**TCP - Transmission Control Protocol (Client uses Telnet to log into Server)**

<table>
<thead>
<tr>
<th>Client Node</th>
<th>Internet</th>
<th>Server Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>Telnet</td>
<td>Net</td>
</tr>
</tbody>
</table>

**EventHelix.com/EventStudio 1.0**

09-May-02 23:39 (Page 1)

Copyright (c) 2002 EventHelix.com Inc. All Rights Reserved.

**Listen on Telnet Port (23)**

Telnet server listens to TCP port 23

**User initiates Telnet**

User types `telnet <IP Address>`

User invokes the telnet application

Telnet establishes TCP connection with the requested server

Telnet server is designed to handle multiple telnet sessions. Thus when a telnet connection request is received, it creates a thread to handle the individual telnet session. This way the telnet server is free to listen to port 23 for more telnet requests.

The Telnet thread listens to the socket assigned for the telnet session.

Telnet notifies the user that it has established the TCP connection with the destination machine.

**Negotiation of Terminal Options**

The communication between client and server is handled with internal commands, which are not accessible by users. All internal TELNET commands consist of 2 or 3-byte sequences, depending on the command type. The negotiation takes place using such commands. Commands begin with the Interpret As Command (IAC) character. IAC is defined as 255. When IAC is received in a telnet stream, the receiver interprets the next one or two bytes as command.

Telnet uses "will", "won't", "do" and "don't" commands to negotiate options between the client and server. "Will" shows desire to use, or confirmation of using, the option indicated by the code immediately following. "Won't" shows refusal to use or continue to use the option. "Do" requests that other party uses, or confirms that you are expecting the other party to use, the option indicated by the code immediately following. "Don't" demands that the other party stop using, or confirms that you are no longer expecting the other party to use, the option indicated by the code immediately following.

**do_SUPPRESS_GO_AHEAD**

Half duplex terminals operated with only transmission in one direction. Thus a go ahead signal was used to signal to the other end that it can transmit. Modern terminals are full duplex so they can work without a go ahead character. "Suppress go ahead" requests the other end to not use the go ahead signal.

**will_TERMINAL_TYPE**

Telnet client informs the server that it wishes to identify terminal type.

**do_TERMINAL_TYPE**

Server responds with the indication that it can handle terminal type identification.

**SB_REQUEST_TYPE_SE**

Server requests terminal type. SB and SE commands enclose the sub-negotiation bytes.
### Transmission Control Protocol (Client uses Telnet to log into Server)

<table>
<thead>
<tr>
<th>Client Node</th>
<th>Internet</th>
<th>Server Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>Net</td>
<td>Telnet</td>
</tr>
<tr>
<td>User</td>
<td>Telnet</td>
<td>Telnet Server</td>
</tr>
<tr>
<td>Net</td>
<td>Server</td>
<td>Telnet Thread</td>
</tr>
<tr>
<td>EventHelix.com/EventStudio 1.0</td>
<td>09-May-02 23:39 (Page 2)</td>
<td></td>
</tr>
</tbody>
</table>

**Client informs Server that it is using a VT220 terminal**

**Server also signals that it will supress go ahead**

**Server says it is wishes to use echo**

**Client informs server that is will support echo**

**Client informs Server that it will support echo**

**Client request no echo mode**

**Server creates an operating system terminal device. A shell operates with standard input and output being handled by the Telnet server's thread**

**Shell prints a welcome message to standard output**

**Telnet thread passes the text message to the other end**

**Telnet prints the message on the screen**

**Server says it won't echo**

**Shell displays login prompt**

**User enters login name**

**Login name is passed to the shell as standard input**

**Shell requests password**

**User enters password**

---

**Text is exchanged between the Telnet Server and Telnet**

**Client**

- SB_VT220_SE
- will_SUPRESS_GO_AHEAD
- dont_reply
- will_ECHO
- reply
- do_ECHO
- reply
- do_ECHO
- reply
- wont_ECHO
- reply

**Server**

- create
- Welcome_Message
- Print: Welcome Message
- wont_ECHO
- dont_reply
- Login_Prompt
- Print: Login Prompt
- Login_Name
- Password_Prompt
- Print: Password Prompt
- Password

**OutputText**

**InputText**
TCP - Transmission Control Protocol (Client uses Telnet to log into Server)

<table>
<thead>
<tr>
<th>TCP States</th>
<th>Client Node</th>
<th>Internet</th>
<th>Server Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>User logs out</td>
<td>Client</td>
<td>Net</td>
<td>Server</td>
</tr>
<tr>
<td>Telnet passes</td>
<td>Net</td>
<td>Telnet Server</td>
<td>Telnet Thread</td>
</tr>
<tr>
<td>Control-D kills</td>
<td>Shell</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

User logs out using Control-D
Telnet passes Control-D to the remote shell
Control-D kills the shell
End of child signal is received by the thread
Shell is now gone
Thread also ends after informing the Telnet server

Telnet server closes connection
Client closes connection