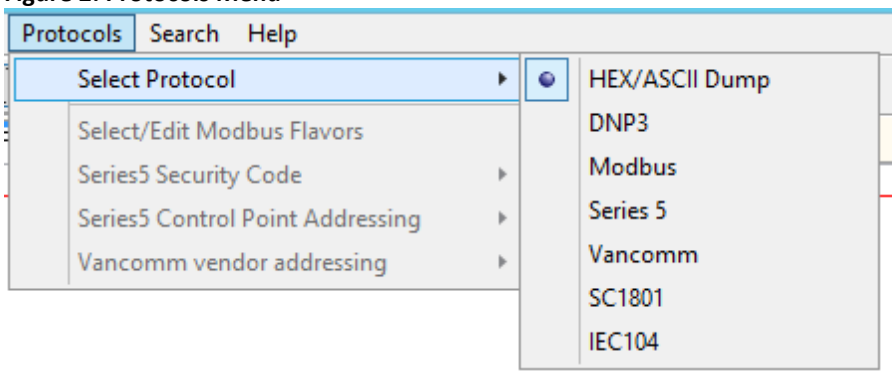
Figure 1: The Protolyzer About Screen



Usage

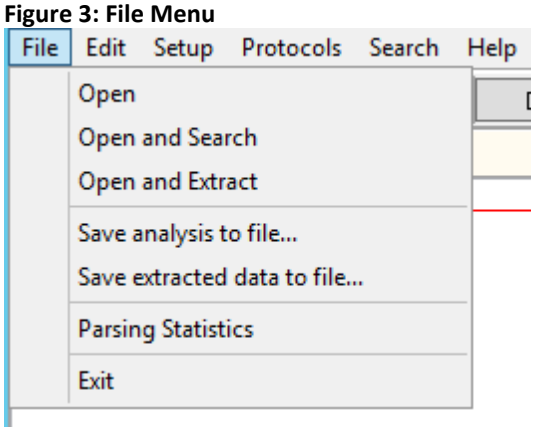
After starting the Protolyzer, you must select a protocol to use. This is done from the **Protocol** menu. The program will remember your most recent protocol selection when you run the program the next time. For some protocols there are options to configure, these options are set from the **Protocol > Select Protocol** menu.

Figure 2: Protocols Menu



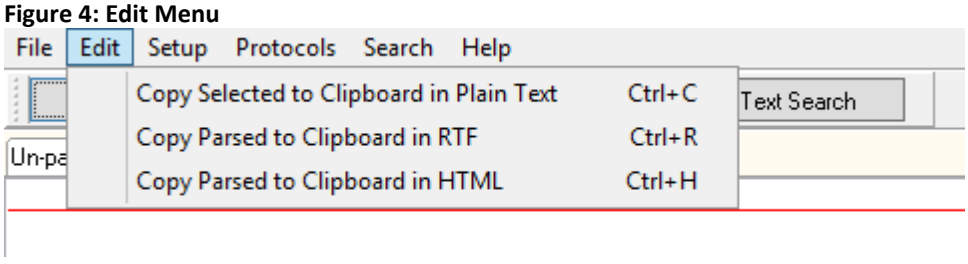
Once you have selected the proper protocol and protocol specific options, you can use **File > Open** to open a text file for parsing. You can also use cut and paste to paste data onto the **Original** tab, and click **Parse**.

Figure 3: File Menu



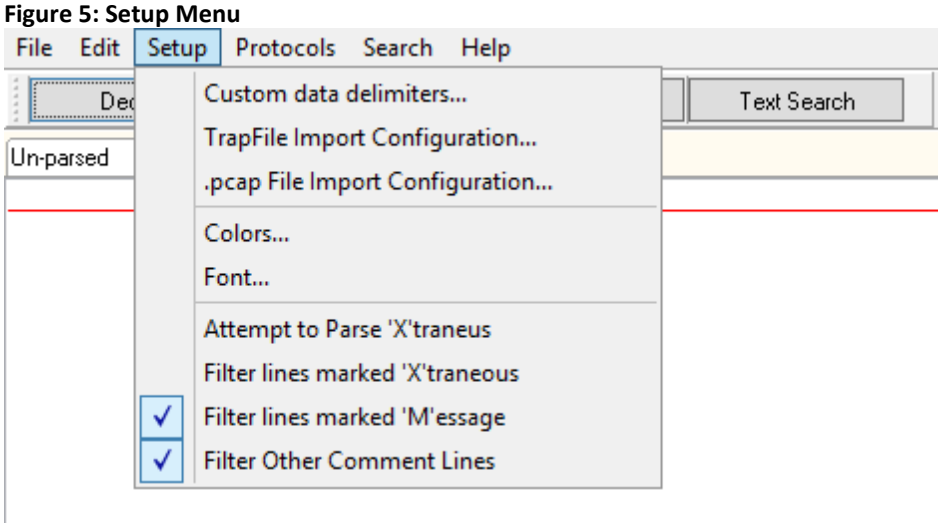
In order to be able to cut and paste parsed DNP data into another program (for example, email), the Protolyzer can put the data onto the clipboard in either RTF or HTML format. Some programs may work better when pasting from RTF format over HTML, or vice-versa. Options to copy selected parsed text to the clipboard are on the **Edit** menu.

Figure 4: Edit Menu



The parse utility uses color coding. By default, the master messages are in blue, remote messages in black and searched for messages are highlighted in yellow. These colors can be changed from the **Setup** menu.

Figure 5: Setup Menu



The Protolyzer will automatically detect RTU trap files based on the contents of the file. If the file has a .pcap extension, it will read the file as a PCAP file. If the file is not a trap or PCAP file, it will expect an ASCII file (most commonly swana files).

NOTE: By using custom delimiters, other data sources may be usable.

Once you have selected the proper protocol, and any protocol specific options, the primary mode of operation is to simply use the File->Open menu option to open a text file for parsing. It will automatically detect RTU trap files. If it is not a trap file, it expects an ASCII file, typically “swana” files. Or other text files if custom delimiters are used. You can also use “cut ‘n paste” to paste data onto the “Original” tab and click the “Parse” button to parse.

The parse utility uses color coding. By default, the master messages are in blue, remote messages in black, and searched for messages are highlighted in yellow. These colors can be changed from the "setup" menu.

In order to be able to cut-and-paste parsed DNP data into some other program, such as into an e-mail, Protolyzer can put the data onto the clipboard in either RTF format, or HTML format. Some programs seem work better when pasting from RTF format over HTML, others vise-versa. Options to copy selected parsed text to the clipboard are on the “Edit” menu.

Protocol Specific Options

Some protocols need specific configuration settings depending on your specific application. These options are selected from the “Protocols” main menu option. These are defined in the sections below.

DNP Specific Options

The DNP parser has a somewhat unique presentation of parsed data compared to other protocols. DNP is unusual in that it has a CRC every 16 bytes, regardless of where in the message that might be. To give the user a choice in how the data bytes verses CRC bytes are presented to the user, there are two options to the parsed data for DNP: “Standard Format” and “Wide Format”. Standard Format is usually preferred as it is more compact. However, Wide Format is more explicit in showing the relationship of the bytes within the DNP packet. DNP parsing format can be selected from the “Protocol” menu.

In Wide Format, CRC bytes are separated from data bytes with a caret (^) symbol.

In Normal Format, the CRC marking differentiates between when a CRC is between objects, and when it is in the middle of an object. If a CRC is located in the middle of an object, it is put in parenthesis. If a CRC comes between objects, the CRC will be shown as the last two bytes of the preceding object, separated by a ^ symbol. Here are two examples:

Example of CRC in the middle of an object (Normal Format):

```
04 00 00(C2 F0)00 00              AI 9 = 0 [Offline Comm_lost]
```

Example of CRC between objects (Normal Format):

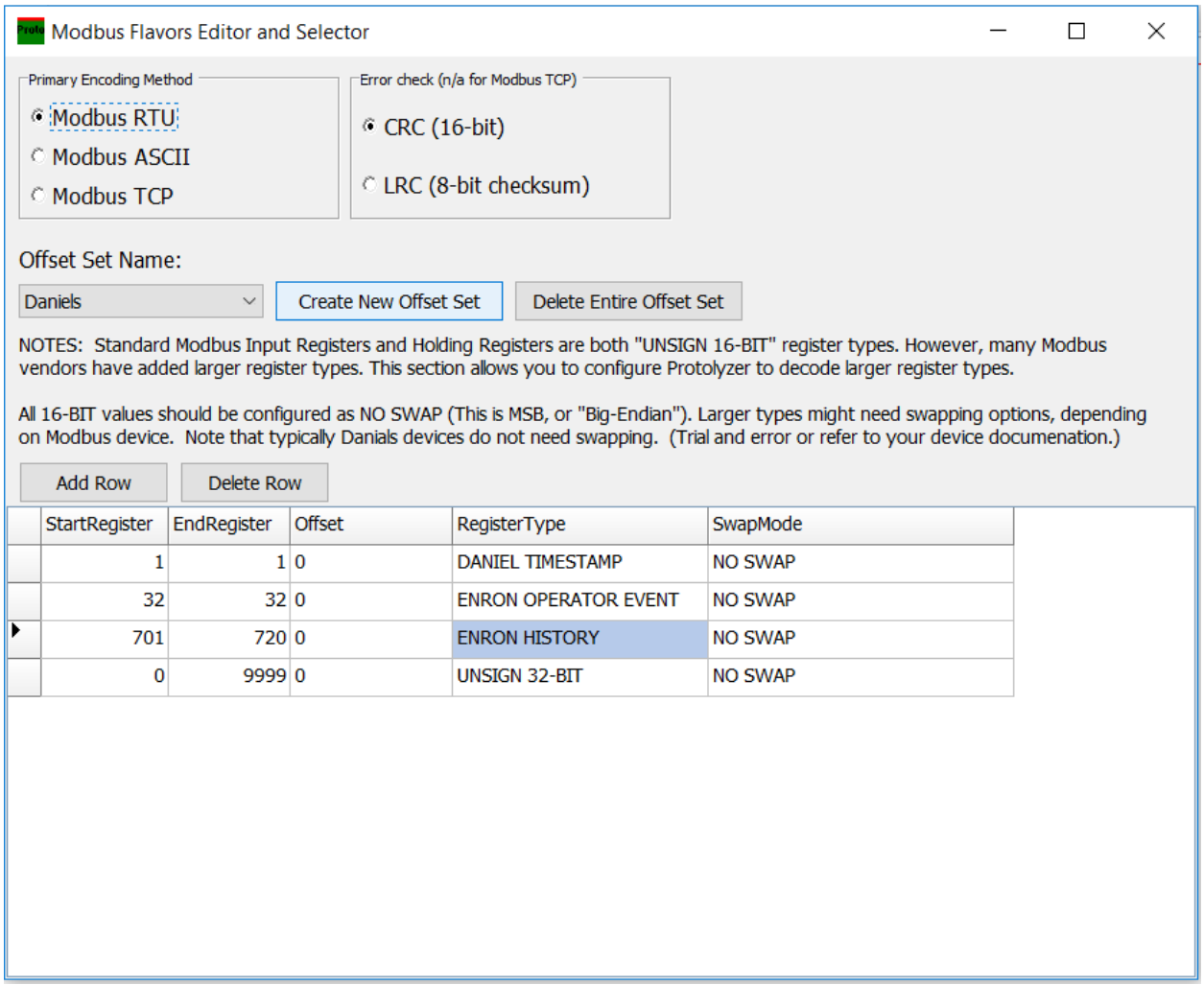
```
04 00 00 00 00^FD 5C              AI 15 = 0 [Offline Comm_lost]
04 00 00 00 00                    AI 16 = 0 [Offline Comm_lost]
```

DNP has two protocol specific options in the “Trap-File Import Configuration” section. There is a checkbox for “Do extra 05/64 parsing (usually not necessary)” and “Split DNP messages based on inter-character timeout (always done on other protocols.)” These options can possibly improve parsing on messages that have a lot of noise. In practice, they are usually not necessary and generally not recommended. These options were added at a customer request and not usually used.

Modbus Specific Options

Modbus has three main modes of operation: Modbus ASCII, Modbus RTU, and Modbus TCP. It also has two possible message validation options: LRC/8 and CRC/16. Additionally, due to the fact that different RTU vendors have used different address space conventions and register types, the ability to configure “Modbus Flavors” is provided.

To configure Modbus operation, first select Modbus as the protocol. Then select “Select/Edit Modbus Flavors” from the Protocols menu. This will bring up the Modbus Flavors configuration page:



On the upper portion of the screen, select the Primary Encoding Methods (Modbus RTU, Modbus ASCII or Modbus TCP) and the error check option (CRC or LRC.) Typically CRC is used for MODBUS RTU, LRC is used for Modbus ASCII, and ModbusTCP has no error check field. Depending on your application, this may be all you need to configure. You may not need to configure offset sets.

The original implementation of Modbus only had 16 bit integer registers. And they were stored MSB (Most Significant Byte first. Also sometimes nicknamed “big-endian”.) Newer Modbus devices often support 32-bit integer and 32-bit float formats. Some even support 64-bit integers and/or floats. However, different brand of RTUs support often use different byte order. Thus there is a swap mode setting.

Also, in the original Modbus implementation, the logical address was generally different than the physical address. For example, the first holding register was given logical address 4001 but was physical address 0. Some RTUs use 40001, some even use 400001. While others just use the same physical address as the logical address. If you wish for Protolyzer to decode displaying a different logical address than physical address, use the offset option. For example, if you wish physical address of 0 to appear as 40001, then you will need to configure an offset set with 40001 as the offset value.

The complete list of register types currently supported are:

COIL/DISCRETE
16-bit signed integer
16-bit unsigned integer
32-bit signed integer
32-bit unsigned integer
64-bit signed integer
64-bit unsigned integer
32-bit float
64-bit float
Individual Bits
Daniel Timestamp
Daniel History
Enron Alarm Event
Enron Operator Event

Note: Configuring register as “individual bits” causes Protolyzer to break registers into individual bits for display. This is useful when registers are being used as status bits.

The Daniel and Enron register types are special register types defined by those vendors. They all happen to be 20 bytes per register.

The supported swap modes are:

NO SWAP (default, standard MSB)
LSB
SWAP BYTE
SWAP WORD
SWAP WORD, BYTE
SWAP DWORD
SWAP DWORD, BYTE
SWAP DWORD, WORD
SWAP DWORD, WORD, BYTE

Note: LSB is equivalent to SWAP DWORD, WORD, BYTE.

So that you can switch quickly between one offset configuration to another offset configuration, named offset sets are available. For example, if you the user sometimes use some Daniels RTUs with certain register definitions, and sometimes use some Enron RTUs with different register definitions, you could create one offset set named “Daniels” and other named “Enron” and switch between them at will.

Also note that the offset definitions are processed in order. Thus you can configure individual exceptions first, and then a general configuration later. In the printscreen above, registers 1, 32, and 701 have special configuration and then a general generic configuration for registers 0 to 99999 is defined.

Fisher ROC/ROC+ Specific Information

There are no Fisher ROC/ROC+ specific menu options. However, there are optional configuration .csv files the user can create. The ROC/ROC+ protocols use “TLP” addressing, which stands for “Type/Location/Parameter”. The T and P parts to that address defines a specific data type. For example, Type 1 (Analog Input)/Parameter 14 (Filtered EU) is a 4-byte floating point value. The ROC/ROC+ protocol defines thousands of possible T/P types. Protolyzer has a built in table containing most common T/P definitions. The table is currently over 2,200 entries. However, some less common T/P definitions may not be included. Further, some RTUs have custom T/P

definitions not in the standard. Thus, you may optionally provide .csv files to configure custom T/P definitions.

The .csv files should be located in the %APPDATA%\Telvent folder. You should also find Protolyzer.ini in that folder, to verify that you are working in the correct folder. There are two optional .csv files you can provide.

RocTypeNameTable.csv:

This file allows you to give a custom name to a type. For example, if you wish to give the name "Customer Defined Type 32" to type number 32, create RocTypeNameTable.csv with the following information:

"32", "Customer Defined Type 32"

Repeat for as many lines as necessary.

RocInfoTable.csv:
This file allows you to give a custom type and description string to T/P definitions. The format of the file is as follows:
"<type number>", "<parameter number>", "<data type>", "<description>"

For example, if you wish define Type 32, Parameter 2 to be a FLOAT, with description "User assigned float value" put in the file the following line:

"32", "2", "FLOAT", "User assigned float value"

Repeat for as many lines as necessary.

Protolyzer supports the following list of FisherROC types in the RocInfoTable.csv:
BINARY: (displayed in bits) unsigned 8-bit quantity (byte)
UINT8: unsigned 8-bit quantity (byte)
INT16: signed 16-bit quantity (word)
UINT16: unsigned 16-bit quantity (word)
INT32: signed 32-bit quantity (longword)
UINT32: unsigned 32-bit quantity (longword)
FLOAT: 32-bit float (single precision)
DOUBLE: 64-bit float (double precision)
TLP: Type/Location/Parameter
STRING1: 1 byte ASCII string
STRING3: 3 byte ASCII string
STRING6: 6 byte ASCII string
STRING8: 8 byte ASCII string
STRING10: 10 byte ASCII string
STRING12: 12 byte ASCII string
STRING20: 20 byte ASCII string
STRING30: 30 byte ASCII string
STRING40: 40 byte ASCII string

Series 5 Specific Options

For Series 5, you need to select either LRC8 or CRC16 as the security code method. This is done from the **Protocols** menu.
Also, control outputs can be addressed individually or in trip/close pairs. This is also configured from the **Protocols** menu

Vancomm Specific Options

Harris brand RTUs using Vancomm protocol use a different addressing standard compared to the original Ferranti standard. Thus select the addressing standard you need from the **Protocols** menu.

Searching and Extracting

The “Search” menu allows for a variety of searching and extracting methods to find and save data from your DNP data stream. Most of the time, the “Protocol Search” is the best choice for data searching. You can search for data to or from specific RTU addresses, for specific object classes, variations, and specific coordinates. Note, however, that for Protolyzer to find data, it has to reparse the file. For this reason, to save a step, you can open a file and immediately search it using the “File->Open & Search” menu option.

The “Data Extract” option operates somewhat similar to the “Protocol Search” option; however it will search for one specific coordinate, and place all the data changes for that coordinate into a chart. Again, this feature requires Protolyzer to reparse the file. And again, to save a step, you can open a file and immediately extract data from it using the “File->Open & Extract” menu option.

The “Data Extract” option also has an option to extract class scans. This is useful if you are wanting to review how often a master is polling for different classes.

There is also a plain text search option, which allows you to look for literal strings. This is usually most useful for finding things like specific timestamps. The plain text search option functions in both the “Original” tab and the “Parsed” tab.

SAGE RTU Trap Files

As already noted, Protolyzer will automatically detect Schneider Electric SAGE RTU trap files. These files have statistics on each byte transmitted by the RTU. Specifically, it has the time in milliseconds from the start of the capture of each byte, along with possible error states of framing, parity and overrun. Clicking on a specific byte of the parsed data will display these statistics on the status bar of Protolyzer.

Configuration Options

Default configuration suffices for most uses. However, Protolyzer can be customized to your preferences. The “Setup” menu allows you to customize Protolyzer operation. These options are stored in Protolyzer.INI so that the settings are retained.

Font: Allows you to configure the font displayed. Only fixed-pitch fonts are selectable.

Colors: Customize the colors used of the parsed text

Custom Data Delimiters: Protolyzer automatically understands “swana” capture files. However, if you are using some other source, you can customize what delimiter Protolyzer should look for.

Trap File Importing: Customizes the trap file import feature. The “extra 05/64 parsing” option tells the import tool to do an extra check for DNP start of message pairs. In general, making use of the intercharacter timeout is better than using the “extra parsing” option. The inter-character timeout is used to separate messages. The default of 200 ms should work in most instances. There is also an option to configure whether the RTU was operating as a master or as a remote. This is mainly for ModbusM and ModbusR parsing.

Attempt to parse extraneous: Swana captures will sometimes mark data bytes as “extraneous”, data that swana considers invalid data. If you wish Protolyzer to attempt to parse this data, use this option.

Filter lines marked as extraneous: Remove sections marked as extraneous by swana from the parsed output.

Filter message lines: Some versions of swana have comment lines marked as “message”. This option removes these from the filtered output.

Filter other comment lines: Remove comment lines from the parsed output.

Command-line Options

Typically, Protolyzer is run without any command line options and the user does all operations through the standard menus. However, sometimes it is convenient to run Protolyzer from a CMD prompt, or from another program. And to pass parameters to Protolyzer to know what protocol to use and/or what data to parse. If you type "Protolyzer /?" from the command-line, a pop-up showing the command line options will appear. The information on the popup is as follows:

Usage: Protolyzer [/MAX] [/NORM] [/P <protocol>] [/M "text to parse"] [/F <filename to open>]
(not case sensitive)

/MAX (or /MAXIMIZED) means window starts maximized
/NORM (or /NORMAL) means window starts normal

<protocol> can be:
- ABCIP (or AB-CIP)
- DNP3 (or DNP)
- FisherROC
- IEC104
- MODBUS
- SC1801
- SELFASTMETER
- Series5 (or SeriesV)
- VANCOMM
- HEXASCII (or HEX-ASCII or DUMP)

The last protocol used by Protolyzer will be used if /P option not present.

Cannot combine both /M and /F

Character #17 (0x11) is used as a carriage return when using /M

If providing a file name, and it is the last parameter, using /F is optional.

Line length limitation of /M is limited, depending on Windows version and how Protolyzer was launched