Chapter 8

Datagrams, Frames, and Packets

8.1 Data Encapsulation

One of the revolutionary concepts that made modern computer networks possible was the encapsulation of data into well-defined and easily managed sets of data. Each of these can be numbered in an order and checked for errors. The Transport Layer has the task of numbering packets and adding error checking before they are sent so that they can be put in order and accounted for by the Transport Layer on the other end.

When we talk about the Internet, we usually talk about sending packets over the network but there are several types of data encapsulation that we also need to learn. Encapsulation on the Data Link Layer can be very different from the higher layers because it may depend on the physical medium, etc. Data Link Layer packets, called frames, may, for example break higher level packets into smaller parts or group them together into larger ones. IP packets come in two basic types: datagrams and segments. The type that is used depends on the Transport Layer protocol. In addition to these basic types, discussed in more detail below, there are a huge variety of legacy encapsulation formats and formats used for for special communication.

The basic structure of encapsulated data is a header, data to be sent (also called the payload or payload data) and sometimes a trailer. The header typically shows where the packet begins and also gives meta-data. Meta-data may include protocol and method identification, error correction and detection, the source and destination address, how many more times the packet should be sent before being disposed, and even some of the history.
of the packet (where it has been sent and data about network congestion). The amount of data in the payload is limited and depends on the protocol. The trailer may include a checksum or other error correction but also tells where the packet ends in cases where the total size or payload size are not included in the header. The unit of information is often the octet, which is exactly one byte, eight bits.

8.2 Frames

Packets in the Data Link Layer are called frames. A frame has markers to show the beginning and end of the packet as well as addresses for sending and receiving.

A typical frame is the Ethernet frame. The Ethernet frame starts with the 7 identical bytes containing the binary for decimal 170, followed by the binary 171 for a total of eight octets. The next six octets are the MAC address of the source device. The MAC address is the address of a physical device, typically belonging to the NIC. The destination MAC address follows. Next, there are several optional header items and finally a two octet number identifying the Ethernet protocol type or the size, depending on which Ethernet type is being used and whether the number is greater than the maximum total 1536 octets.

8.3 Packets

Frames in Internet Protocol (IP) are called packets. Packets are used primarily on the Network layer so their role is to deliver a packet from one logical address (in the case of IP, an IP address) to another. Here we will discuss IPv4 (IP version 4) packets. IPv6, which is becoming more important and is the only other IP version used at this time, is discussed in Chapter 8.

The IP packet header is the header used by routers to send packets through the network from source to destination so understanding the IP packet also gives a brief introduction to the basic concept of routing. The router depends on several key fields in the IPv4 header: the source and destination IP addresses, a counter called the Time to Live (TTL), and the checksum. The IPv4 source and destination addresses are each 32 bits, four octets. The TTL counter is only one octet so its maximum value when first
8.4 DATAGRAMS

set is 255. (However, most systems in end-to-end communication set it to 128 or less to start.) The TTL performs the essential role of determining whether a packet has been going from router to router without being delivered to its destination. For example, if a router is misconfigured, it might send some packets in circles and they would continue to increase in number. Instead, every time a router receives a packet, it decrements the counter and checks whether the value is zero. If the packet is at zero, the routing device discards the packet and sends a special packet back to the source to tell the source that the packet was not delivered. (The name “Time to Live” implies clock time in seconds and that was the original intention of this field. Instead it should be considered the number of “hops,” as the counter was renamed in IPv6.)

Corrupt packets are also detected in IPv4. Each router computes the checksum and discards corrupt packets. The checksum is a simple one’s complement (inversion) sum, which is inverted again to form the final two-octet sum. If the packet is corrupt, it is discarded. Routers sometimes change the headers of packets they are forwarding and in these cases the checksum must be recomputed before forwarding.

8.4 Datagrams

A datagram is the general word for a packet that is over a connectionless transport protocol. That is, the transmission, timing, and order are not guaranteed. For example, a packet sent over User Datagram Protocol (UDP) is a datagram. A datagram contains a header and a payload (with the data to be communicated). In UDP, the header contains source port number and a destination port number. A port number is very different from an address. Because one device (or even one program) may be handling many network connections, sometimes from different sources, the operating system (OS) and its programs need to know how to handle a packet. Each port number is two octets.

8.4.1 Port numbers

A port number is simply a number that identifies some process using network resources. When a packet is sent, it is sent out “through” this abstract “port” and when a packet is received, it is accepted through a port. In the beginning,
ports were simply for the convenience of handling packets and keeping track of network connections. As networks became global and ubiquitous, LANs were receiving more and more unexpected packets from the WAN, packets that were unwanted or even harmful. So ports are used for a very important purpose now: If a packet arrives to a port that not being used (for example as a server port), or no connection was made over that port to begin with, that packet can be discarded. Usually this task is done by the gateway router (for example NAT router) connected to the WAN. If you want to set up a special service or peer-to-peer connection through such a device, you may have to tell it to expect connections on that port. Even though ports are just numerical abstractions, they play an important role in keeping our network connections secure so programmers must take care when opening and closing ports. Users are safer when they close ports they aren’t using by using the settings in the OS. Otherwise, users don’t have to handle ports directly because the programs and OS will handle opening and closing necessary ports. In TCP and UDP, port numbers are two bytes so there are 65,535 possible port numbers for each protocol. The allocation of these numbers is discussed in Chapter 10.

8.5 Segments

IP packets sent over a connection-oriented protocol, such as Transmission Control Protocol (TCP), are called segments. (Here, of course, the word segment means something very different from the physical segments of a LAN, as described in Chapter 4.) These packets must contain a method of checking the integrity of each packet (error checking) and a method of enumerating (numbering) the packets to check packet order and to confirm their arrival. So a TCP packet is more complex than a UDP packet.

The TCP header starts the same as the UDP header, with two port numbers, one for the source and one for the destination. The next field is the four-octet sequence number, the number that allows the protocol to check for packets out-of-order or missing. The next field, also four octets, is the acknowledgement number. This number is used to acknowledge that previous packets have been received. Next is the data offset, which is four bits telling the number of remaining 32-bit words in the header before the data will begin. After the data offset, there are 12 flags. A flag is a boolean value usually encoded as one bit each. The first three flags are not used yet.
so they are normally set to zero. The nine flags used in the TCP segment give information about synchronization and acknowledgement, as well as the significance of other fields in the header.

<table>
<thead>
<tr>
<th>Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS, CWR, ECE</td>
<td>First three flags are used to inform routers about network congestion (RFCs 3168 and 3540).</td>
</tr>
<tr>
<td>URG</td>
<td>The urgent pointer is significant.</td>
</tr>
<tr>
<td>ACK</td>
<td>The acknowledgement number is significant (set in all but the first SYN).</td>
</tr>
<tr>
<td>PSH</td>
<td>The packet wants data pushed to the application buffer.</td>
</tr>
<tr>
<td>RST</td>
<td>Reset the connection.</td>
</tr>
<tr>
<td>SYN</td>
<td>Begin synchronization (set in first packets from each end only).</td>
</tr>
<tr>
<td>FIN</td>
<td>This is the final packet from the sender.</td>
</tr>
</tbody>
</table>

Table 7.1 TCP flags, bits that contain boolean values to communicate how the packet is to be used.

The number of possible sequence numbers is very large, at $2^{32}$ (more than 4 billion). In the early Internet, therefore, sessions could begin exchanges at a random point in the sequence and the only purpose was to allow the packets to be sequenced. As the networks got busier and unexpected packets began to arrive more often, the sequence number, like the port number, also has the role of authenticating a packet to ensure that it is actually part of the expected transaction. In order to understand the sequence number and acknowledgement number, we will need to discuss in detail the TCP transaction, in particular how two agents start communication in TCP.

8.6 The TCP Three-way Handshake

A handshake is part of an end-to-end protocol in which two agents (software programs or devices) start a reliable connection. The TCP handshake begins with one agent (usually the client in a client-server relationship) sending the
other a special TCP packet called a SYN packet, with a pseudo-randomly generated sequence number $A$, to start the sequence. SYN stands for synchronize and is also the name of the flag bit that is set in a SYN packet. If the receiving agent (usually the client in client-server) will accept the connection, it sends back a packet called a SYN-ACK packet. ACK means acknowledge. The SYN-ACK packet has its acknowledgement number set to the next number after the SYN packet’s sequence number, $A + 1$. The SYN-ACK has its sequence number set to a pseudo-randomly selected number $B$. To complete the handshake, the machine that sent the first SYN, upon receiving the correct SYN-ACK, sends an ACK packet with the sequence number set to $A + 1$ and the acknowledgement number set to $B + 1$.

After this handshake, both agents can send or receive packets, each time incrementing the sequence number. Returning packets acknowledge packets received with the acknowledgement number. In this way, packets out of order can be reordered and lost or corrupt packets can be resent. There is a lot of other information available in the TCP protocol that makes it a good method for sending data that must be exactly correct and complete. Confirming each packet requires much time and more when packets are lost or corrupt so TCP is not preferred for data that must be used immediately, such as real-time voice data or streaming video for two-way interaction.

TCP also allows agents communicating on the web to confirm the other agent’s IP address before sending data. The TCP handshake requires that both parties send their correct IP addresses. If the IP address were not correct, the return packets would not reach their destination and the handshake could not be completed. Spoofing means when an agent intentionally uses the wrong address, name, e-mail address, etc. As you can see from the handshake, spoofing an IP address over TCP would be difficult. So once the TCP handshake is complete, both parties can be fairly confident that the IP address of the other party is correct.

8.7 Glossary

**ACK** — ACKnowledgement packet in the TCP/IP three-way handshake that initiates a connection-based communication session.
acknowledgement — a confirmation that something was received. In TCP/IP an acknowledgement can be explicit (such as an ACK packet during handshake) or implicit (via the value of the acknowledgement number field in a data packet).

acknowledgement number — a field in a TCP/IP packet that informs the receiver how many of its packets have been received and processed by the sender.

buffer — an area of memory in which data is stored or exchanged.

congestion — a measure of the deterioration of communication quality due to excessive load being placed on the network.

corrupt — made unreliable by error or alteration.

datagram — a basic unit of data transfer in a packet-switched network for which none of arrival time, order of arrival, or even eventual delivery are guaranteed. Datagrams in a TCP network are encapsulated in link-layer frames, exchanged between network-layer endpoints, and can encapsulate transport-layer TCP segments.

decrement — to decrease (typically by one).

destination — the address of the machine to which a packet is sent over a network.

encapsulation — placing information for one layer or service within a representation belonging to a different layer of service. In networking, encapsulation is achieved by prepending headers and/or appending trailers to data packets. An entire packet (including header) at one layer in the communication model becomes the payload in a packet belonging to a lower layer in the model.

Ethernet frame — The unit of information exchanged between link layer entities in an Ethernet network.

Explicit Congestion Notification (ECN) — an optional extension to TCP/IP (defined in RFC 3168) which allows end-to-end notification of network congestion without dropping packets. (TCP/IP networks usually signal congestion by simply dropping packets.) The ECN flag can be set by any network device to signal impending congestion, which informs the receiver and (indirectly) the sender that the rate of communication is too high. The sender can reduce its transmission rate before any packets have to be dropped.
**flag** — a bit set in a packet header that notifies the receiver of a specific condition or event such as urgent data, impending network congestion, packet fragmentation, etc.

**forwarding** — the relaying by a network device of a packet from one network segment to another.

**frame** — the unit of data transfer between link-layer communication endpoints. Frames in a TCP/IP network encapsulate network-layer datagrams.

**handshake** — a negotiation between two network devices to set the parameters for a virtual communications channel established between them before normal communication begins.

**header** — information prepended to a packet of data before it is transmitted between network devices. A header might add information about the protocol being used, the source and destination process identities, the source and destination machine addresses, the current state of the packet of data, or the current state of the sender’s communication endpoint.

**hop** — a local area transfer of a packet between two connected network devices that form part of the chain of devices between the original source and the final destination. A hop corresponds to the forwarding of a packet from one physical network segment to another. The maximum number of hops for an IP packet is limited (by the TTL field in the IP header) to prevent a packet circulating endlessly when a route leads it back to a network device through which it has already passed.

**MAC address** — the link-layer (local area) physical address of a network device.

**meta-data** — non-application data that describes the packet of which it is a part. Meta-data is typically located in a packet header or trailer and includes information such as protocol identification, error detection/correction, etc.

**octet** — another word for *byte* but emphasizing that one byte contains eight bits.

**one’s complement** — (of a binary number) the result of inverting each bit (swapping ‘1’s and ‘0’s) in a word.

**packet** — a single unit of data exchanged between two devices on a network.
payload — the application-level data included in a packet, typically everything between the header and trailer (if present). Note that the application-level data might be a packet (including its own header and trailer) belonging to a higher layer in the protocol stack, encapsulated as the payload in packet belonging to the lower layer protocol.

port number — a 16-bit number identifying the process, agent, service or protocol responsible for handling the payload contained in a packet. UDP packets arriving at an Internet host use the port number to specify which running process should receive the data. TCP connections use a well-known port number to connect to a server, and then a pseudo-randomly allocated port to continue client-server communication after the handshake is complete. On Unix-like systems a list of well-known port numbers is kept in the file /etc/services.

segment — a single unit of data exchanged between transport-layer endpoints engaged in TCP stream communication. TCP segments are encapsulated in IP datagrams.

sequence number — an indicator of the state of communication between two endpoints, representing in particular the number of packets exchanged in a given direction and therefore providing a quantity by which packets can be ordered. Sequence numbers are usually incremented by one for each successive packet is transmitted, or incremented to the value of the most-recently reliably-delivered packet in a transport layer connection.

source — the address of the machine from which a packet originated.

spoofing — to fraudulently misrepresent some data associated with communication: packet source address, e-mail sender address, etc.

SYN — a flag that can be set in the header of a TCP packet to indicate the synchronisation phase of the connection handshake.

synchronization — the act of establishing common conditions between two endpoints that wish to engage in transport-layer connection-oriented communication. In TCP/IP, the synchronisation phase establishes conditions such as the initial sequence numbers that each endpoint will use to identify the order in which packets are sent and the most-recently received packet that has been reliably delivered to the destination process.
Time to Live (TTL) — a counter associated with IP packets that limits the number of hops a packet may make, thereby preventing infinite loops. Each time a packet is forwarded the TTL field is decremented by one; if it reaches zero, the packet is dropped (not forwarded).

trailer — meta-data appended to the end of a packet (in contrast with the header, which is prepended to the start). The trailer can include a checksum (or other error detection/correction information), an explicit packet end marker, etc.

Transmission Control Protocol (TCP) — a transport-layer protocol in the Internet TCP/IP suite that provides reliable, connection-oriented, stream-based communication between two endpoints. TCP is often used for client-server communication (such as between a web browser and a HTTP server).

User Datagram Protocol (UDP) — the network-layer protocol in the Internet TCP/IP suite that provides unreliable, connectionless, datagram-based communication between two endpoints. UDP is often used for real-time communication where duplicated, dropped or out-of-order packets are either unimportant or are dealt with explicitly by the application layer.