TCP/IP Illustrated, Volume 1

The Protocols

W. Richard Stevens

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Library of Congress Cataloging-in-Publication Data
Stevens, W. Richard
TCP/IP Illustrated: the protocols/W. Richard Stevens.
p. cm. — (Addison-Wesley professional computing series)
Includes bibliographical references and index.
ISBN 0-201-63346-9 (v. 1)
1.TCP/IP (Computer network protocol) I. Title. II. Series.
TK5105.55S74 1994
004.6'2—dc20

Text printed on recycled and acid-free paper

ISBN 0-201-63346-9

10 11 12 13 14 15 16 MA 00 99 98 97

10th Printing July, 1997

Each TCP segment contains the source and destination *port number* to identify the sending and receiving application. These two values, along with the source and destination IP addresses in the IP header, uniquely identify each *connection*.

The combination of an IP address and a port number is sometimes called a *socket*. This term appeared in the original TCP specification (RFC 793), and later it also became used as the name of the Berkeley-derived programming interface (Section 1.15). It is the *socket pair* (the 4-tuple consisting of the client IP address, client port number, server IP address, and server port number) that specifies the two end points that uniquely identifies each TCP connection in an internet.

The *sequence number* identifies the byte in the stream of data from the sending TCP to the receiving TCP that the first byte of data in this segment represents. If we consider the stream of bytes flowing in one direction between two applications, TCP numbers each byte with a sequence number. This sequence number is a 32-bit unsigned number that wraps back around to 0 after reaching $2^{32} - 1$.

When a new connection is being established, the SYN flag is turned on. The *sequence number field* contains the *initial sequence number* (ISN) chosen by this host for this connection. The sequence number of the first byte of data sent by this host will be the ISN plus one because the SYN flag consumes a sequence number. (We describe additional details on exactly how a connection is established and terminated in the next chapter where we'll see that the FIN flag consumes a sequence number also.)

Since every byte that is exchanged is numbered, the *acknowledgment number* contains the next sequence number that the sender of the acknowledgment expects to receive. This is therefore the sequence number plus 1 of the last successfully received byte of data. This field is valid only if the ACK flag (described below) is on.

Sending an ACK costs nothing because the 32-bit acknowledgment number field is always part of the header, as is the ACK flag. Therefore we'll see that once a connection is established, this field is always set and the ACK flag is always on.

TCP provides a *full-duplex* service to the application layer. This means that data can be flowing in each direction, independent of the other direction. Therefore, each end of a connection must maintain a sequence number of the data flowing in each direction.

TCP can be described as a sliding-window protocol without selective or negative acknowledgments. (The sliding window protocol used for data transmission is described in Section 20.3.) We say that TCP lacks selective acknowledgments because the acknowledgment number in the TCP header means that the sender has successfully received up through but not including that byte. There is currently no way to acknowledge selected pieces of the data stream. For example, if bytes 1–1024 are received OK, and the next segment contains bytes 2049–3072, the receiver cannot acknowledge this new segment. All it can send is an ACK with 1025 as the acknowledgment number. There is no means for negatively acknowledging a segment. For example, if the segment with bytes 1025–2048 did arrive, but had a checksum error, all the receiving TCP can send is an ACK with 1025 as the acknowledgment number. In Section 21.7 we'll see how duplicate acknowledgments can help determine that packets have been lost.

The *header length* gives the length of the header in 32-bit words. This is required because the length of the options field is variable. With a 4-bit field, TCP is limited to a 60-byte header. Without options, however, the normal size is 20 bytes.