Laura Chappell

TCP Sequencing and Sliding Windows

Before you can effectively analyze communications on your company's network, you must understand how protocols work. After all, how can you identify what's wrong with network communications if you do not understand how these communications are supposed to work?

For example, if you are managing a TCP/IP network, you should understand how TCP (and IP for that matter) works. As a connection-oriented protocol, TCP requires an initial handshake process to establish a connection between TCP peers. (For more information about the TCP handshake, see "Inside the TCP Handshake," *Novell Connection*, Mar. 2000, pp. 34–35. You can download this article from www.ncmag.com/past.) TCP also uses two other functions to facilitate communication between TCP peers—a sequence and acknowledgment function, which ensures that packets are received in an orderly fashion, and a sliding window function, which increases overall throughput between TCP peers. This article explains how both of these functions work.

THE SEQUENCE AND ACKNOWLEDGMENT FUNCTION

The sequence and acknowledgment function guarantees that packets are ordered correctly and prevents missing TCP segments. After all, you would not want the middle of your file backup suddenly heading south while the rest of that file backup reached its proper destination.

To track TCP segments as they are sent, TCP devices include a sequence number in the TCP header of each packet they send. Each side of a TCP communication tracks its partner's sequence number to ensure that the packets arrive in the correct order.

To track this sequence number, the receiving TCP device sends its partner the sequence number that device expects to receive next. For example, if host A sends a packet with sequence number 1, host B acknowledges receipt of this packet and states that sequence number 2 is expected next. Of course, the process is more complex than this simple example, as the next few sections of this article explain.

The Initial Sequence Number

During the TCP handshake process, each partner in the TCP connection selects its own starting sequence number. Each TCP partner then increments this sequence number by the amount of data included in each packet.

For example, Figure 1 shows a packet sent from device 10.3.30.1 to device 10.3.71.7. (See p. 36.) The TCP header shows that 10.3.30.1's sequence number is 135698. By examining the Total Length field in the IP header, you can determine that the packet contains 38 bytes of data after the TCP header.

How do you figure out how many bytes of data this packet contains? The Total Length of the packet shown in Figure 1 equals 78 bytes. You then subtract the 20-byte IP header and the 20-byte TCP header.

The Acknowledgment Number

Upon receipt of the packet shown in Figure 1, device 10.3.71.7 responds with an acknowledgment. The acknowledgment packet indicates the next-expected sequence number in the acknowledgment field. (EtherPeek from Wild Packets Inc., which I used to capture the packets for this article, decodes the next-expected sequence number as the Ack number.) In this case, the next-expected sequence number is 135736, as shown in Figure 2. (See p. 36.)

When you are analyzing the sequence and acknowledgment process, keep in mind the following equation:

Sequence Number In + Bytes of Data Received = Acknowledgment Number Out

The next sequence number device 10.3.30.1 will use is 135736 (135698 plus 38).

Illustration: Norman Felchle

The following is a simplified explanation of how a sequenced communication may occur. (Remember that the acknowledgment number field contains the value of the next sequence number that is expected from the other side of the TCP communication.)

Host 1 ———>
Sequence number 1 with 9 bytes of data
Acknowledgment number field = 100

<——— Host 2
Sequence number 100 with no data (ACK)
Acknowledgment number field = 10 (in 1 + 9 bytes of data)

Host 1 ———>
Sequence number 10 with 5 bytes of data
Acknowledgment number field = 100

<——— Host 2
Sequence number 100 with no data (ACK)
Acknowledgment number field = 15 (in 10 + 5 bytes of data)

<——— Host 2
Sequence number 100 with 20 bytes of data
Acknowledgment number field = 15

Host 1: ———>
Sequence number 15 with no data (ACK)
Acknowledgment number field = 120 (in 100 + 20 bytes of data)

Did you notice when Host 2 began sending data to Host 1? (Find the point in the communication sequence when Host 2 sent two consecutive communications. The second communication included data.)

The acknowledgment number field increments only when data is received. Because the data flow can change directions (such as Host 1 sends data to Host 2 and then Host 2 sends data back to Host 1), the sequence number field on one side of the TCP communication may increment for a while and then stop incrementing. The sequence number field on the other side of the TCP communication will then begin to increment.

The communication sequence example starts with Host 1 sending data and then reverses when Host 2 has data to send. This pattern is typical of two-way communications.

Note. During the TCP startup and teardown sequence, a "phantom byte" causes the sequence number and acknowledgment number fields to increment by 1 even though no data is exchanged. This phantom byte can be confusing when you have just learned that the sequence number field increments only when data is sent.

The next section explains the sliding window, which helps reduce the ping-pong communication style of connection-oriented protocols such as TCP.

THE SLIDING WINDOW

TCP communications can send a set of packets without requiring an intervening acknowledgment for each packet sent. This capability is called a *window*.

The size of the window depends on the following factors:

- The amount of traffic allowed on the network
- The amount of TCP buffer space the receiver has advertised

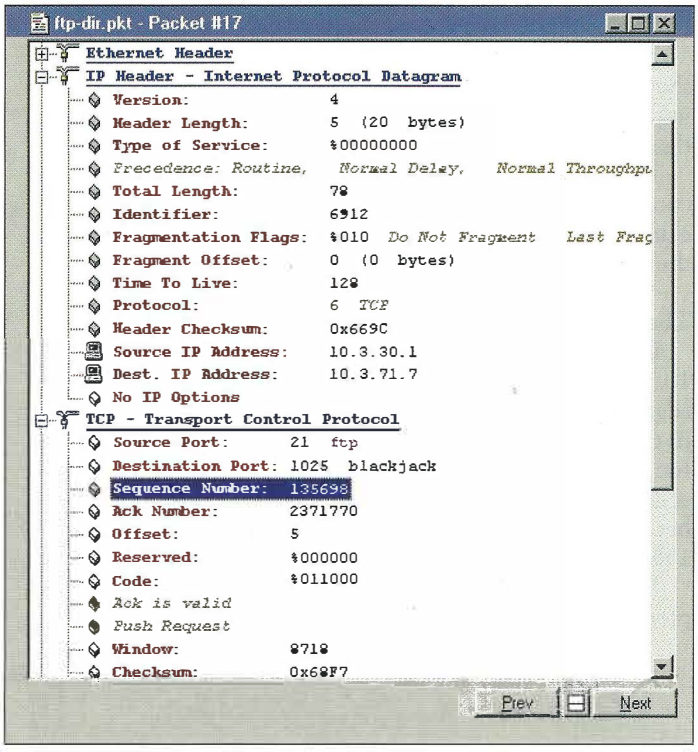


Figure 1. *The current sequence number is 135698, and the total length of the packet is 78.*

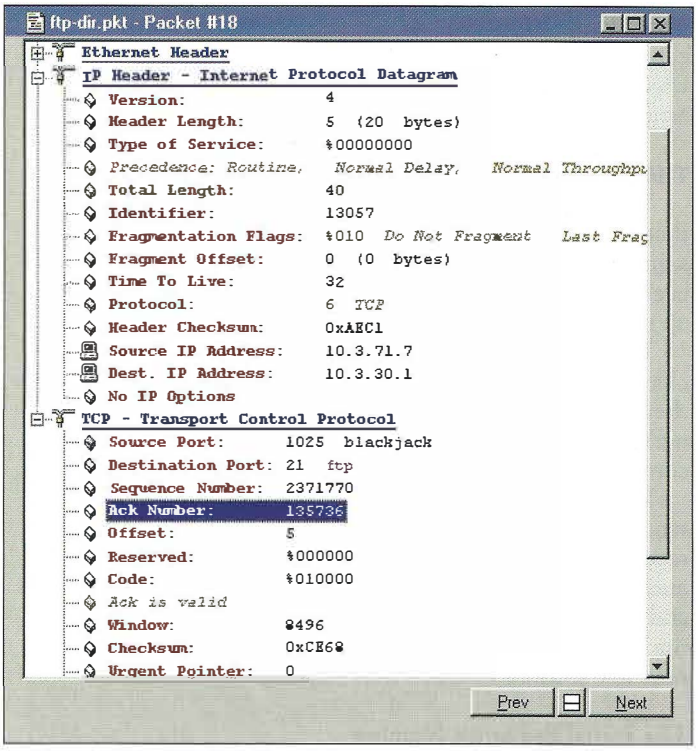


Figure 2. *The Ack number indicates the next-expected sequence number.*

The window will always be the lower of these two values.

Available bandwidth is based on data transferred without error. If a packet is lost in transit, the sender assumes the network is congested. Receiver buffer space, on the other hand, is based on the receiver's side of the communications—that is, on how many buffers the receiver has available.

Figure 3 illustrates how the window moves and works on a set of data packets labeled A through J. As you can see in Figure 3, the data set A+B has already been sent and acknowledged. The current window has sent data set C+D+E+F, and the sender is waiting for an acknowledgment.

The window will now move to the right to send the next data set G+H+I+J. The window continues to slide to the right as acknowledgments are received.

If a receiver's buffer space fills up, that receiver advertises a window of zero (0). When buffer space becomes available again, the receiver sends another packet with the new window size defined. (For more information about the TCP sliding window, read Request for Comments [RFC] 2001. To download this RFC, visit www.ietf.org/rfc/rfc2001. You may also want to read *TCP/IP Illustrated, Volume 1: The Protocols* by W. Richard Stevens [Addison-Wesley Professional Computing Series] ISBN 0-201-63346-9.)

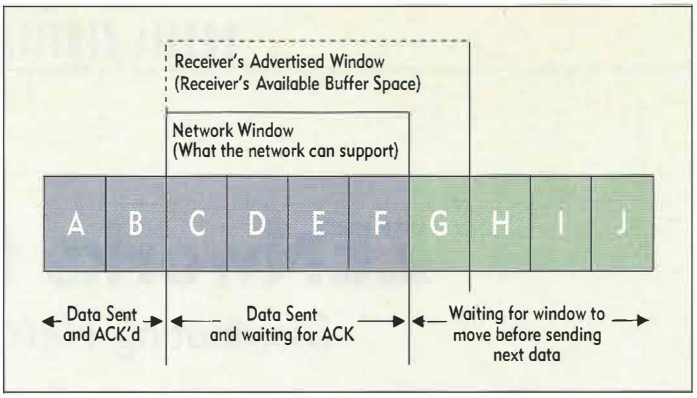


Figure 3. *The sliding window moves to the right to send more data.*

PUTTING IT ALL TOGETHER

When you examine TCP-based communications using a protocol analyzer, interpreting the sequence number and the acknowledgment number information can be confusing. Figure 4 illustrates the EtherPeek summary window and provides some pointers on how to interpret the sequence numbers shown.

In Figure 4, I have captured a portion of a data transfer process: Device 10.3.30.1 is sending some data to device 10.3.71.7. The following information is shown in the Plug-in Info column:

S = [sequence number]
L = [length after the TCP header]
A = [acknowledgment number]
W = [advertised window size]

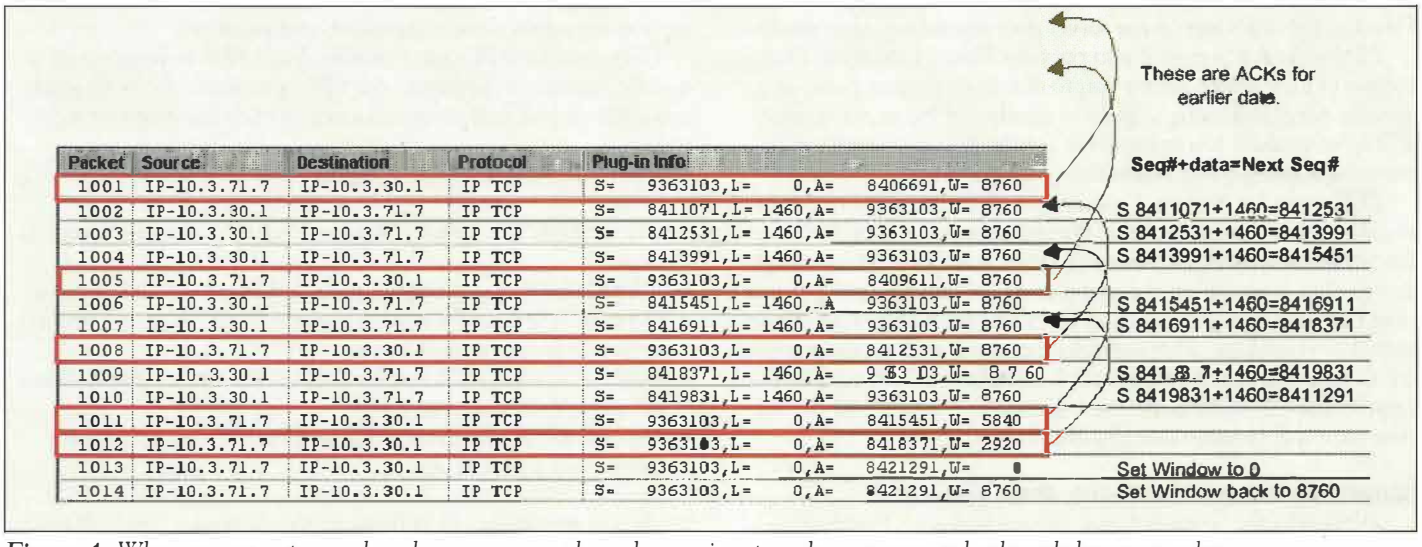
Just by looking at the length field, you can see which way the data is flowing. Packets 2, 3, 4, 6, 7, 9, and 10 contain data from device 10.3.30.1. The sequence number values are indicated on the right, and the arrows depict the correlation between acknowledgments and their acknowledged packets.

You may notice that the final packets shown indicate that device 10.3.71.7 has run out of buffer space—it has advertised a window of zero (0). After the application takes some of the data out of the receiver's buffer, the receiver advertises a larger window.

If you want to check out TCP sequencing and sliding windows on your company's network, try capturing and analyzing an FTP file transfer or HTTP web browsing session. It's helpful to know what is happening during these FTP and HTTP communication sessions.

For more information about TCP communications, read Advanced Network Analysis *by Laura Chappell, which is available as part of a four-book CD set. You can order this CD set online through www.podbooks.com.*

Laura Chappell is the Senior Protocol Analyst for the Protocol Analysis Institute. She has recently announced a hands-on network analysis workshop in Zurich, Switzerland on June 27-30 and an analysis course in Manchester, England in Q4 2001. For more information about these workshops, visit www.packet-level.com.



Packet	Source	Destination	Protocol	Plug-in Info
1001	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8406691,W= 8760
1002	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8411071,L= 1460,A= 9363103,W= 8760
1003	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8412531,L= 1460,A= 9363103,W= 8760
1004	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8413991,L= 1460,A= 9363103,W= 8760
1005	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8409611,W= 8760
1006	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8415451,L= 1460,A= 9363103,W= 8760
1007	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8416911,L= 1460,A= 9363103,W= 8760
1008	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8412531,W= 8760
1009	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8418371,L= 1460,A= 9363103,W= 8760
1010	IP-10.3.30.1	IP-10.3.71.7	IP TCP	S= 8419831,L= 1460,A= 9363103,W= 8760
1011	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8415451,W= 5840
1012	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8418371,W= 2920
1013	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8421291,W= 0
1014	IP-10.3.71.7	IP-10.3.30.1	IP TCP	S= 9363103,L= 0,A= 8421291,W= 8760

Figure 4. *When you use a protocol analyzer, you must learn how to interpret the sequence and acknowledgment number.*