

Multiple Messages Monitor

Diagnostic aid

Thu, Dec 19, 2002

This note concerns a scheme for monitoring the success of combining multiple Acnet messages into a single datagram. Each Acnet datagram is monitored for correctness by scanning each message within to see whether it has been properly constructed.

Checks of Acnet header that can be made by monitoring the following fields:

message type	& 0xE can only be 0, 2, 4. 0x0200 signifies cancel.
dest node	if REQ/USM, may be native or acnet node#, range 0x0500–0x10FF.
source node	if REQ/USM, must match local Acnet node#
task name	Anything possible
task-id	Anything ok
message id	Anything ok
message length	Restricted to remaining datagram size, but ≥ 20 .

The actual destination node depends upon the message type. Whatever it is, it should be the same for all messages contained within the same datagram. The source node also depends upon the message type and must be the same for all messages. The task name may be different among multiple messages. The task-id may also be different.

The message-id may be anything. But the message length must be even and more than 18 bytes. It also must not imply that the message extends beyond the size of the datagram.

To find where such frames exist within the transmit buffer, follow the transmit parameter list structure. It includes a pointer to the start of the frame and also the datagram (or frame) size. After the frame header, and also the IP header, check the UDP header for the source port# used for Acnet, 6801. This identifies an Acnet header-laden message.

What should a data stream record look like, if that were used for logging errors? Its name might be "AMSGLOG ". Each record would be 16 bytes in length, as follows:

<i>Field</i>	<i>Size</i>	<i>Meaning</i>
tNode	2	target node from Acnet header
dgSize	2	datagram length
nMsg	2	#message within datagram
iError	2	error code
cDate	8	date-time of occurrence

It will also be important to capture the datagram contents in order to help analyze what went wrong. Perhaps some reasonable-size buffer can be used to hold the most recent datagram in which an error was found. Ultimately, there should be no such errors, of course.

If a fragment is detected, it should not be analyzed, since the first one will be seen to be truncated, and any other one will not likely begin with an Acnet header. On the other hand, the entire datagram is first built in the transmit fame buffer before the first fragment is sent out. And to facilitate building the subsequent fragments, the rest of the datagram beyond the first fragment is skipped. this means that the entire datagram is actually in memory, so it is conceivable it could be checked. But knowing the full datagram size, before fragmentation, may not be so easy.