UDP

The UDP module implements the layer 4 User Datagram Protocol of the Internet Protocol. It processes UDP headers on behalf of applications passes packet between the application and the IP layer. The module uses the data types and definitions from the *ip.h* header file in the IP module. The module itself is machine independent and does not require any configuration.

Process Information

Prototype Name	udp
Link Order	does not matter
Process Name	“udp”

Process Operation

Active and Passive Opens

The UDP layer distinguishes between ‘listener’ dataflows that are open to receive any packets on that port, but are not bound to a remote IP address, and ‘connected’ dataflows which represent a point-to-point connection. The type of the dataflow is determined at *OPEN* time. Listener dataflows are created by sending an open request to the UDP layer with no destination IP address. This is termed a passive open. In order to transmit data packets, the UDP layer must have an active dataflow. Unlike TCP, UDP flows cannot change from passive to active since there is no concept of a connection state. In general, packets are initially passed upwards on a listener data flow, however if there is an active to the same incoming port, packets are preferentially passed upwards on that dataflow.

Errors

The UDP code generates ICMP ‘port unavailable’ messages if there is no dataflow on which to pass packets upwards. This is essential for the ‘traceroute’ protocol, which uses this error condition to terminate its probe.

Messages

The module accepts the *Standard* messageset with the following processing.

CLOSE messages are handled in the main process. Any pending read requests are returned to the sender and the local context data structure is unlnked and freed.

FETMBLK/GETMBLK messages are added to the queue of input requests for the dataflow in the queue handler. Replies to GETMBLK messages from the lower layer are passed up as replies to these messages. The module first tries to match a dataflow that was actively opened, and if there is no matching dataflow, tried to find a listener for the incoming port. The buffer is returned with the read pointer set just after the end of the UDP header.

FLUSH messages are passed downstream (it is not expected that these will be used over the UDP protocol stack). The replies are passed back upstream.

NEWMBLK messages are processed within the queue handler. The message is passed downstream after increasing the request size to allow for the UDP header. The reply is passed back upwards after the header space is reserved at the start of the buffer, unless the request was generated internally, which is signalled by placing the address of the *ip_link* file in the source context of the message.

OPEN messages are handled in the main process. The *ipaddr* field of the URL determines if this is an active or passive open (see above) and the *port* field contains the UDP local port number. By the time a dataflow reaches UDP, the destination machine's IP address should have been resolved (for example using the DNS or NIS services). The UDP process does not look at the *host* field in the URL, only the *ipaddr* field.

OUTMBLK/PUTMBLK messaages of zero length are passed downstream from the queue handler. Otherwise messages have the UDP header added to the message from the main process, and the UDP checksum calculated, before being passed downstream. Replies are passed upstream unless the message is a reply to an internally generated request (only used for ICMP error messages).

RETMBLK messages are passed downstream to free the buffer. When the reply is received, and linked message blocks are used to set up new receives (for fragmented packets), before the reply is passed upstream.

Shared Library Macros and Routines

udp_new_port

int *udp_new_port*(**void**)

The *udp_new_port* routine returns a TCP port number outside the range of reserved ports that is not otherwise in use within the machine.

udp_stats

UDP_STATS **udp_stats*(**void**)

The *udp_stats* routine returns a pointer to the statistics structure maintained by UDP, and defined in *ip.h*. The only statistics currently maintained are the number of packets discarded by the protocol.