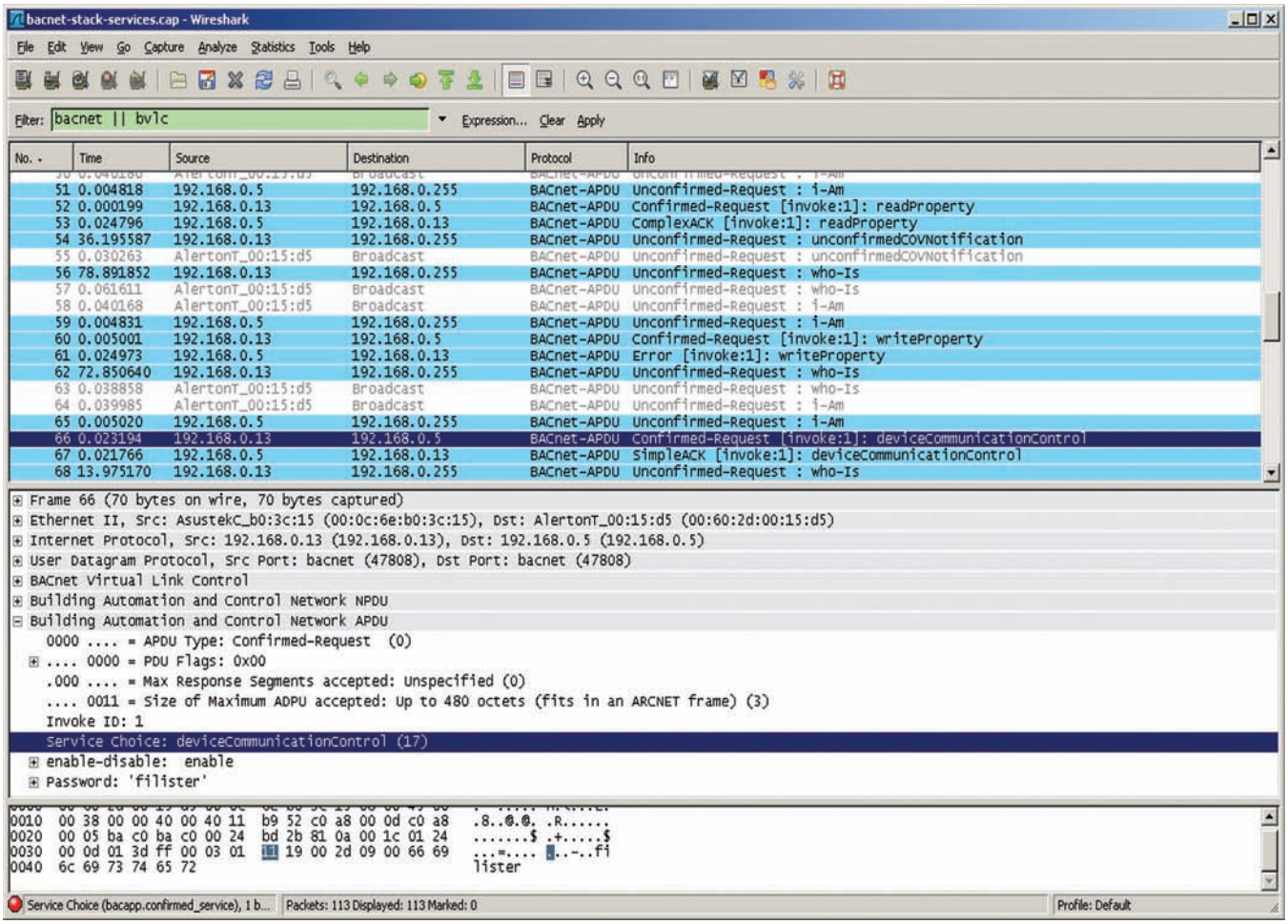

Figure 1: Wireshark displays a variety of BACnet services from various sources, and is useful for troubleshooting, developing, or learning about the BACnet protocol.

Analyzing BACnet®

By Steve Karg, Member ASHRAE

Wireshark* (*Figure 1*) is a general purpose network protocol analyzer software application that is cross-platform (runs on various computer operating systems including Linux, Windows, and Mac OS X) and open source (released under the GNU General Public License). Wireshark can be downloaded for free from www.wireshark.org.

It was created by Gerald Combs and first released to the public in 1998 under the name Ethereal. A typical open source application, Wireshark is the result of thousands of contributions by hundreds of people. Its ability to analyze BACnet and many other protocol packets stems from the efforts of many people contributing enhancements and bug fixes over the years. The Wireshark Web site includes additional documentation and tutorials, a bug tracking tool to aid in product improvement, technical support and training services, and developer information that enables developers to easily contribute to the project.

My first use of Wireshark (at that time

*Wireshark is a registered trademark of Gerald Combs.

About the Author

Steve Karg is a senior engineer at Watt Stopper/Legrand in Birmingham, Ala. He has been an active member of ASHRAE SSPC 135 (BACnet) since 2001, and convenes their Lighting Applications working group. He wrote the open source BACnet Stack at SourceForge.net®, and continues to help maintain the BACnet decoder in Wireshark.

the software was known as Ethereal) to handle BACnet decoding happened in 2005 when a customer site hundreds of miles from me was experiencing problems with a controller that stopped responding after connecting it to the BACnet network at the site. The problem occurred intermittently, but usually after several weeks of BACnet network activity. I placed a laptop PC at the site, connected the BACnet network to the laptop PC using an Ethernet hub, and set Ethereal in packet record mode. A couple of weeks later, the maintenance supervisor called me and said that the controller had stopped responding. He stopped the Ethereal packet recording operation, saved the data, and sent me the 400 MB file on a CD-ROM.

My first look at the BACnet data from the customer site using Ethereal left me desiring more detailed information. The BACnet decoding only showed BACnet Confirmed or Unconfirmed APDU messages and raw data, without naming the BACnet service or showing data names or values in the services. Understanding the nature of this open source application, I immediately set to work downloading the Ethereal protocol analyzer source code and the required libraries, reading the developer documentation, and compiling.

In only a few days, I had modified the Ethereal code to display the specific BACnet services, and submitted a patch to the Ethereal developers. The following day another BACnet patch arrived from a developer in Berlin who had reworked an earlier patch submission, which had added the majority of BACnet application decoding. The large capture from the customer site revealed that my controller was receiving bursts of 30 to 50 WriteProperty requests about every 10 seconds. I configured a similar test in my office to simulate the customer site. My controller stopped responding in a few hours, and I was able to debug my application code and correct the firmware.

Additional patches to Ethereal by me and others enhanced the property decoding and fixed a number of subtleties over the following weeks. The BACnet decoding in Wireshark continues to improve and evolve along with the BACnet standard, and today is very good.

Wireshark can monitor and decode most BACnet packets that are received primarily from an Ethernet interface. It can also receive packets from an ARCNET interface. The software is not yet able to directly capture or decode packets from a BACnet PTP (serial) or a BACnet MS/TP (EIA-485) interface. However, BACnet MS/TP can be supported by adding an external interface (*Figure 2*), which sends Ethernet SNAP protocol packets. Wireshark does not yet support BACnet Segmentation. These features will probably be added someday because someone will eventually be motivated to add the missing functionality to this highly regarded open source tool.

Wireshark can import and export packet files in a variety of network analyzer formats. The libpcap file format is the software's default file format.

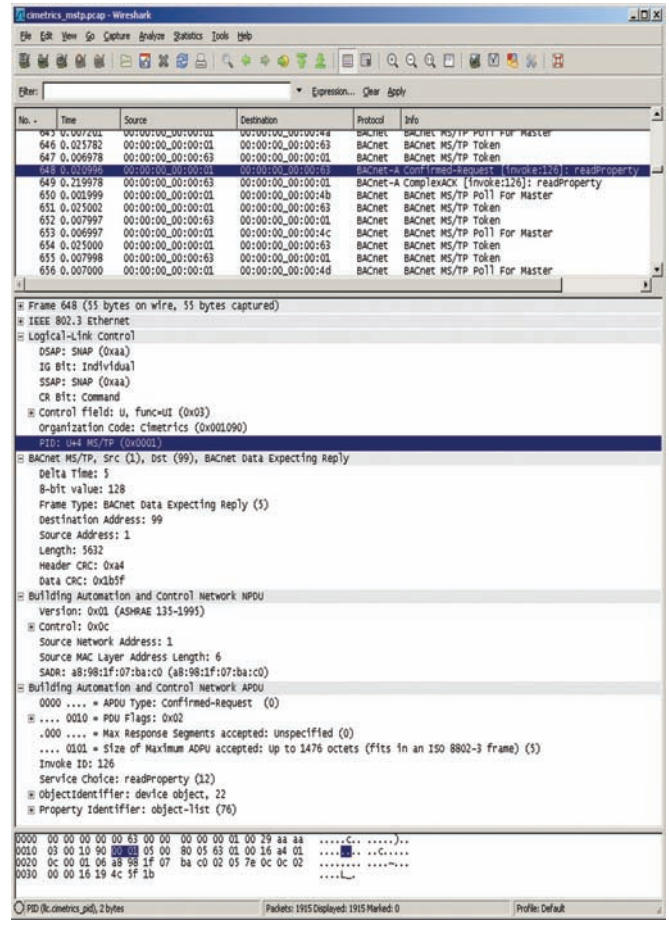

Figure 2: Wireshark and a BACnet MS/TP capture from an external interface, which sends Ethernet SNAP protocol packets.

Using Wireshark for Live Captures

To monitor or record BACnet traffic, you must be able to "see" the network traffic from the computer running the protocol analyzer. This usually requires connecting the computer and BACnet devices to an Ethernet hub, as unicast traffic between devices may not be seen on all ports of an Ethernet switch (bridge). Ethernet switches may be used if they have the ability to span, monitor, or mirror all port traffic and send it to a single port. The computer network interface must also support promiscuous mode, where the interface supplies the protocol analyzer with all the network packets it sees.

Selecting the network interface to monitor or capture is accomplished through the Capture menu options, under Interfaces or Options. The Capture Options dialog (*Figure 3*) offers the selection of a capture interface, optional display of packets in real time or automatic scrolling, MAC, network, or transport name resolution, and the ability to save a file or multiple files while capturing. The Options dialog also provides the ability to limit the capture by providing "Stop Capture" options after a number of packets, megabytes, or minutes.

Wireshark supports capture and display filtering, and the syntax for a capture filter and a display filter is different. A capture filter limits the packets captured to a couple of specific header fields. The capture filter expressions can include a specific protocol (ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp, udp), a direction (src, dest, src and dest, src or dest), and

logical operations (nor, and, or). These filter expressions can be used on a BACnet/IP network to filter out any non-BACnet/IP traffic. A common capture filter to only capture standard BACnet/IP packets would be "`udp port 47808`".

Before we start our capture, we should consider our Display Options. Wireshark can display the packets in real-time and can automatically scroll the packets while they are being received, which is fun to watch, but not as useful when trying to see specific packets. The latest builds of the software enables automatic scrolling during a capture by selecting the last packet on the display. The software also permits hiding of the Capture Info dialog box that summarizes the count and type of packets captured.

Another option to consider is Capture File(s). Although Wireshark can save a capture or portions of a capture after it is displayed, sometimes it is necessary to capture over a longer period of time. Wireshark has the ability to capture to a file or even multiple files. The software automatically appends unique identifiers to files when multiple files are used. The file or files can be limited to size, time or number of files, or can save indefinitely, only limited to the amount of disk space available. When I am capturing for days, weeks, or months, I normally disable the Display Option for "Update list of packets in real time" to limit the amount of display memory used.

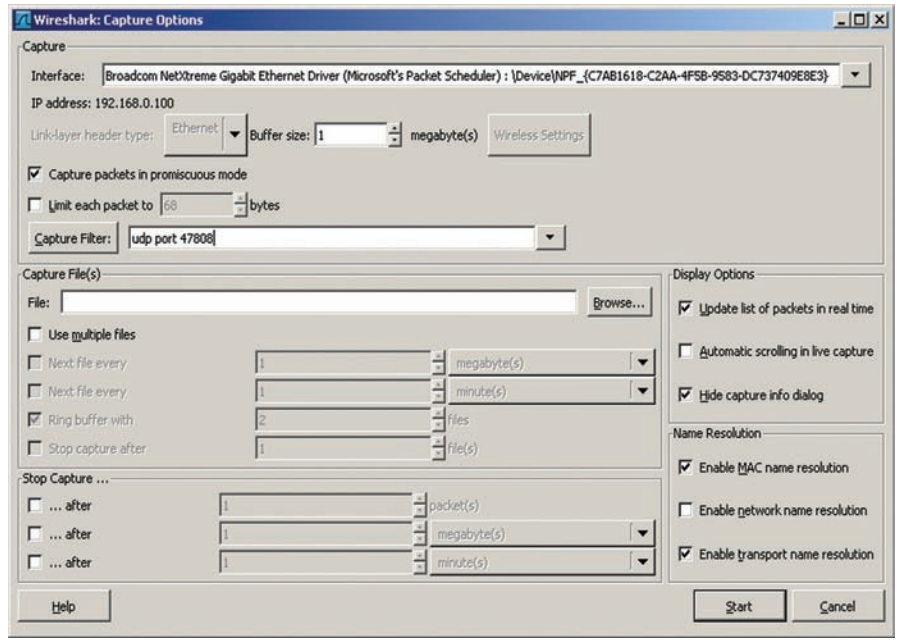

Figure 3: Wireshark Capture Options dialog box allows control of the capture display, name resolution, capture files, capture interface, capture filter, and the ability to stop after so many packets, bytes, or time.

Selecting "Start" from Capture Options begins the capture. Selecting "Stop" from the "Capture" menu stops the capture. The capture can be saved to a file or discarded.

Display Filtering

A common BACnet configuration issue that I have seen during the commissioning of a BACnet network is due to some device or workstation writing to a Present_Value property of an object at a higher priority than it should. This usually makes an output point unresponsive to other network applications. Finding the writer of the high priority usually involves some forethought at the end-device, or just some monitoring with Wireshark. Sifting through many hours or days of data packets is made easier by using the display filtering that is built into the software.

The display filter (*Figure 4)* allows for protocol specific filtering, during the live capture, after the capture has been stopped, or after opening a capture file in the display. It can be built using the Wireshark Filter Expression wizard, by typing specific values into the filter box, or by using the context menu. Each protocol decoder has its own expression words. The BACnet decoders are currently defined as BVLC (BACnet Virtual Link Control or BACnet/IP specific decoding), BACnet (BACnet NPDU or network layer decoding), and BACapp (BACnet APDU or application layer decoding). A common display filter to view all BACnet messages containing an NPDU would be "bacnet." A display filter to view only BACnet WriteProperty service packets would be "`bacapp.confirmed_service == 15`," where 15 is the service choice for the WriteProperty service of a BACnet APDU Confirmed Request.

Many, but not all, of the decoded variables are available for use in a display filter. The name of the variable can be found by first selecting the line of the variable decoding, and seeing the text, such as "bacapp.confirmed_service," shown in parenthesis in the lower left corner status box of the Wireshark window. Display filters (*Table 1*) can be applied to a data capture by using the "Apply as Filter" context menu in Wireshark. The context menu can be activated by selecting a particular packet or data element of a packet, and using the context button on the mouse (e.g., the right button on a Windows mouse).

The packets displayed using a display filter can also be saved as a separate file using "Save-As" from the "File" menu. This is useful when working with large capture files.

Some BACnet installations use BACnet/IP with a UDP port other than 47808 (0xBAC0). The Wireshark context menu allows for decoding BACnet messages on other UDP ports using the "Decode As..." option (*Figure 5)* and selecting BVLC.

The default Wireshark time display format is seconds since the beginning of the capture. Other useful formats include seconds since the previous packet, and time of day. Being able to change the reference or start time for the time display is use-

ful when checking for bandwidth issues. I monitored a college campus Ethernet network port using a hub connected to a BACnet router, which also connected to a 156K ARCNET segment. I found, using the software, that the campus network was producing thousands of BACnet UnconfirmedCOV (change of value) broadcast packets per minute, and this was saturating the 156K ARCNET segment and causing slow response and retransmits from the devices on the ARCNET segment. Adjusting the offending objects COV_Increment property around the campus eliminated the issue.

Developing BACnet Products

Wireshark is well known for being used in software and communications protocol development, and its use during BACnet software development is no exception. Timing information, the raw bytes on the wire, and the decoded meaning of the bytes on the wire provide immediate feedback to a BACnet developer. The Wireshark source code may also provide deeper insight into how particular packets or properties may be formed and decoded.

Wireshark, accompanied by a BACnet testing tool, is a common setup for testing and validating a BACnet device. The software can provide the evidence (a capture) from a testing session, which can be used by the BACnet developer to find and fix a problem discovered during testing.

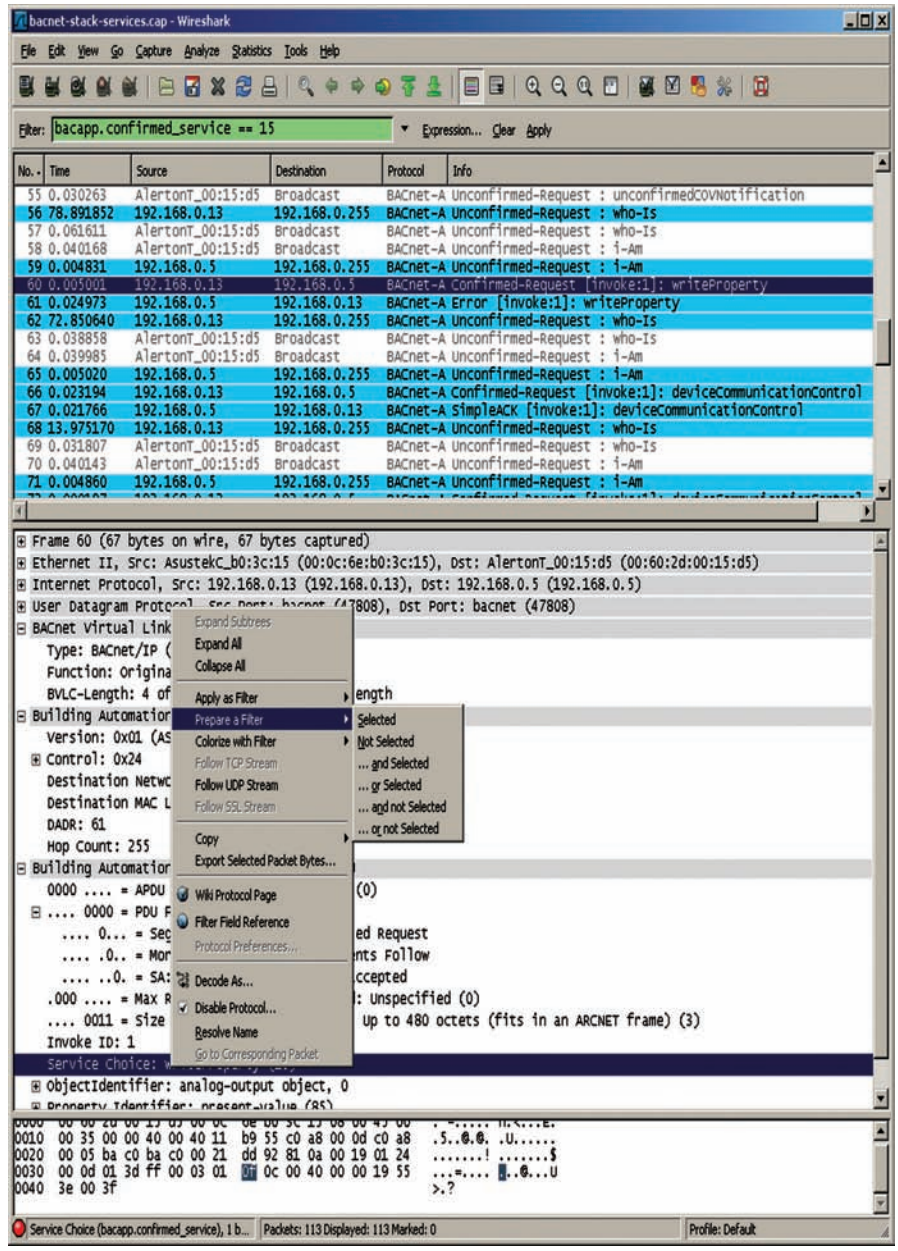

Figure 4: Wireshark default display, showing the ability to prepare or apply a display filter.

Wireshark From the Command Line

Wireshark can also be used from the command line (Command Prompt under Windows; shell or console under Linux or Unix). The first tshark command line option that I normally use is "-D", which provides me an enumerated list of interfaces that I can use for capturing. Then I use the "-i 2" or whichever interface number I want to capture from as the command line option to tshark. The "-w filename.cap" command line option is useful when I want to capture to a file, but don't necessarily want to launch the Wireshark GUI. The amount of disk space or number of files stored during a capture can be controlled using the "-b" command line option followed by duration, file size, or number of files options. More details about the specific options can be found by asking tshark for help using the "--help" command line option.

Other command line tools are shipped with Wireshark including:

- dumpcap—captures network traffic but does not display the network traffic;
- capinfos—reads a saved capture file and displays statistics about that file;
- editcap—edit and translate the format of a capture file;
- mergecap—merges multiple capture files into one file;
- text2pcap—generates a capture file from an ASCII hexadecimal dump of packets; and
- rawshark—a tool for displaying packets and specified fields from a capture file.

Free to Download, Use, and Share

It is not always convenient for me to travel to a site that is experiencing BACnet-related network problems so that I can analyze and solve the problem. I was asked to help diagnose an installation where two BACnet vendors were unable to communicate using BACnet/IP. I asked for a Wireshark capture of the communication attempts. They did not have Wireshark installed on their laptop computers, so they went to the Wireshark Web site to download the free tool. I guided them through the capture process after successful installation of the tool, and they emailed me the results. They also shared Wireshark with the local site technicians who would be able to perform Wireshark network analysis and send us the captures.

I pondered the WhoIs and I-Am requests from the capture for a clue into the problem, but both requests appeared to be well formed. Although both devices were using the standard BACnet/IP port number 47808, I did notice that each was using a different IP broadcast address. Further investigation revealed that the site was using an unusual IP subnet mask for their control systems on the network: 255.255.254.0. The BACnet device using an older GNU/Linux operating system was unable to form the correct IP broadcast address by using just the IP subnet mask. The vendor changed the network configuration script in their device to calculate and set the broadcast address directly. The local technicians installed the patch, performed some Wireshark network analysis, and sent us the captures that showed that the broadcast problem was solved.

Summary

Wireshark is software that understands the detailed structure of many different networking protocols, including BACnet. The software is able to capture, display, and save the various BACnet messages, services, attributes, and properties along with their meanings. It has powerful display filters that can be used to selectively highlight and color packet information. The software runs on most computer platforms, whether graphical or command line. It can import and export a variety of network analyzer file formats. Best of all, Wireshark is free!

Capture Filters	
`udp port 47808`	BACnet/IP packets on UDP port 47808
`udp port 47808 or udp port 47809`	BACnet/IP packets on UDP port 47808 or 47809
Display Filters	
`bvlc || bacnet || bacapp`	BACnet packets
`bacnet`	BACnet NPDU packets
`bacnet.mesgtyp`	BACnet Network Layer (router) packets
`bvlc`	BACnet/IP packets
`bvlc.function == 0x0b`	BACnet/IP Broadcast packets
`bacapp`	BACnet APDU packets
`bacapp.confirmed_service == 12`	BACnet ReadProperty packets
`bacapp.confirmed_service == 15`	BACnet WriteProperty packets
`bacapp.unconfirmed_service == 0`	BACnet I-Am packets
`bacapp.unconfirmed_service == 8`	BACnet WhoIs packets
`bacapp.unconfirmed_service == 2`	BACnet UnconfirmedCOVNotification packets

Table 1: A list of commonly used Wireshark filters for BACnet. Display filter expressions can be combined using logical operators such as "and", "or", "xor", and "not". A display filter expression can use a variety of comparison operators such as "==", "!=", ">", "<", ">=", and "<=". Additional details can be found in the Help Contents included with Wireshark.

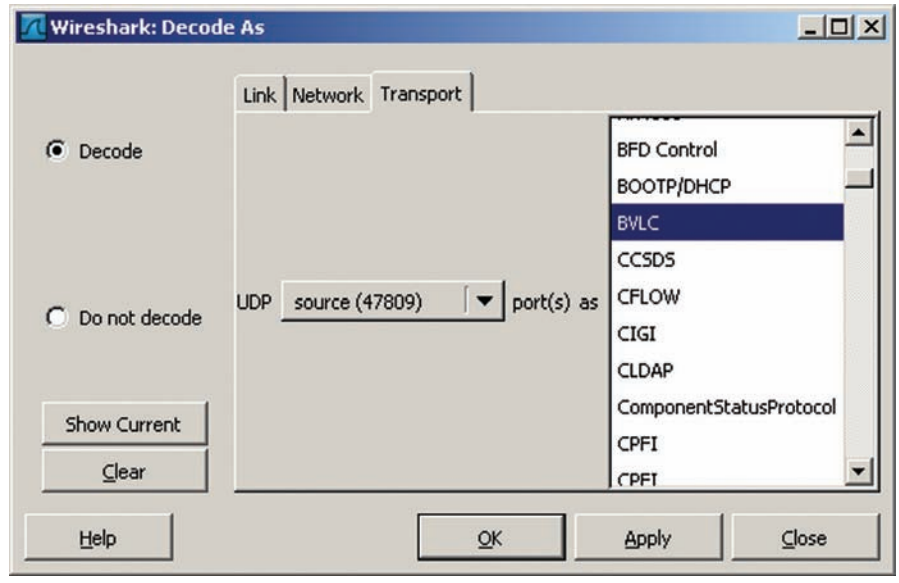

Figure 5: Wireshark "Decode As" dialog showing BACnet/IP port 47809 decoded as BVLC.