

IEDnet/IED-24/VIS Software Tools

Ver 1.0 16 April 2018

The document provides basic information about tools that have been developed for IEDnet or IED-24 communication diagnostics, and well as those for .VIS file manipulation. The tables below list software tools and contain links to the tool summaries which follow. In Word, simply Ctrl-Click on the EXE Name to jump to that section. In the PDF version of this document, simply click on the EXE Name to jump.

This is not a list of all tools, but a collection of those that are likely still relevant to debug or technical support as of this writing. It is also an attempt to highlight features of tools that may be less commonly known or understood, or at least provide a minimal “user guide” for these tools that currently doesn’t exist.

IEDnet Related Tools

EXE Name	Description
RebMon	Monitor rebooting IEDnet devices
INetBoot	Tool for sending IEDnet reboot command to IED (pre-IED-24) devices
NetMon32	This tool monitors IEDnet traffic on the local computer, and also can be used for generating test messages to IEDnet devices.
netReplay	Tools for playing back individual IEDnet commands, such as those captured by NetMon32.
netPlayback	Tools for playing back a script (batch) of IEDnet commands, such as those captured by NetMon32.
IEDRespTimer	Monitor IEDnet traffic to specific devices and compute response statistics

IED-24 Related Tools

EXE Name	Description
Interp24	Aid for interpreting IED-24 messages captured in NetMon32.
ViewProp	Tool for viewing/editing objects/properties in IED-24 devices (Titan/528)
ViewAddr	Tool for manually viewing/setting the network address in IED-24 devices (Titan/528)
I24PING	Ping IED-24 devices using IED-24 command to insure they are responding as IED devices

500VIS and .VIS File Related Tools

EXE Name	Description
VISMessageSender	Tool for sending VIS messages to 500VIS displays.
VISEdit	Tool for viewing/editing .VIS files
CreateVIS	Create a directory of VIS files from a CSV files documenting Take numbers and their text. Also can create Msg Scheduler INI file Takes.
DocVIS	Process a directory of VIS files and create a CSV file of Take numbers, their text and opt. duration, as an aid for Take library documentation.
VISCleaner	“Touch up” text properties contained in the headers of VIS files in a directory.
GCLbundler	Create a GCK Take Import Library file from a directory of WAV and VIS files.

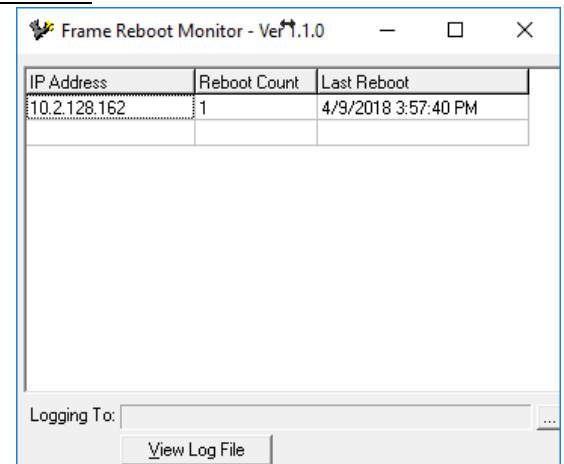
IEDnet Related Tools

RebMon

This tool is for monitoring the rebooting of IED devices that broadcast the IEDnet Network Connect status messages when they reboot.

Usage: For most people, it is sufficient to simply run the application and wait for it to detect device reboots. The only other option is to capture reboots to a log file by clicking on the [...] button to access a standard Save dialog box for selecting the destination for the log.

One can also specify the log file on the command line. This is useful for situations such as running RebMon from the Windows start folder.



INetBoot

This tool for sending IEDnet reboot command to IED (pre-IED-24) devices. IED-24 devices should be rebooted via the button in ViewProp. This is a command-line only program. One runs it with the target IP address on the command line, e.g.,

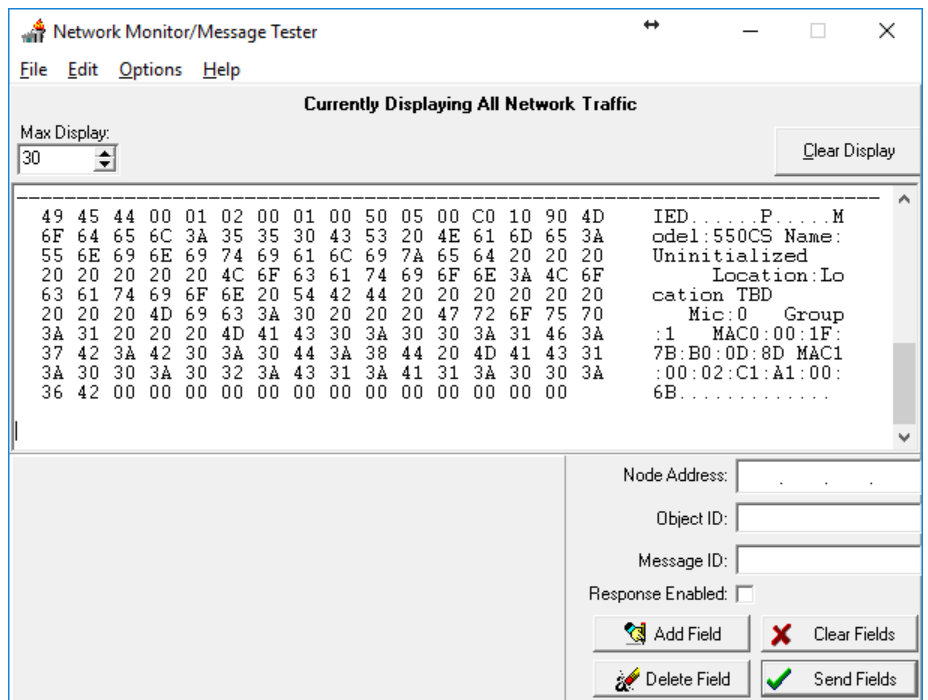
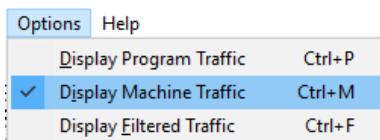
```
iNetBoot 10.2.150.100
```

In this manner, this tool might be employed in a batch file or as part of a startup script.

NetMon32

This tool monitors IEDnet traffic on the local computer, and also can be used for generating test messages to IEDnet devices.

Usage: There are two basic functions of this tool: monitoring IEDnet traffic and generating test messages (while viewing the responses to them). When monitoring, one has the option of filtered or unfiltered (i.e., all traffic). One can change the filtering via the Options menu or associated hot-keys:



The terminology here means the

following:

- Program Traffic – Responses to messages generated by NetMon32, using the fields on the lower portion of the main window (discussed below).
- Machine Traffic – All IEDnet traffic on this computer
- Filtered Traffic – IEDnet traffic filtered by one or more criteria. Selecting this option makes the edit boxes for these criteria visible as below. Filter IP bytes are entered in decimal, other fields in hexadecimal (e.g., Filter Object of 1000 means 0x1000).

Currently Displaying Filtered Network Traffic


Filter IP:	Filter Object:	Message ID:
<input type="text"/>	<input type="text"/>	<input type="text"/>

Monitor data only shows the last **Max Display** number of packets in the window. This value has a maximum of 60 packets. Monitor data can also be recorded to a file via the **File** menu --> **New Log File** option. When selecting this option, one is prompted to record in more compact “simple format” which lists one packet per line of the file, but only the first 16 bytes of data, but with the header decoded as shown in the example below:

```
*****
2018-04-09 11:56:42:485 10.255.255.255 -> 0x0100/0x0001 0203
2018-04-09 11:56:42:489 10.2.128.201 <- 0x0100/0x0001 0203
2018-04-09 11:56:42:505 10.2.50.100 <- 0x0100/0x8001 0203 <0><C8><0><1><1><0><3>JFK
2018-04-09 11:56:42:551 10.172.30.81 <- 0x0100/0x8001 0203 <0>k<0><1><D3><1><E>Display S
*****
```

In this format, the direct (sent or received) is indicated by the “arrows” between the IP address and the object IDs (0x0100 in the example). Left-to-right indicates outgoing packets. The decode just to the right of the “arrows” show the Object ID/Message ID and Message Number from the IEDnet headers. If simple format is not selected, the file will look just like the packet display shown in capture window of NetMon32.

Filtering can also be done on multiple simultaneous filters and only during select times by editing the **Schedule** found under the **Edit** menu. This allows one to log to multiple log files simultaneously as well, although it is acceptable/legal to have all entries refer to the same log file. The simple format cell should be given a 1 to enable the simple format for this schedule entry.

 Capture Schedule

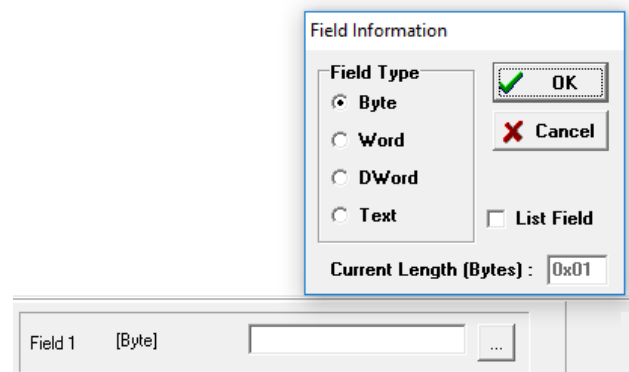
Scheduled Logging is Disabled While Schedule Window is Open

Start Time	Stop Time	IP	Object ID	Message ID	Log File Name	Simple Log
3:23:09 PM	11:59:00 PM		3000		C:\Documents and Settings\jgreen\Desktop\Test.L	1
3:23:09 PM	11:59:00 PM		4200		C:\Documents and Settings\jgreen\Desktop\Test.L	1
15:00	11:59:00 PM				C:\Documents and Settings\jgreen\Desktop\Tes7.	1

The Schedule can also be defined in the optional NetMon32.INI file. Defining them in the INI file allows one to enable NetMon32 to automatically load a (filter) schedule whenever it is run, such as configuring it to run in the Windows Start menu, so it will re-initialize properly after a system reboot. The recommended way to generate this INI file is to edit the Schedule and click

the **OK** button. This will write the INI file out in the proper format. If you do not wish to have that schedule active anymore, you can delete or rename the INI file manually.

To **generate test messages**, one must fill in the Node (IP) Address, Object ID and Message ID, the latter two in hexadecimal without formatting characters (i.e., just 1000 for object ID 0x1000). One can optionally add data fields for the message by clicking on the **Add Field** button for each additional data field. One defines the data type for the field (default: Byte) by clicking on the [...] button beside the field edit box. The **List Field** checkbox indicates that the field is a list of Bytes, Words or DWords (never Text), which are entered by separating the values with commas.



The **Current Length** readout is handy for Text fields, where the text length goes into the IEDnet message as the field preceding the Text field. One can enter the text first, then call this window back up, read the length of the text field, and then use this value in the preceding field.

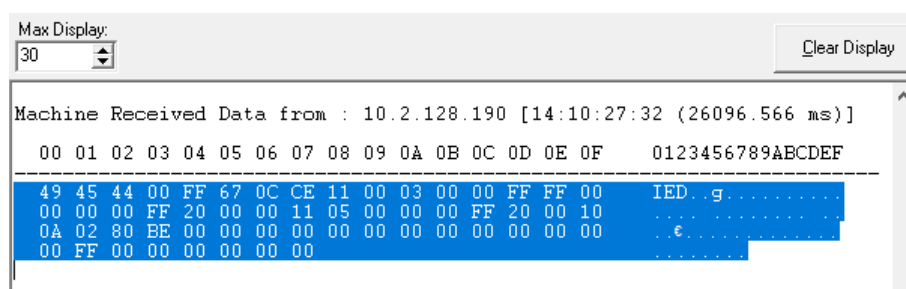
One manually sends the message with data fields if defined, by clicking on the **Send Fields** button on the main NetMon32 form. Messages can be saved to a file and later recalled from file via options under the **File** menu.

The **Response Enabled** checkbox on the main NetMon32 form allows for some *automatic* message generating capabilities. When enabled, NetMon32 will monitor incoming traffic and when it sees a match to the Object ID and Message ID fields, it will generate a response message (including ORing in the response bit 0x8000 with the Message ID) and all the defined data fields back to the sender.

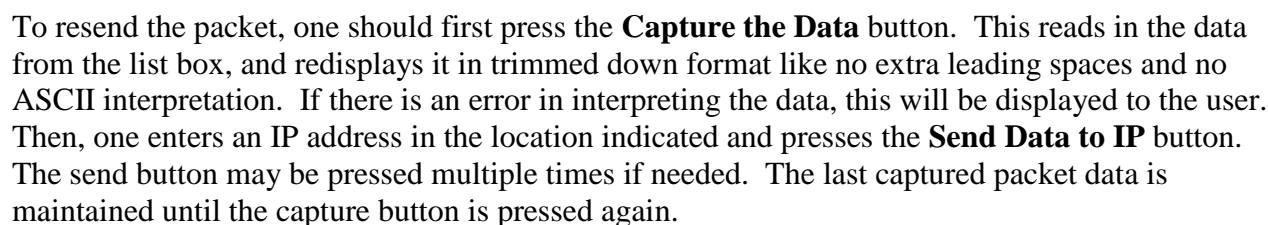
netReplay

This tool is for playing back one IEDnet file previously captured with NetMon32 or other tool, including Wireshark.

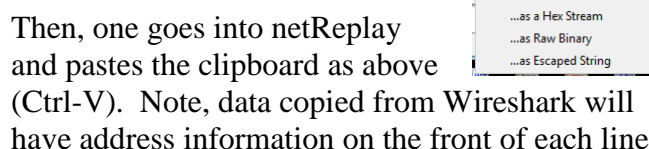
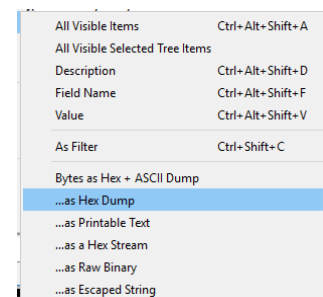
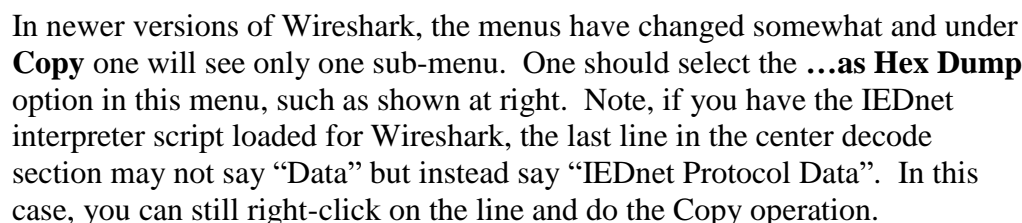
Usage: To use with NetMon32, one should copy a packet (not in simple format) to the Windows clipboard from the NetMon32 window or log file. One should copy *everything* below the line of dashes in a capture, such as is shown below, by first highlighting with click-drag and then using the Ctrl-C shortcut to copy it to the Windows clipboard.



Next, one should click into the list box area of netReplay when it is clear (e.g., press the **Clear Data** button first), and paste it with the Ctrl-V shortcut. Usually, one should be sure **Data Type** is set to "Hex w/ ASCII". The option for Com-Net data type is rarely used.



To copy an IEDnet packet from a Wireshark capture, one should Find the packet, go to the Data portion in the decode window (middle area of Wireshark), and right-click on it. Then follow the pop-up menus to select **Copy, Bytes, Offset Hex** as shown in the example below.



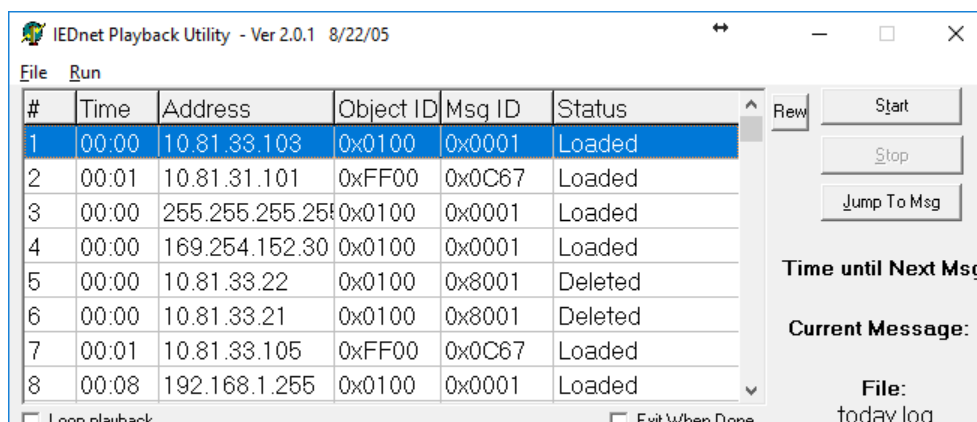
(0000 on the first line, 0010 on the second line, etc.) These numbers must be deleted in the list box before pressing the Capture button.

netPlayback

This tool is for playing back a whole script of IEDnet messages, such as a log file previously captured with NetMon32.

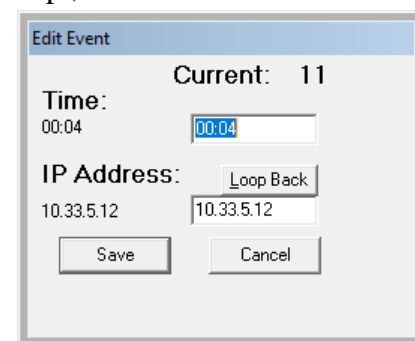
Usage: One should start with a NetMon32 capture file or copy the contents of the NetMon32 capture window (multiple packets) to a text file. The netPlayback tool will automatically ignore any packets in the capture which are responses rather than commands. The tool can read in several types of files. By default, the Open dialog box is set to read in netPlayback scripts, or files ending in .NSC. One may also select NetMon log files, plain text and other file extension options in the Open dialog box. When a script is loaded, the main window will display the messages as shown below. Note the Deleted statuses on the response lines. These packets will not be played, and when this script is saved back out, they will be dropped from the file.

One can put the script file on the command line when starting netPlayback to have it load automatically. If one sets up a file association in Windows to open .NSC files with netPlayback, then simply double-clicking on any .NSC file will open it in netPlayback.



At this point, one could play this script verbatim by pressing the **Start** button. By default, it will play to the end and then stop. To have the script loop over and over, one should check the **Loop playback** checkbox. Alternately, one could have the program exit when the end of the script is reached. The **Rew** button moves the cursor back to the top packet in the script, such as for another manual play once through. One can also click on any packet (row) in the list and click on the **Jump to Msg** button to have it play through the script *starting at the highlighted row*.

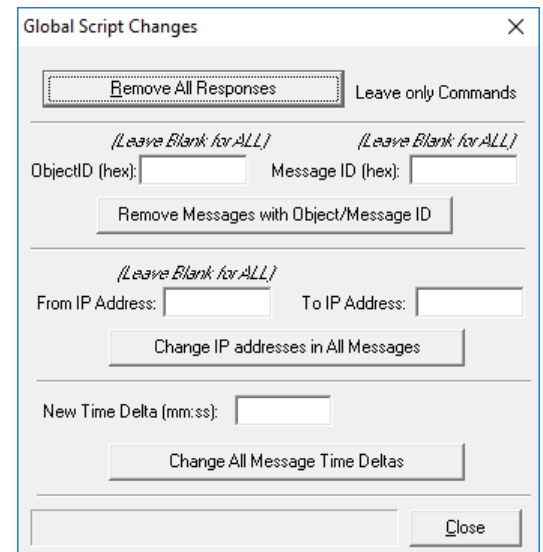
The time column is relative time, meaning the amount of time to delay before sending this packet. By default the internal script clock is running once per second, so packets will go out no faster than that. That is, even if the time column says 00:00, there will be a one second delay before it goes out. To speed this up, one can set the timer interval via the **Set Timer Interval** under the **Run** menu. The units here are milliseconds, i.e., 1000 = one second.



There are various tools for editing scripts inside netPlayback. Double-clicking on a row, brings up a dialog box to edit that row, such as shown at right. One can change the (delay) time and destination IP address. Pressing the **Loop Back** button is a quick way to set the IP address to 127.0.0.1.

There is also an option to do global changes to the script via the **File** menu, **Global Change...** option. The available edit options are shown in the four sections of this change window shown at right. As indicated some fields can be left blank to indicate “all”. For example, one can change all messages to go to a particular IP address, such as an IED device under test, by leaving the **From IP Address** field blank, and only entering a **To IP Address** value. Alternately, one can change only messages with a select IP address by entering the **From IP Address** value where indicated.

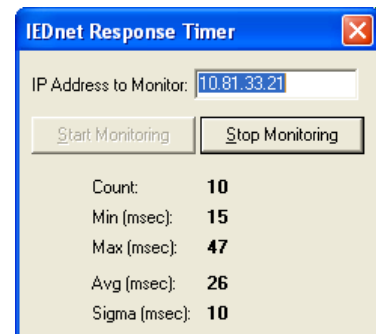
The editing is done when any of the four upper buttons is pressed, and can be played back after that if desired. These changes are saved to a file when the file is saved via the **File** menu, **Save** option on the main window.



IEDRespTimer

This tool monitors traffic to/from a specified device and computes statistics on device responses. This tool might be useful for testing device responses as well as testing network performance.

Usage: After launching this tool, configure the IP address and press **Start Monitoring**. Then, one would go into some other tool or application like ViewProp or IPAUConfigTool and do operations there such as load data. This tool matches up responses with commands, notes the time difference and computes statistics on it, such as in the example screenshot at right.

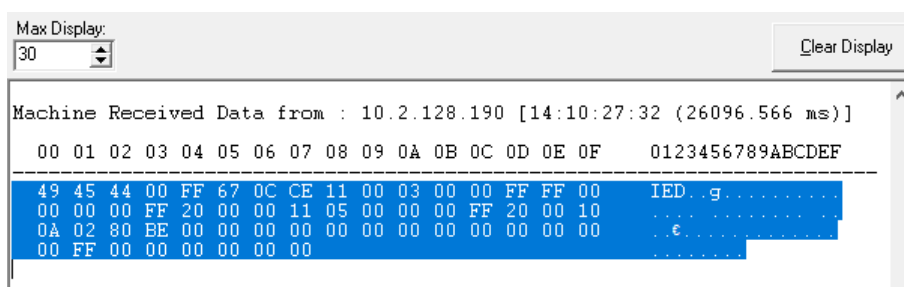


IED-24 Related Tools

Interp24

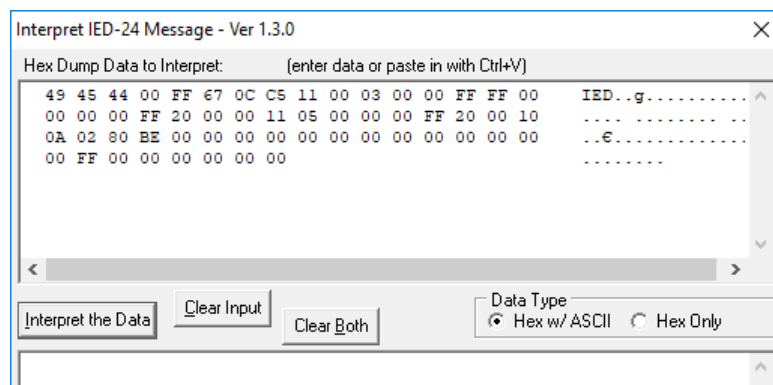
This tool is typically used in conjunction with NetMon32 to decode the IED-24 headers that can be contained in IEDnet messages.

Usage: Typically one copies an IED-24 message from the NetMon32 capture, either from the main window list box (using Ctrl-C to copy highlighted packet) or from a capture file. One should copy *everything* below the line of dashes in a capture, such as is shown below, by first highlighting with click-drag and then using the Ctrl-C shortcut to copy it to the Windows clipboard.



Then one should click in the top list box of Interp24 (or clear the box first via the **Clear Input** button), and then paste the text with the Ctrl-V shortcut. By default the **Data Type** will be “Hex w/ ASCII” and should be left this way.

Then one clicks on the **Interpret the Data** button. This generates an interpretation such as the one shown below. The second line shows the Method ID and whether a command or response (vis the > sign). If the method is a common one, then the command name is listed on this line in parentheses such as the lower example (I24GetAllObjs).



```
IED-24 Message
Method 3m103, >Command
  Message #: $11C5, Status: $00 $03
  To Object: 0-65535-0, From Object: 0-8447-0
  Data Bytes: 11 05 00 00 00 FF 20 00
-----
```

```
Method 1m0 (I24GetAllObjs), <Response
```

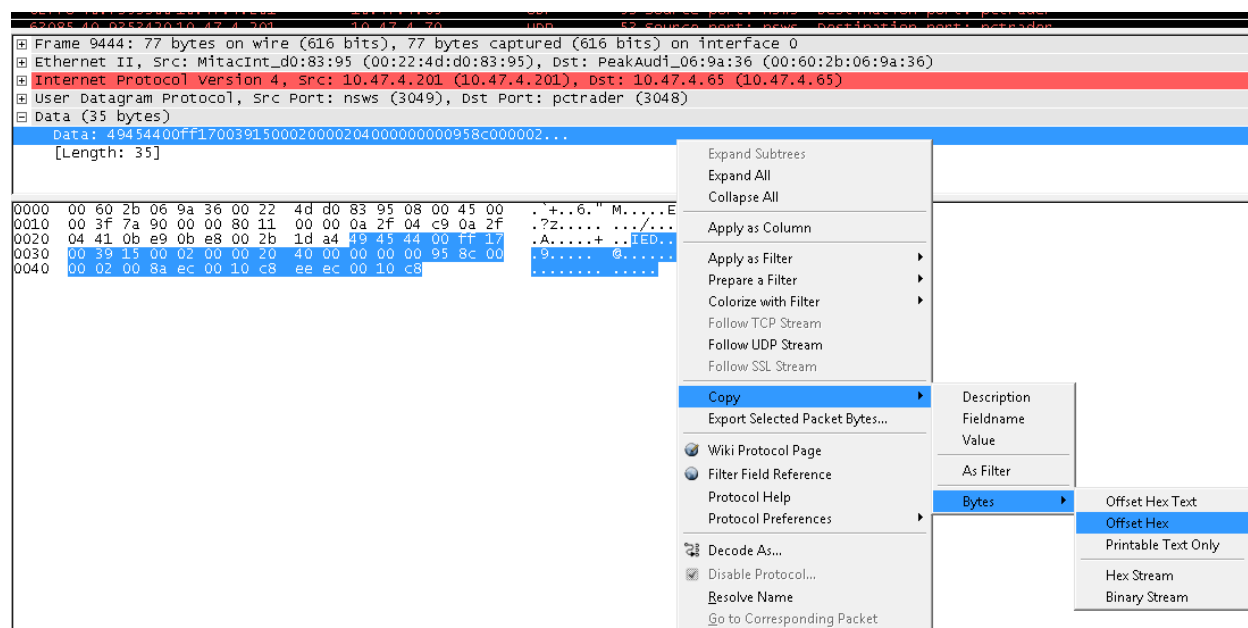
The other lines indicate portions of the IED-24 header. The last line shows the first few bytes of the data after the header. It is important to know that not all bytes are shown, but by having the first few shown, users should be able to look back at the original packet capture and locate the data portion to extract any additional information.

There are a few special circumstances such as the I24Get and I24Set commands, where the program will provide additional interpretation of the data bytes such as the example below.

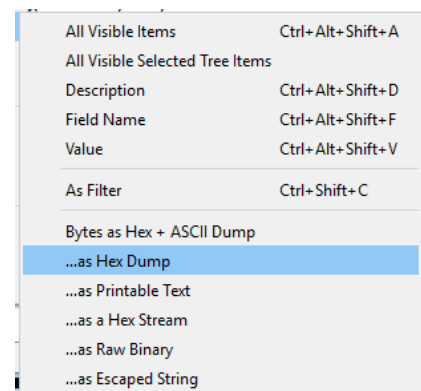
```
IED-24 Message
Method 0m2 (I24Set), >Command
Message #: $0E06, Status: $00 $00
To Object: 0-32752-100, From Object: 0-64657-1
PropID = 2p0 (1 Bytes) Data = 105/105/-47.5dB
-----
```

In this case instead of showing raw data bytes, it displays an interpretation with the data with field ID and value decoded in a few ways (signed byte, unsigned byte and IED db1) depending on what data type it is.

To get a packet captured from Wireshark, one should find the packet, go to the Data portion in the decode window (middle area of Wireshark), and right-click on it. Then follow the pop-up menus to select **Copy, Bytes, Offset Hex** as shown in the example below.



In newer versions of Wireshark, the menus have changed somewhat and under **Copy** one will see only one sub-menu. One should select the **...as Hex Dump** option in this menu, such as shown at right. Note, if you have the IEDnet interpreter script loaded for Wireshark, the last line in the center decode section may not say “Data” but instead say “IEDnet Protocol Data”. In this case, you can still right-click on the line and do the Copy operation.



Interpret IED-24 Message - Ver 1.3.0

Hex Dump Data to Interpret: (enter data or paste in with Ctrl+V)

```
0000 49 45 44 00 ff 17 00 39 15 00 02 00 00 20 40 00
0010 00 00 00 95 8c 00 00 02 00 8a ec 00 10 c8 ee ec
0020 00 10 c8
```

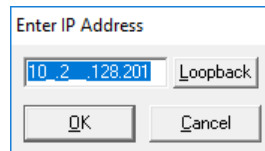
Then, one goes into Interp24 and pastes the clipboard as above (Ctrl-V). Note, data copied from Wireshark will have address information on the front of each line

(0000 on the first line, 0010 on the second line, etc.) These numbers *must be deleted* in the list box before pressing the **Interpret the Data** button.

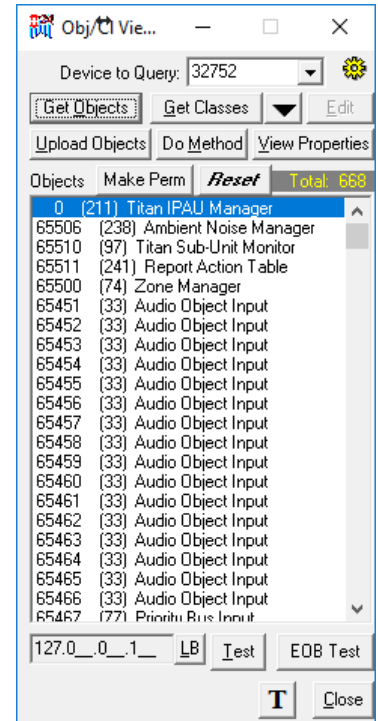
ViewProp

ViewProp (View Properties) is a tool to inspect and modify any IED-24 object on devices that support it. In addition, there are some control methods available as well as real-time meter monitoring capabilities in ViewProp.

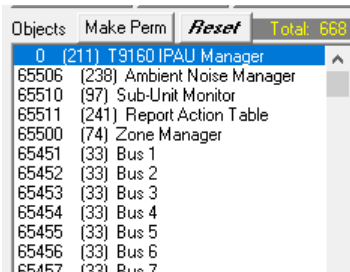
Usage: There are several ways to use ViewProp. The most common way is to simply inspect and modify objects. To do this, one sets the Device ID in the **Device to Query** box (32752 is the default that any device should respond to), and the IP address in the box near the bottom. Clicking in the box, brings up an IP edit window such as shown below.



One types in the new IP address, followed by the **OK** button. The **Loopback** and **LB** buttons set the IP address to 127.0.0.1, which is useful shortcut when running a device simulator on the same computer as ViewProp. Once both of these are set, one clicks on the **Get Objects** button to get a listing of Object such as the example shown at right.



Sometimes, the object name displayed is a generic name that doesn't

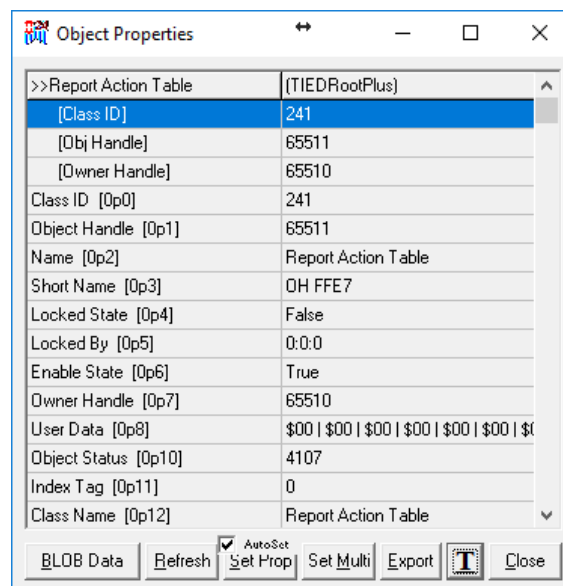
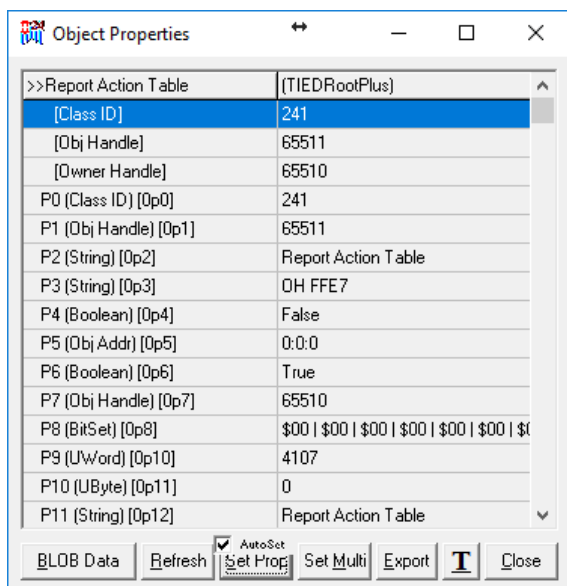


provide much information about the real purpose of the object.

One can load any additional descriptions for the objects by clicking on the **[T]** button at the bottom. This loads new information for *some* objects such as the example shown at left, which are the same objects as above with descriptions loaded/displayed.

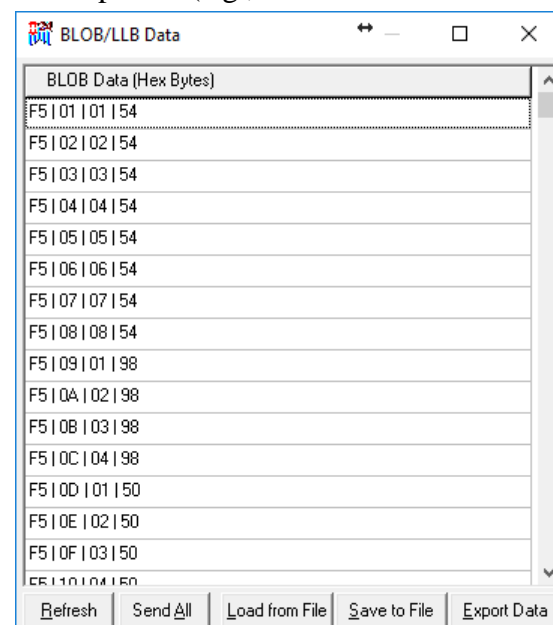
In typical usage, the next step is to inspect an object. This is done by either selecting an object in the list and clicking on the **View Properties** button, or using the shortcut of double-clicking on the

object of interest. This will bring up a view of properties with their hierarchy designation (e.g., Op1, 2p2) and data type as shown below at left. One can load a list of descriptions for the properties from the device by clicking on the **[T]** button at the bottom.



To modify a property, one clicks in the data cell for the property and enters a new value. Then, if the **AutoSet** checkbox is set, one can simply hit the [Enter] key on the keyboard to send the value. Alternately, one needs to click on the **Set Property** button at the bottom. It is usually a good idea when setting properties to click on the **Refresh** button to pull the values back and verify they got set as expected. Reasons they may not get set as expected include: (1) the property is read-only, (2) the property does not have the resolution expected (e.g., no fractional values, or fractions to only a certain resolution like 1/2 or 1/4 units), and (3) the property didn't get Set.

If the object is one of the Binary Large Object (BLOB) type data holders, then the **BLOB Data** button will appear. Clicking on this brings up a window that displays the raw BLOB data, such as the example shown at right. If one knows what they are doing, they can edit the data in the grid and update the object with the data via the **Send All** button. This is not straightforward, so typically the only use of this window is to: (a) determine that the data is filled in or not (i.e., some non-zero rows exist) and (b) how many rows are filled in (i.e. is the table complete). The better way to edit



Current Elements [3p2]	64
Selected Element [3p3]	2
Selected Element Data [3p4]	\$F5 \$02 \$02 \$54
Table ID [4p0]	1
Selected Unit Number [4p1]	245
Selected Sensor Number [4p2]	2
Selected Action Type [4p3]	1
Selected Action Code [4p4]	5
Selected Action Item [4p5]	2

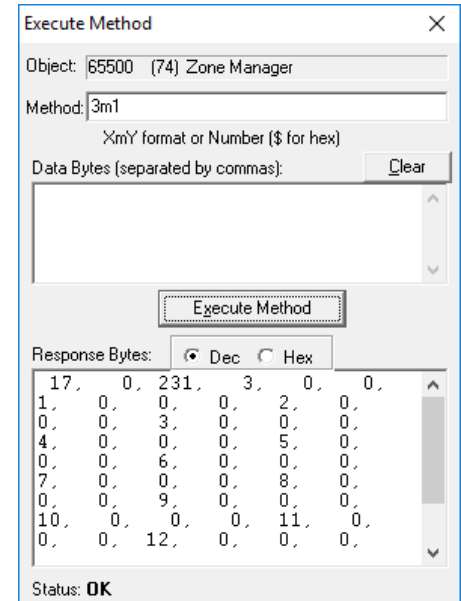
this data for most users is back in the object properties window. To do this one, sets the “Selected Element” property to a row number (1..N), followed by **Set** and **Refresh** buttons. This retrieves the selected element, broken down into the properties below it. One can usually see the raw form in the “Selected Element Data” property just like it appears in the BLOB

view, followed by the meaning of that raw data. Every BLOB table type is different in what data fields are shown. The values can be edited right here, followed by **Set**, if desired.

The **Get Objects** operation described above, really only gets a list of object names and identifiers. The **Upload Objects** button actually retrieves all properties for all objects and allows some actual manipulation of the objects, such as the EQ bands in a Titan frame.

***Note:** Due to changes in IEDnet starting with Windows 7, this feature no longer functions properly and should not be used.*

Advanced users, with access to the IED-24 reference manual, can use the **Do Method** feature to execute methods of objects. To use, one clicks on an object of interest (e.g., Zone Manager) and then the **Do Method** button. This brings up an Execute Method dialog box such as the one shown at right. One enters the Method ID in XmY notation, where X is the object hierarchy level and Y is the method ID (same notation as used in the reference manual). Then, if the method takes parameters, these are entered in the Data Bytes area. Note, these must be bytes in decimal or in hexadecimal if one uses the dollar sign (\$) prefix (Delphi convention). So, if the method takes words or double-words, it is up to the user to break these down into bytes in the proper order (lowest byte first) when entering them in this area. If no parameters are required for the method, then leave this area blank. The **Execute Method** button takes the Method ID and Data Bytes, sends them to the device, gets back any response and displays the response data bytes (parameters) in the lower list box. The example at right is the current state of all the zones in the zone manager. One can see the Zone numbers (1, 2, 3, ...) in the data with lots of zeroes in between, because all zones were idle when this was executed.



The screenshot shows the 'Execute Method' dialog box. The 'Object' field is set to '65500 (74) Zone Manager'. The 'Method' field is set to '3m1'. The 'Data Bytes' field is empty. The 'Execute Method' button is highlighted. Below the dialog box, the 'Response Bytes' are displayed in a list box, showing a grid of values: 17, 0, 231, 3, 0, 0, 0, 0, 3, 2, 0, 0, 0, 0, 3, 0, 0, 0, 4, 0, 0, 0, 5, 0, 0, 0, 6, 0, 0, 0, 7, 0, 0, 0, 8, 0, 0, 0, 9, 0, 0, 0, 10, 0, 0, 0, 11, 0, 0, 0, 12, 0, 0, 0. The status at the bottom is 'OK'.

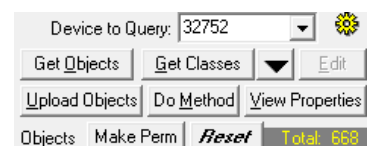
An example of using Data Byte parameters might be to set a zone state, with method 3m0. In this example, we want to set the zone state to active (value=3), connecting to bus 2, at priority 1, and page level of 0 dB (which is 200 in scaled dB1 data type). In the example, we want to set one zone, number 7 to this state/bus/priority/level. The Data Bytes for this example would be:

3, 2, 1, 200, 1, 0, 7, 0

Note, most parameters are bytes, but since the zone count and zone number parameters are words, the extra bytes of zeroes are needed there to fill out the words. Again, this operation can only be done by advanced users with access to all the IED-24 object and data type documentation.

There are two often used buttons on the main window:

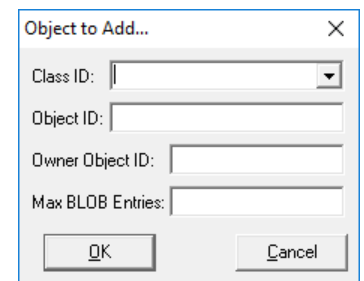
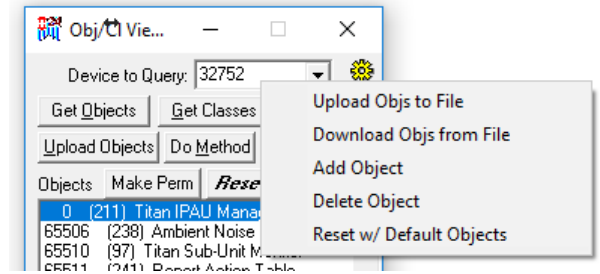
- **Make Perm** – This tells the device to save all the current properties to non-volatile storage to become the default after any future reboot of the device.
- **Reset** – This tells the device to do a soft reboot. Of course, if a Make Perm operation was not performed beforehand, this will clear properties that were recently set.



The screenshot shows the main window of the IEDnet-IED24-VIS Tools. The 'Device to Query' dropdown is set to '32752'. The 'Get Objects' button is highlighted. Other buttons include 'Get Classes', 'Edit', 'Upload Objects', 'Do Method', 'View Properties', 'Objects', 'Make Perm', 'Reset', and 'Total: 658'.

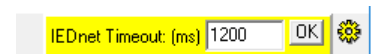
The button with a down-arrow (▼) brings up a menu of additional more advanced operations:

- **Upload Objs to File** – Get all objects and their properties and save to an .IOB file for possible future restoration or just to preserve any special setup, such as “hardwiring” that has been done on some convention center type projects.
- **Download Objects from File** – Send all objects in an .IOB file to the device. This action first prompts to delete all objects in the device first or not. Often, this operation is used to “patch” a few objects or add some additional DSP processing, in which case the objects should not be deleted first. Generally, .IOB files come with instructions from Engineering on whether to delete all objects first or not.
- **Add Object** – This is a really advanced feature that requires knowledge of IED-24 objects and how they relate (owner objects). One will have to fill in the fields in the dialog box that pops up, shown at right.
- **Delete Object** – This is an advanced feature. One selects the object in the list area first before clicking on this option, and answering Yes to the confirmation pop-up box.
- **Reset w/ Default Objects** – This command tells the device to reboot and load the *factory default* objects. If one makes a mistake with the Download Objects, Add Object or Delete Object operations, the changes can be erased via this operation. If one has not done a Make Permanent operation yet, those mistakes can be erased with a simple (soft) Reset, rather than resorting all the way back to factory defaults.



*Note, changes to the device done with the Download Objects, Add Object and Delete Object operations will not be retained through a power cycle until the Make Permanent operation is performed via the **Make Perm** button.*

Clicking on the gear icon, brings up the available settings for ViewProp, of which there is only one, the IEDnet time-out value (default 1200 msec). if one is dealing with a particularly slow device/network, increasing this value, e.g., to 2500 or 3000 might be helpful.



ViewAddr

This utility allows viewing and modifying the IP address(es) or an IED-24 device.

Usage: Runing the exe brings up a window such as shown at right. The default IP address of 10.2.150.173 is one of the default IP addresses for devices out of the factory (T9160 frames). You will want to change it to the current IP address of the device, first. The list of default IP addresses is in on the IED in-house network file server at *P:\Docs\Default IP Addresses.txt*. The **Set Defaults** button sets the IP Address back to the 10.2.150.173 value (the most commonly used default).

The device handle of all F's is a default that any IED-24 device should respond to, so one can talk to the device even if the true device handle is not known. Once communications is established via one of the other operations, this edit box is changed to show the actual handle. It can be changed back to all F's, such as when changing to talk to a different device, by clicking on the **FFFF** button.

A new device handle can be set by entering the value where indicated and clicking on the **Set Handle** button, and responding Yes to the prompt presented. The **Verify & Set Handle** button will check that the handle already in the device matches what is entered and if not, optionally, set it to match (an additional user prompt).

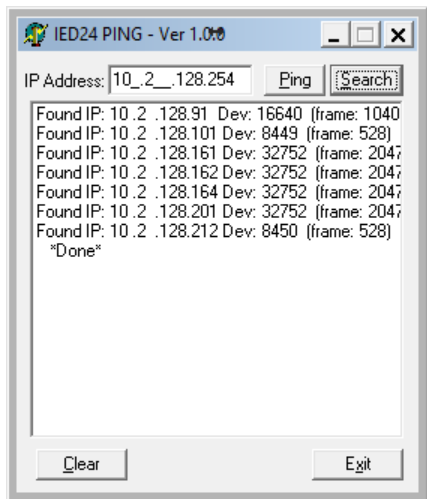
The most common use of this tool is to set the transport address(es) of the device. Most devices only have one transport channel, channel 1 (or hex \$001). The T9160 family has this port, which is the direct channel from the CPU (almost never connected to the outside world) and the channel that goes through the CobraNet CM-1 module, channel hex \$101. One needs to set the Transport Channel property to either \$001 or \$101 before pressing either of the transport action buttons. For example, setting this to \$101 and pressing **Get Transport**, retrieves all the information as shown at left. Setting new values of IP address, subnet mask and default router (gateway) are done in the obvious manner. Note, they are all set at once and changes *might* make the device unreachable by the computer running ViewProp, should the IP be changed to a different subnet.

The lower portion is for viewing the Serial Number (S/N) of the IED-24 device. The **Retrieve** button is used to pull it back for display in the edit box. The **Query** button is rarely/never used. It can be used to query a device *if the serial number is already known* (e.g., was printed on a device label). It will error out if the serial number field is blank or incorrect. The Query button will pull back the transport information if the serial number matches.

I24PING

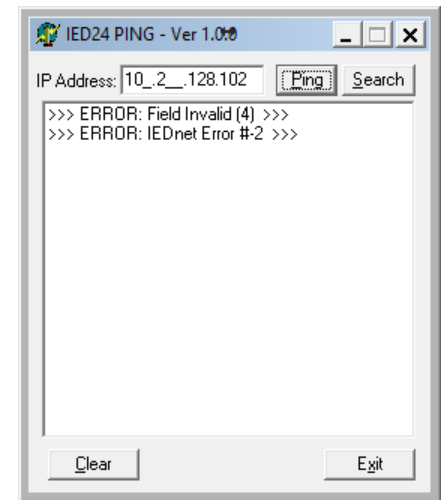
This tool is used to check IP addresses for network devices that support the IED-24 protocol.

Usage: There are two basic uses for this tool: (1) check a specific address and (2) search for devices on the network. The specific address check is done by first entering the IP Address, and then clicking on the **Ping** button. One may get an IED-24 error back from the device, but *any IED-24* response means it is an IED-24 device at that IP address. If, instead, one gets an IEDnet error -2 (time-out), this means the IP address is not valid or the device is not an IED-24 device. In the example at right, the top response is an IED-24 response from a valid IP address, while the second response is not valid.



The other use of this tool is to cycle through all IP addresses where the last octet of the IP address is varied from 1 to 254. One enters an IP address to set the top portion of the address and then presses the **Search** button. The tool will cycle through all addresses, showing the current address in the edit box as it goes. As it finds each device, they are printed in the list box area as shown in the example at left. When it is complete, it prints the **Done** line. This process typically takes about one minute to complete.

The **Clear** button simply clears the list box area, such as before a new Search operation.

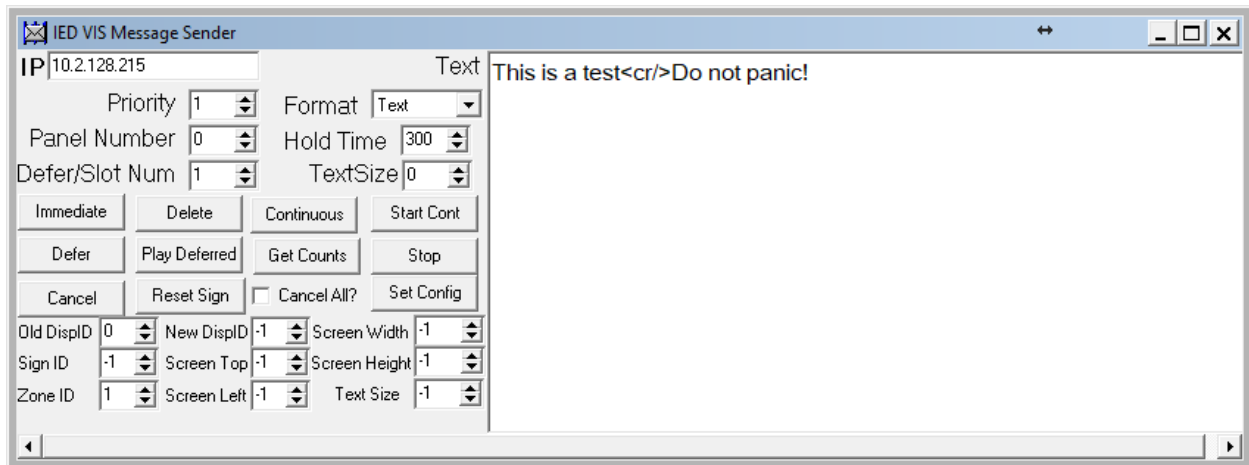


VIS Related Tools

VISMessageSender

This is a tool for sending VIS test messages to IED500VIS devices like signs/displays running VisDID (immediate or continuous).

Usage: The typical usage of the tool is in sending test messages to IED displays. One enters the IP address in the top box, which could be a broadcast address. One enters the text to display in the large **Text** box, including any formatting mark-up supported by the displays. One then sets either the **Sign ID** or **Zone ID** to a positive value. Also, for immediate messages, one should set a non-zero value in the **Hold Time** field. A zero in this field will tell the display to not remove the last page of the message until another message needs to be displayed. Typical value for this is 300, which corresponds to 5 seconds (the time is in units of $1/60^{\text{th}}$ of a second). When everything is set, one usually clicks on the **Immediate** button to send the message and have it displayed once on the display(s).

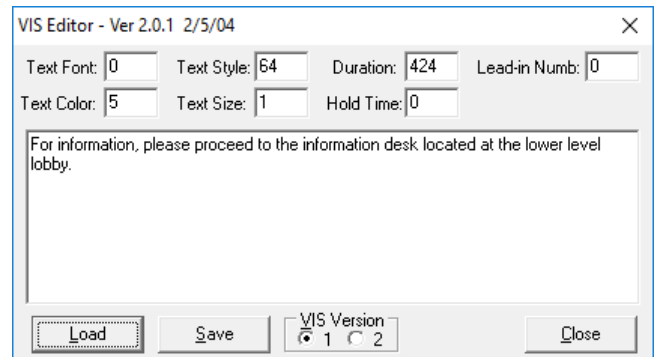


There are buttons also for sending continuous message, deferred message (immediate message that doesn't play until the Play Deferred command is sent). But all these functions are for really advanced users testing their 500VIS code.

VISEdit

This tool is used to edit IED .VIS files.

Usage: One can edit (or create) VIS files with this tool. The common usage is to open a VIS file via the **Load** button. The header is parsed into the fields at the top, and the text content is displayed in the list box area. The VIS file version from the header is used to set the radio buttons at the bottom. One can edit any of the fields or the text before pressing **Save** to store it back into a file. "Editing" includes, changing the setting of the VIS Version before saving it back.



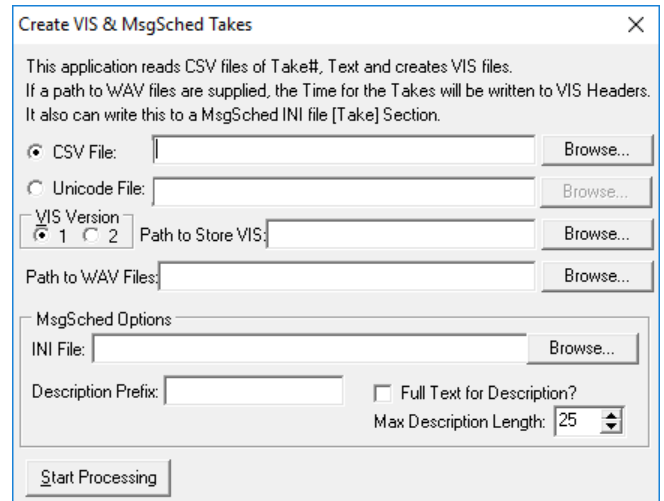
VISEdit will also accept the name of a VIS file on the command line, which means if you configure Windows to associate this application with files of type *.VIS, then one only needs to

double-click on the VIS file to open it in VISEdit.

CreateVIS

This tool is used to create a number of VIS files from a comma-separated variable (CSV) file. (CSV files are one format that Excel can read in and display in rows and columns.) Typically, this tool was used by IED recording engineers who created new Takes in a database and exported the descriptions (text) to a CSV file from which this tool could be used to create the VIS files. It also works as the inverse of the file created by the DocVIS tool.

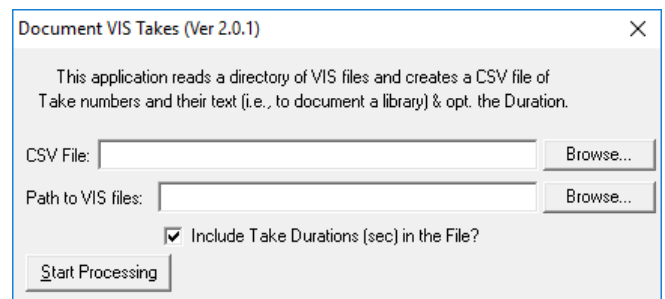
Usage: One selects either a CSV or Unicode text (CSV) file as the input source via the top choices. Then, one defines whether to output VIS version 1 or 2 files. (Pretty much only legacy 8000 transit systems use the version 1 files, so usually you set this to version 2.) Optionally, one can point to a directory that contains the corresponding WAV files. If this last step is included, then the CreateVIS tool will check the play time of each associated WAV file (matching Take numbers in filenames) and include the duration into the VIS file headers of each created file. One can either type in the filenames and paths or use the **Browse** buttons to find them. The MsgSched options shown are no longer used and can be ignored. After setting up the files and paths, one presses the **Start Processing** button to kick off the process.



DocVIS

This tool is used to create a comma-separated variable (CSV) file. (CSV files are one format that Excel can read in and display in rows and columns, as well as being available to import into other documentation.)

Usage: One simply fills in the name of the CSV file and the path to the VIS files, either by typing them into the appropriate edit boxes or by clicking on the **Browse** buttons and navigating to them on the local system. After setting up the files and paths, one presses the **Start Processing** button to kick off the process.

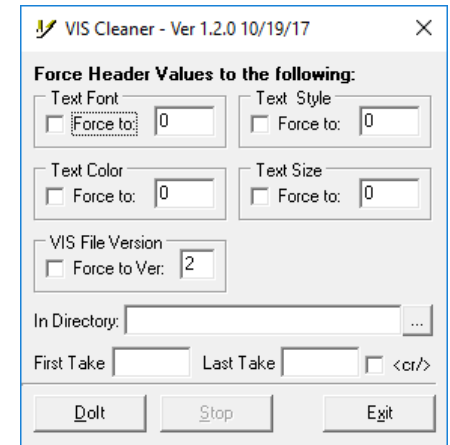


VISCleaner

This tool is used to touch up the headers on a bunch of VIS files all at once (e.g., a whole directory's worth).

Usage: The tool allows tweaking any of the four text properties:

- Text Font – number corresponding to entries in the VisDID.ini file
- Text Style – number corresponding to entries in the VisDID.ini file
- Text Color – number corresponding to entries in the VisDID.ini file
- Text Size – zero for “default” or the number of lines to fit on the display (typically 4 for landscape displays). VisDID will select a font size that makes the selected number of lines fit.

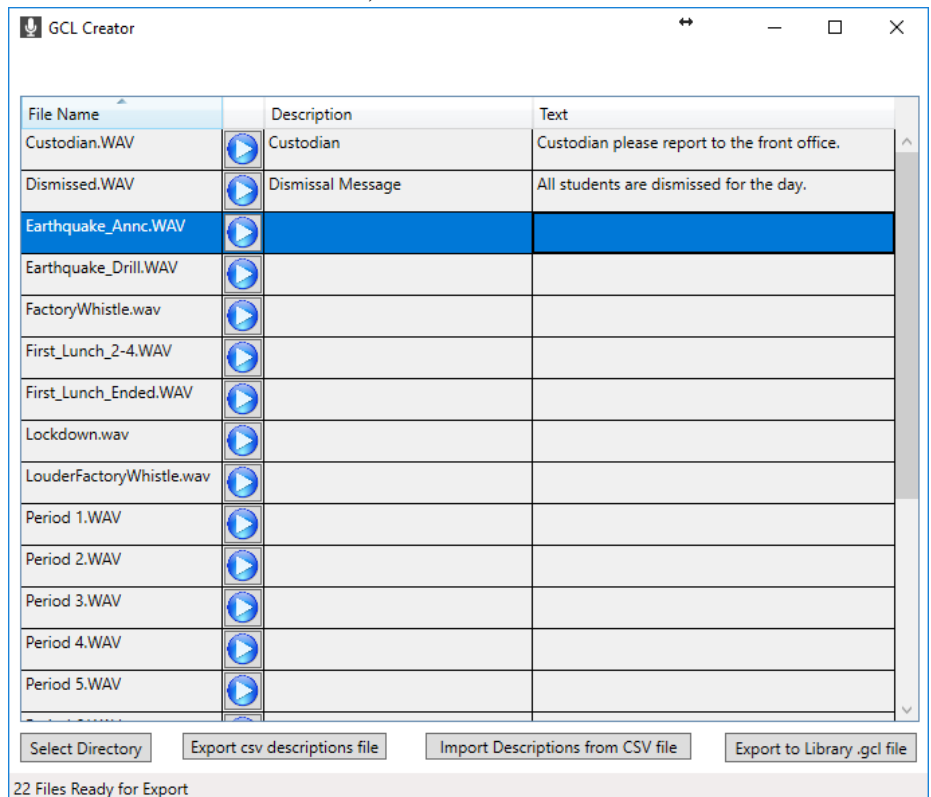


Plus, one can force the VIS file version to a value (only 1 and 2 are defined as of this writing).

GCLbundler

Create a GCK Take Import file from a directory of WAV and VIS files. Technically, this is both a WAV and VIS tool, but since what it does it bring multiple VIS file contents into a library for importation into GCK, where it is stored in the vACS.xml file, that it is included here.

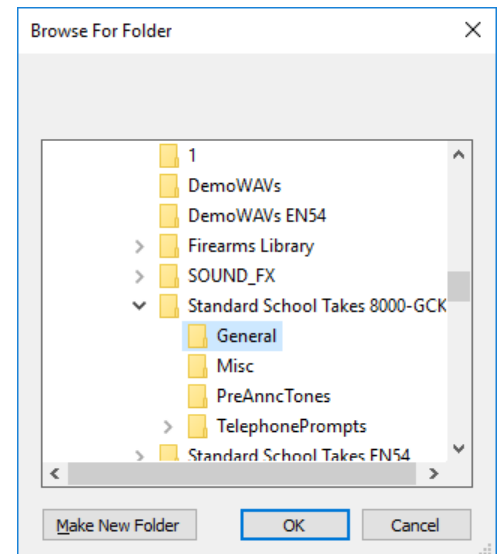
Usage: The GCL Bundler has one main window as shown at right. Before running it, one should copy all the WAV files one wishes to put into the bundle file into one directory. If .VIS files are available, one should copy them to the same folder.



So, first one clicks on the **Select Directory** button, which brings up a navigation window such as shown at right. One navigates to the directory with the desired WAV files in it and clicks on **OK**. This will populate the left-most column in GCL Bundler and if VIS files were found that match, the Text columns as well.

One will need to manually enter all the Descriptions such as “Fire Evac” and “Tornado Warning” into cells in this tool. If VIS files (or CSV file) are not available, one will need to enter the full text into the Text columns, perhaps by listening to the Takes. One can listen to each take by clicking on the Play button located in the column between the Take File Name and Description.

File Name		Description
Custodian.WAV		Custodian



The GLOBALCOM library file is created by clicking on the far right button **Export to Library GCL File**. The GCL file is a ZIP file that contains all the WAV files and a Takes.XML file that describes the files (contains the filename, description and text information).