

The figure shows the movement of the data in one direction. At the sender, the

buffer has three types of chambers. The white section contains empty chambers that

can be filled by the sending process (producer). The colored area holds bytes that have been sent but not yet acknowledged. The TCP sender keeps these bytes in the buffer until it receives an acknowledgment. The shaded area contains bytes to be sent by the sending TCP. However, as we will see later in this chapter, TCP may be able to send only part of this shaded section. This could be due to the slowness of the receiving process, or congestion in the network. Also note that after the bytes in the colored chambers are acknowledged, the chambers are recycled and available for use by the sending process. This is why we show a circular buffer. The operation of the buffer at the receiver is simpler. The circular buffer is divided into two areas (shown as white and colored). The white area contains empty chambers to be filled by bytes received from the network. The colored sections contain received bytes that can be read by the receiving process. When a byte is read by the receiving

process, the chamber is recycled and added to the pool of empty chambers.

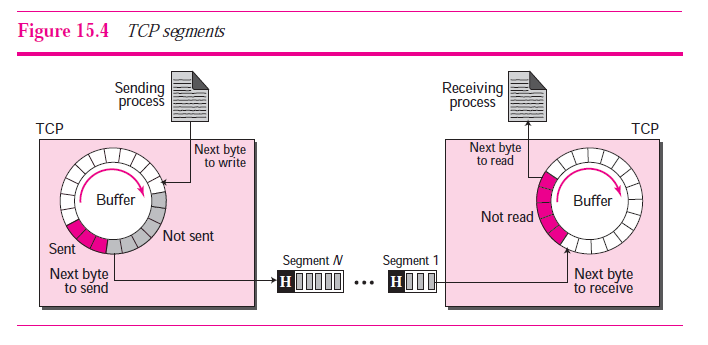
***Segments***

Although buffering handles the disparity between the speed of the producing and consuming processes, we need one more step before we can send data. The IP layer, as a service provider for TCP, needs to send data in packets, not as a stream of bytes. At the transport layer, TCP groups a number of bytes together into a packet called a *segment*. TCP adds a header to each segment (for control purposes) and delivers the segment to the IP layer for transmission. The segments are encapsulated in an IP datagram and transmitted. This entire operation is transparent to the receiving process. Later we will see that segments may be received out of order, lost, or corrupted and resent. All of these are handled by the TCP sender with the receiving application process unaware of TCP’s activities. Figure 15.4 shows how segments are created from the bytes in the buffers.

Note that segments are not necessarily all the same size. In the figure, for simplicity,

we show one segment carrying 3 bytes and the other carrying 5 bytes. In reality,

segments carry **hundreds** or **thousands** of bytes.



**Full-Duplex Communication**

TCP offers *full-duplex service,* where data can flow in both directions at the same time. Each TCP endpoint then has its own sending and receiving buffer, and segments move in both directions.