# *Telnet*

The Telnet module implements the session protocol layer for the telnet protocol. It listens on the assigned TCP port for incoming connections and passes the data portions upwards to the application (usually the console process). In ROME systems, the telnet process is provided to allow remote access to machines. As such the process only handles one incoming connection at a time, and does not support locally-originated sessions. The process strips telnet-layer options from the data and otherwise ignores them.

## Process Information

| | |
|---|---|
| Prototype Name | telnet |
| Process Priority | below driver-level |
| Process Name | should be “telnet” if the standard Console process is used. |

## Process Operation

The module has a queue handler and a main process, and accepts the messages defined in the *Standard* messageset for dataflows into and out of the TCP layer with the following processing:

| | |
|---|---|
| CLOSE | messages are handled in the main process and where any outstanding *F/GETMBLK* messages on the (upstream) file are returned, and the downstream file is closed to initiate transport layer termination. |
| FETMBLK/GETMBLK | messages are added to the file’s queue of pending input requests in the queue handler. GETMBLK replies contain variable amounts of data receiver over the network interface. Any telnet-specific options are removed in the main process and the rest of the data are passed up as replies to *F/GETMBLK* messages. |
| FLUSH | messages and replies are passed transparently between the upstream and TCP layers, to cause a TCP-level flush. This is necessary, for example, to generate prompts without trailing newlines. |
| NEWMBLK | messages are passed down to the TCP layer, and the replies passed back up from within the queue handler. The telnet layer does not add further protocol overheads to the data stream. |
| OPEN | messages are handled in the main process where a new data structure is created for each open file and a pointer to it is returned as the *dest_context* field in the reply. |

| | |
|---|---|
| OUTMBLK/PUTMBLK | messages are passed downstream and the replies passed upstream from the queue handler. |
| RETMBLK | messages are generated internally for buffers that contain no user data (for example TCP options) and are flagged by having the address of the *tcp_file* as their source context. Otherwise the messages and their replies are passed respectively downstream and upstream from within the queue handler. |