

Big Save Timing

Client vs server

Mon, Mar 4, 2002

Included with Big Save statistics posted on the Web is timing data that purports to analyze the response time of each front-end that contributes data to be logged. The slowest and fastest response times are noted, along with the average response time over all Acnet packets used in the requests sent to that front-end. Each front-end is processed by a separate thread, so that within that thread, a request is made, a response is received, and the next request is made, etc. The client thread measures the response time of each request from its point-of-view. The posted results often seem incredible, at least in the case of the Linac and IRM front-ends. The objective of this note is to find a way to make sense out of the often unbelievable statistics posted on the Web about front-end performance during Big Save operations.

From the client point-of-view, for each front-end that is targeted, a request is sent, and the response time of the corresponding reply is logged. But one can also look at it from the front-end point-of-view. When a response is returned to the client, there will be a time that can be measured before the next request, if any, is received. Analyzing this timing, along with the timing between its receipt of the request message and the subsequent transmission of the reply, may provide a better clue about what is happening during Big Saves.

Focusing on Big Saves of Linac data, almost all of it is obtained via the server node0600. All together, 223 packets are sent to this node during every Big Save. This entire operation typically takes about 25 seconds or so. These two numbers together might imply an elapsed time of about $25000/223 = 110$ ms/packet. Nearly all of this time cannot be attributed to the front-end response time, given an understanding of how the Linac front-ends actually perform. The worst-case response time to such requests is expected to be about 20 ms, with a typical response time of 4 ms or less. This assumes that each request packet datagram arrives in a single ethernet frame; i.e., it is not fragmented. The server node that received the request must, of course, forward that same request to the target node(s) that can actually supply the requested device data; the replying node(s) must deliver the reply device data; and the server node must forward the composite reply data to the client. If there are no delays, all this can happen in 4 ms. (It helps that the Linac front-ends use 100 MB ethernet.) The reason why a delay of 16 ms, say, could occur has to do with exactly when the request arrives at the server and is forwarded to the target node(s). Each of the target nodes is busy during the first part of the 15 Hz cycle, which begins 3 ms after the Booster reset clock event. Most of this busy time is spent awaiting replies from the SRMs via which most hardware I/O is actually interfaced.

One diagnostic available in the server node, as well as in any of the Linac nodes, is a log of the Acnet RETDAT request packets as they are received. Information logged shows the arrival time (request message processing), the expected reply length, the number of devices requested, the FTD, and the message-id from the Acnet header. If desired, this log can be enabled to log only requests coming from a single node.

Another available diagnostic is provided by the FMON local application. It can log a selected 56 bytes of each datagram communicated between a given node and the local node. With no offset specified, this includes 18 bytes of information identifying the IP addresses and sockets, the 18-byte Acnet header, and the first 20 bytes of the request message, which nearly includes the first device requested. Of course, the time of the reception or transmission is also logged.

Given that we can find out which client node performs the Big Save, either diagnostic can be configured to note the time of request processing. The first can only show request arrivals; the second can show traffic both ways between the client and server nodes.

A third diagnostic captures all network transactions, but there is far too much of this occurring to analyze something that covers an elapsed time of 25 seconds. From these descriptions, it seems clear that the FMON tool will be most useful for this analysis.

Unfortunately, it turns out that the node performing the Big Save is `DAE04.fnal.gov`, but it does not do it directly, but rather indirectly via `DAE05.fnal.gov`, which is also used for any data requests made to Linac nodes. This includes data logging one-shot requests, which are numerous. There are 15 Hz replies occurring 24/7 from `node0600` to this node. Those could be filtered out, since they would be multiple reply messages. But there are still many one-shot requests occurring very often that cannot be easily distinguished from Big Save one-shot requests.

A smarter monitoring LA might detect one-shot request messages that ask for a certain number of devices, such as 75. Then it could show the one-shot replies for only those messages. In practice, one could catch most of it by looking for replies that are one-shot and that are also at least 300 bytes in size, since $4 \times 75 = 300$. Can this be an option for `FMON`? One option could select only one-shot requests or replies. The other option could be Big Save specific, capturing only requests that specify 75 devices and replies that are at least 300 bytes long. Maybe the number of devices could be specified.

The `TR_FLAG` parameter at present uses only the least significant two bits in a word to select datagrams received or transmitted. Another bit could specify one-shot requests/replies only. Another could specify only `RETDAT`, at least for the first message in a datagram. Besides that, the number of devices to be matched could be specified. Of course, all of these special cases apply only to the Acnet protocol.