PORTQUEUE

Port queuing can be established, modified, and turned-off by means of the PORTQUEUE command.

Syntax: `PORTQueue PORT=`*`port`* `[,TIMEOUT=`*`sec`*`] [,DEPTH=`*`num`*`]`

Arguments:

PORT= - The port number for which queuing is to be established or modified. Values range from 1 to 65535.

TIMEOUT= - This parameter specifies how long a queued connection may wait for service before it is discarded. Values may range from 1 to 60 seconds. If not specified, the stack will assign a value of its choice for a new specification and leave existing values as found.

DEPTH= - The maximum number of connections that may be queued at any given time. Once this number is reached, additional incoming requests are refused. This value may range from 0 (no queuing) through 100. If not specified, any existing value will remain unchanged.

Example:

```
IPN237I portqueue port=4099, timeout=5, depth=10
IPN405I Port queue values successfully set
```

Exposition: Standard TCP/IP processing is simple in concept. Each connection consists of a series of interactions between two applications, in the following order:

- Application A issues a Passive OPEN (listen)
- Application B issues an Active OPEN
- The applications issue optional SENDs and RECEIVEs, as appropriate
- Each application issues a CLOSE

In general terms, the application that issues the Passive OPEN (listen) is considered to be the "server" and the application issuing the Active OPEN, the "client".

For illustration, consider a web server. It issues its "listen" on well-known port 80. When a browser (client) wishes to obtain a web page, it issues an active OPEN to port 80 and sends its request. Once the server fills the request, the connection is closed and the server re-issues its "listen" to await the next client's request.

In practice, the difficulty lies in coping with connection requests that occur when the server is already involved with processing another request, i.e., when there is no "listen" in effect.

The rules that govern the stack are quite explicit and require it to forcefully reject (RESET) any connection request that cannot immediately be paired with an existing listen connection.

The most common way of ensuring that all (most) connection requests are successful, is for the server to maintain multiple listen connections on the same local port. As each of these Passive OPENs completes, the server immediately replaces it with another. Processing of the established connections can overlap to any degree desired.

This level of programming is difficult and still permits some requests to be "lost" due to processing delays at the application level. It is even more difficult to provide the ability to change the depth of

queuing without requiring modifications to the program's code.

The PORTQUEUE command eliminates the needs for most multi-threading in applications.

To use the Port Queuing Facility, one must first determine the port number on which connection requests are to be queued. For example, a web-server application would choose well-known port 80.

Once the port is designated as eligible for queuing, incoming connection requests that cannot be paired with already existing listen connections will be assigned to a "blank" listen connection, automatically provided by the stack. This connection is negotiated by proxy with a closed window. When the server eventually issues a "listen", the next opened-by-proxy connection is assigned, the window is opened to its normal value, and processing commences in the expected manner.

Related Commands:	QUERY PORTQUEUE	-	Displays statistics associated with queued connection requests.