                            PRE-EMPTION

In circuit-switching systems, once a user has acquired a circuit, the
communication bandwidth of that circuit is dedicated, even if it is not
used. When the system saturates, additional circuit set-up requests are
blocked. To allow high precedence users to gain access to circuit
resources, systems such as AUTOVON associate a precedence with each
telephone instrument. Those instruments with high precedence can
pre-empt circuit resources, causing lower precedence users to be cut
off.

In message switching systems such as AUTODIN I, incoming traffic is
stored on disks (or drums or tape) and processed in order of
precedence. If a high precedence message is entered into the system, it
is processed and forwarded as quickly as possible. When the high
precedence message arrives at the destination message switch, it may
pre-empt the use of the output devices on the switch, interrupting the
printing of a lower precedence message.

In packet switching systems, there is little or no storage in the
transport system so that precedence has little impact on delay for
processing a packet. However, when a packet switching system reaches
saturation, it rejects offered traffic. Precedence can be used in
saturated packet switched systems to sort traffic queued for entry into
the system.

In general, precedence is a tool for deciding how to allocate resources
when systems are saturated. In circuit switched systems, the resource
is circuits; in message switched systems the resource is the message
switch processor; and in packet switching the resource is the packet
switching system itself.

This capability can be realized in AUTODIN II without adding any new
mechanisms to TCP (except to make precedence of incoming connection
requests visible to the processes which use TCP). To allow pre-emptive
access to a particular terminal, the software (i.e., THP) which supports
terminal access to the TAC can be configured so as to always have a
LISTEN posted for that terminal, even if the terminal has a connection
in operation. For example in the ARPANET TENEX systems, the user TELNET
permits a user to have many connections open at one time - the user can
switch among them at will. To the extent that this can be done without
violating security requirements, one could imagine a multi-connection
THP which always leaves a LISTEN pending for incoming connection
requests. If a connection is established, the THP can decide, based on
its precedence, whether to pre-empt any existing connection and to
switch the user to the high precedence one.

If the user is working with several connections of different precedence
at the same time, the THP would close or abort the lowest precedence

Cerf   [Page 1]

connection in favor of the higher precedence pre-empting one. Then the
THP would do a new LISTEN on that terminal's port in case a higher
precedence connection is attempted.

One of the reasons for suggesting this model is that processes are the
users of TCP (in general) and that TCP itself cannot cause processes to
be created on behalf of an incoming connection request. Implementations
could be realized in which TCPs accept incoming connection requests and,
based on the destination port number, create appropriate server
processes. In terms of pre-empting access to a remote terminal,
however, it seems more sensible to let the process which interfaces the
terminal to the system mediate the pre-emption. If the terminal is not
connected or is turned off, there is no point in creating a process to
serve the incoming high precedence connection request.

For example, suppose a routine FTP is in operation between Host X and
Host Y. Host Z decides to do a flash-override FTP to Host X. It opens
a high precedence connection via its TCP and the "SYN" goes out to the
FTP port on Host X.

FTP always leaves one LISTEN pending to pre-empt lower precedence remote
users if it cannot serve one more user (and still keep a LISTEN
pending). In this way, the FTP is naturally in a state permitting the
high precedence connection request to be properly served, and the FTP
can initiate any cleaning up that is needed to deal with the
pre-emption.

In general, this strategy permits the processes using TCP to accommodate
pre-emption in the context of the applications they support.

A non-pre-emptable process is one that does not have a LISTEN pending
while it is serving one (or more) users.

The actions taken to deal with pre-emption of TCP connections will be
application-process specific and this strategy of a second (or N+1st)
LISTEN is well suited to the situation.

Pre-emption may also be necessary at the site initiating a high
precedence connection request. Suppose there is a high precedence user
who wants to open an FTP connection request from Host Z to Host X. But
all FTP and/or TCP resources are saturated when this user tries to start
the user FTP process. In this case, the operating system would have to
know about the precedence of the user and would have to locally pre-empt
resources on his behalf (e.g., by logging out lower precedence users).
This is a system issue, not specific only to TCP. Implementation of
pre-emption at the source could vary greatly. Precedence may be
associated with a user or with a terminal. The TCP implementation may
locally pre-empt resources to serve high precedence users. The
operating system may make all pre-emption decisions.