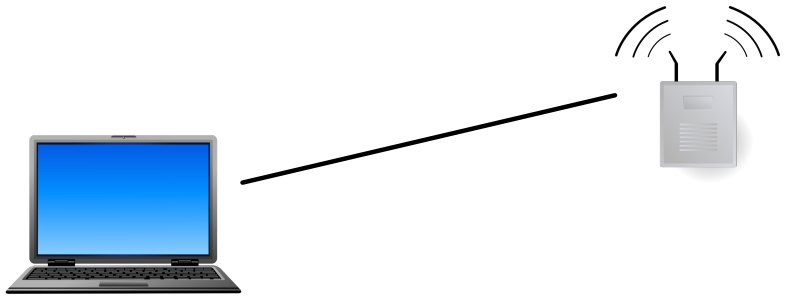
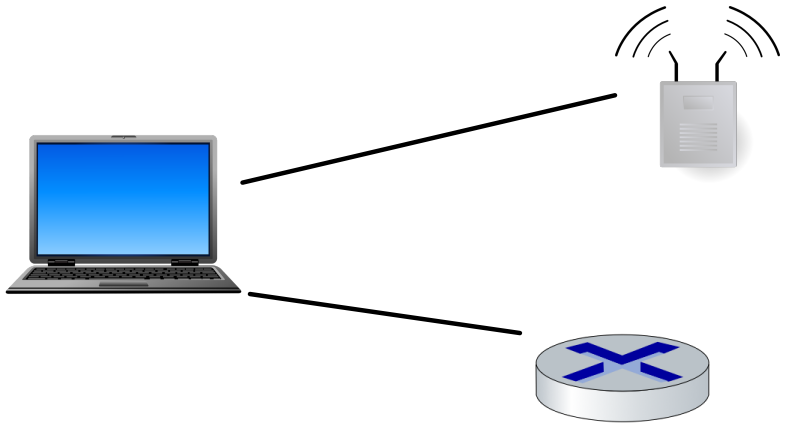
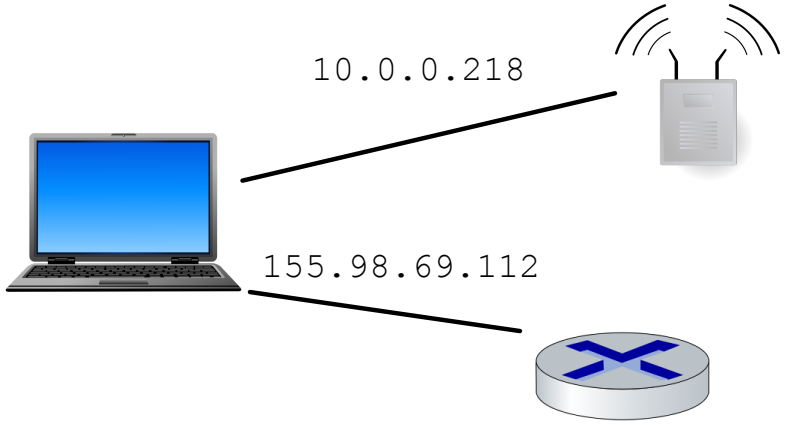
One Machine, Many Networks



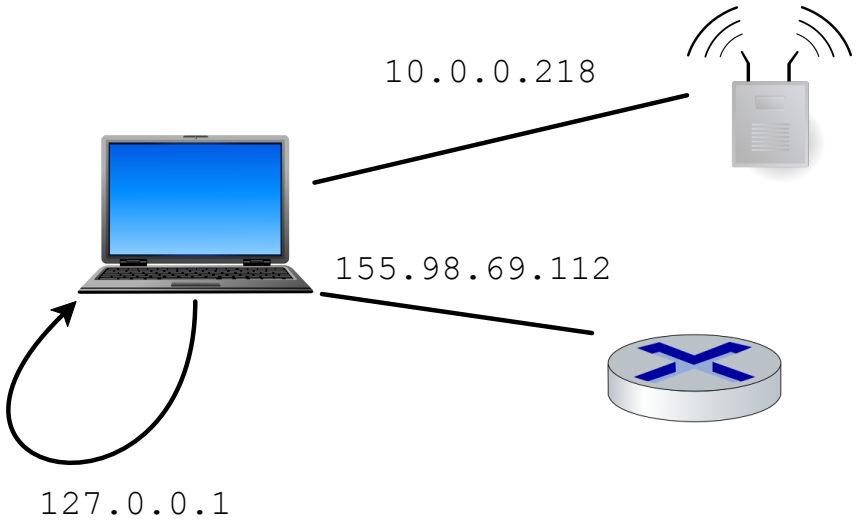
One Machine, Many Networks



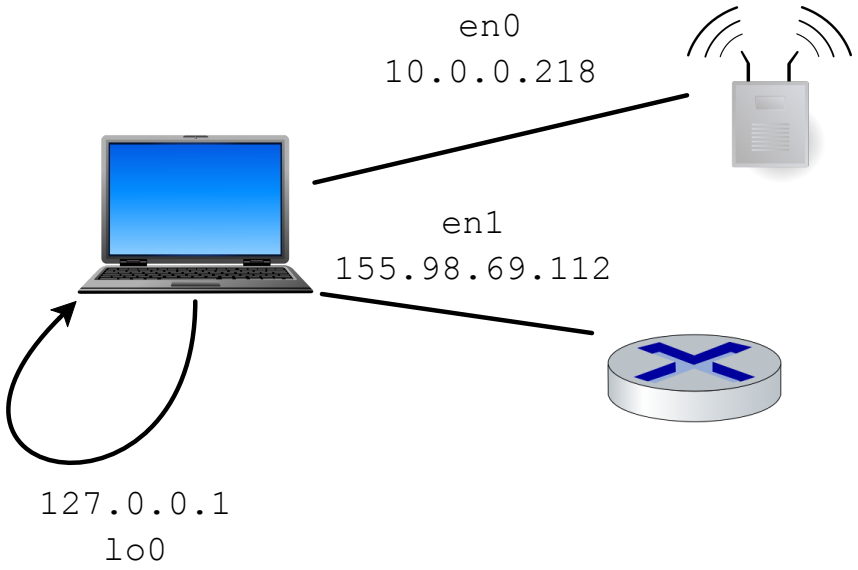
One Machine, Many Networks



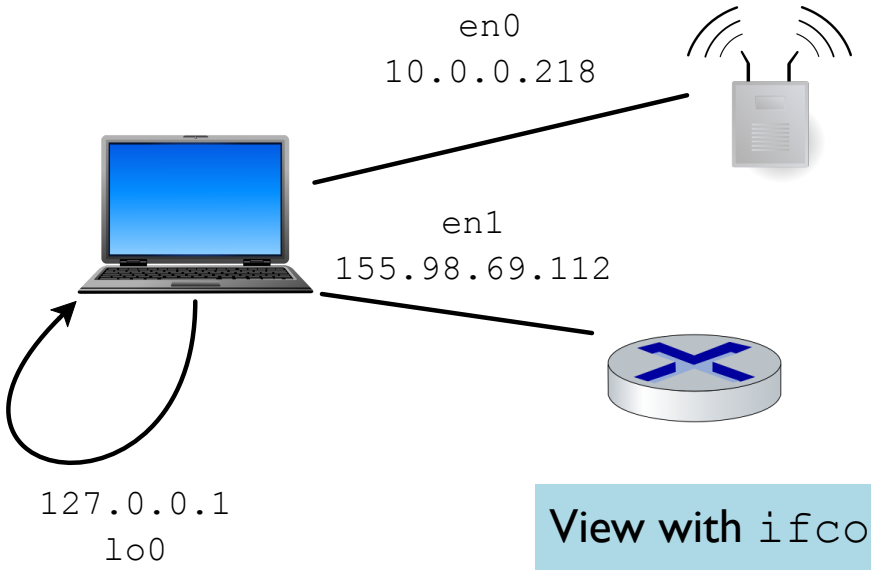
One Machine, Many Networks



One Machine, Many Networks



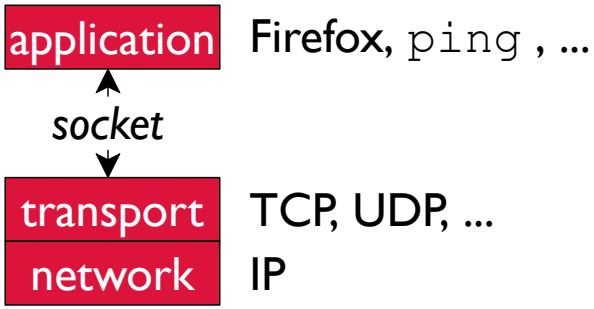
One Machine, Many Networks



View with `ifconfig`

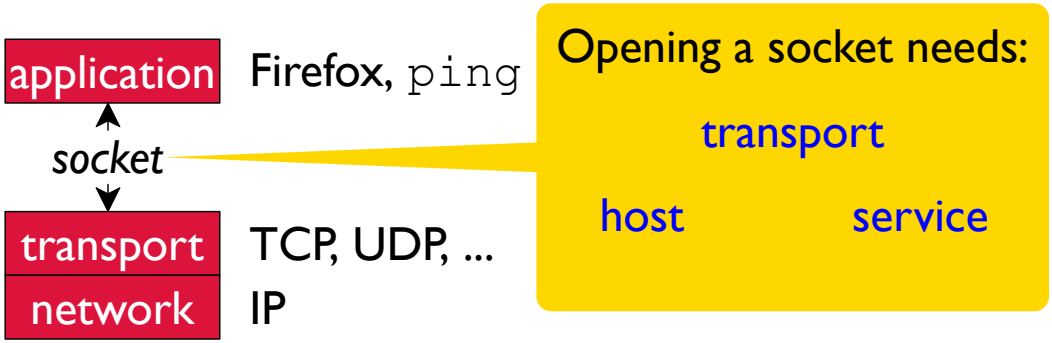
RECAP

Network Layers



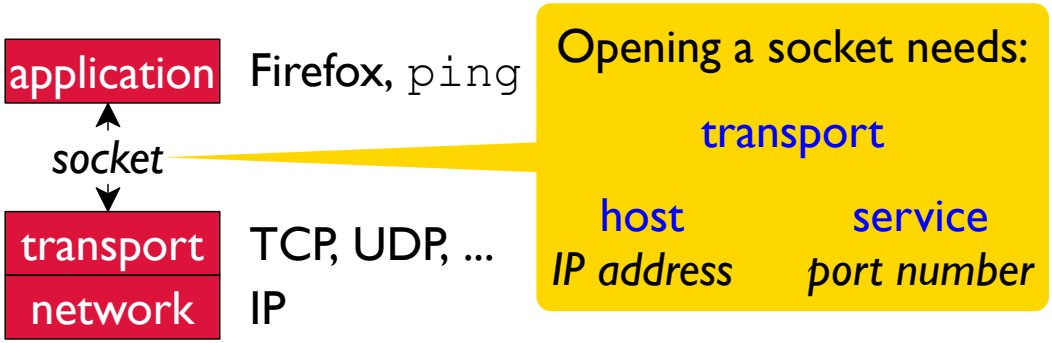
RECAP

Network Layers



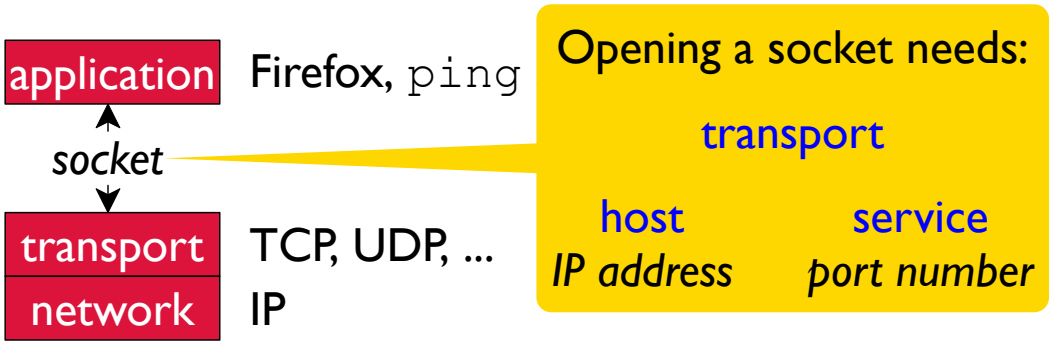
RECAP

Network Layers



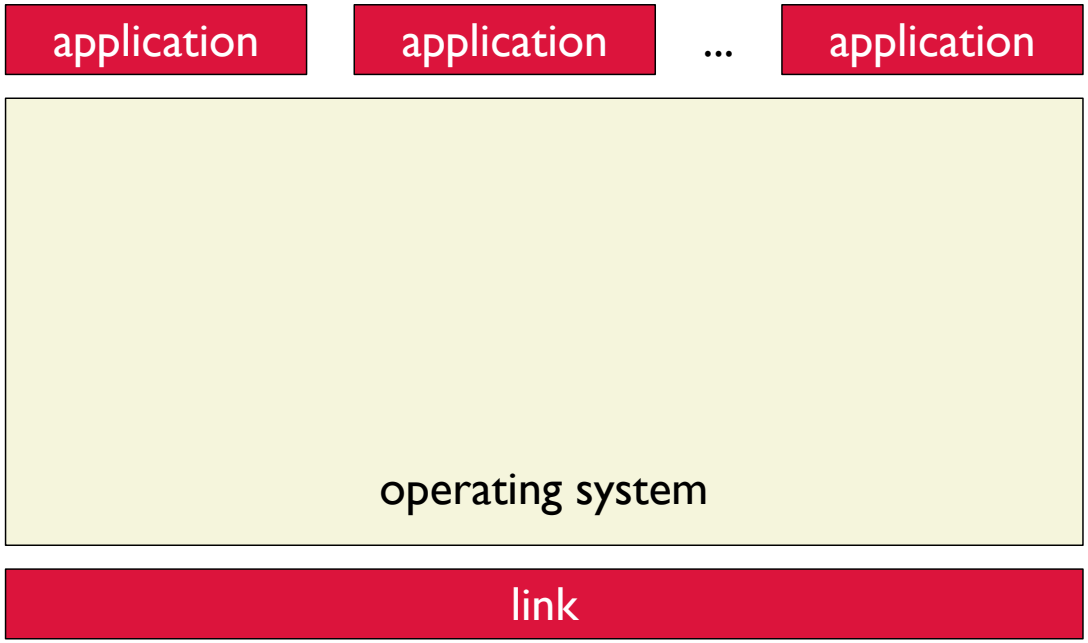
RECAP

Network Layers

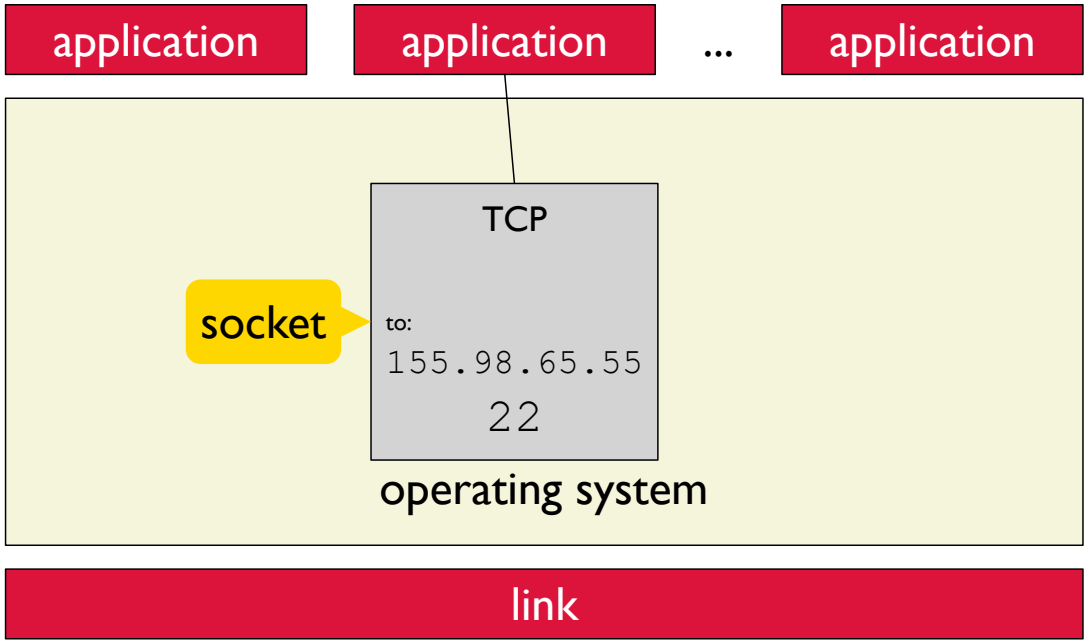


Web server	TCP	`www.cs.utah.edu`	`80`
Modern web server	TCP	`www.cs.utah.edu`	`443`
Mail receiver	TCP	`smtp.cs.utah.edu`	`25`
Name resolver	UDP	`8.8.8.8`	`53`

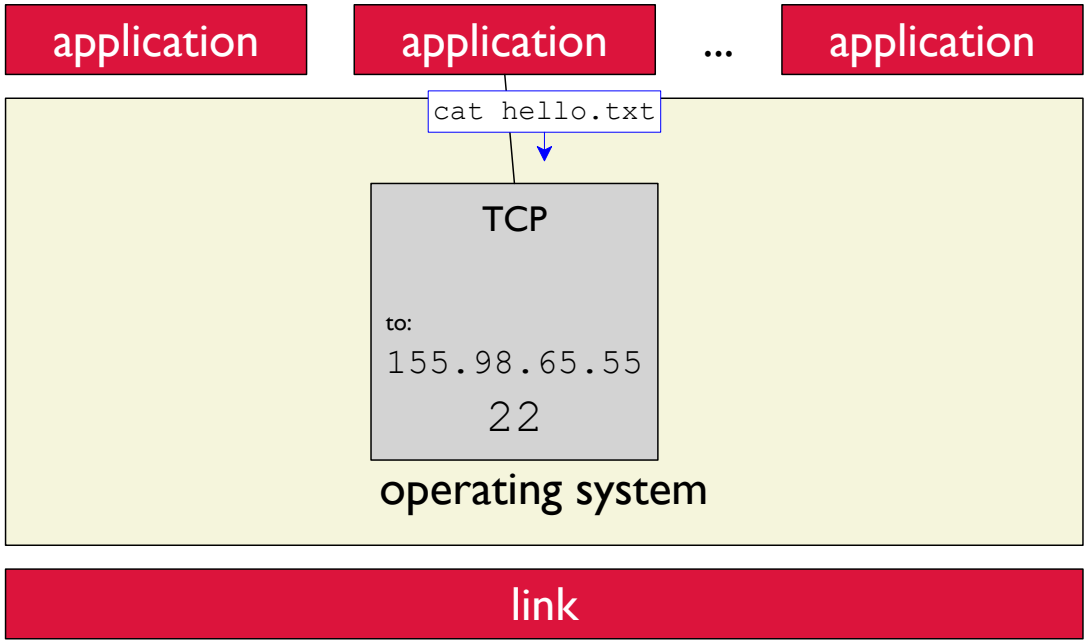
Using Sockets



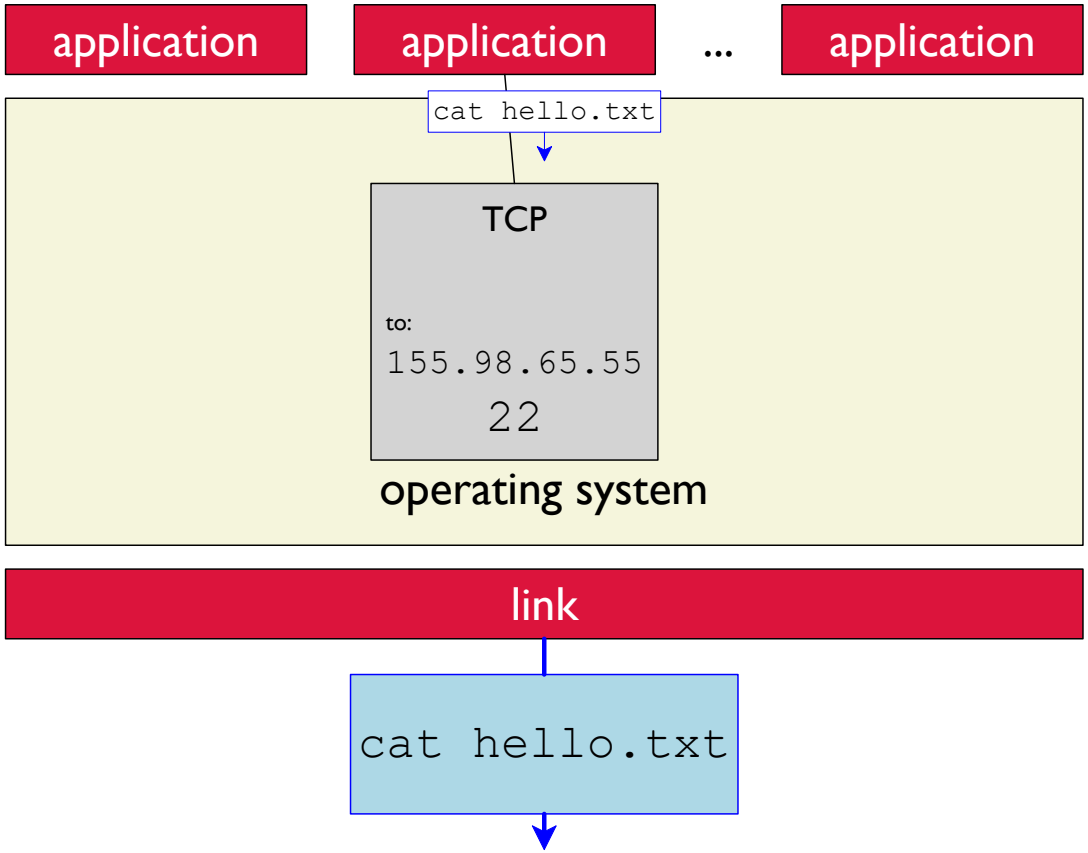
Using Sockets



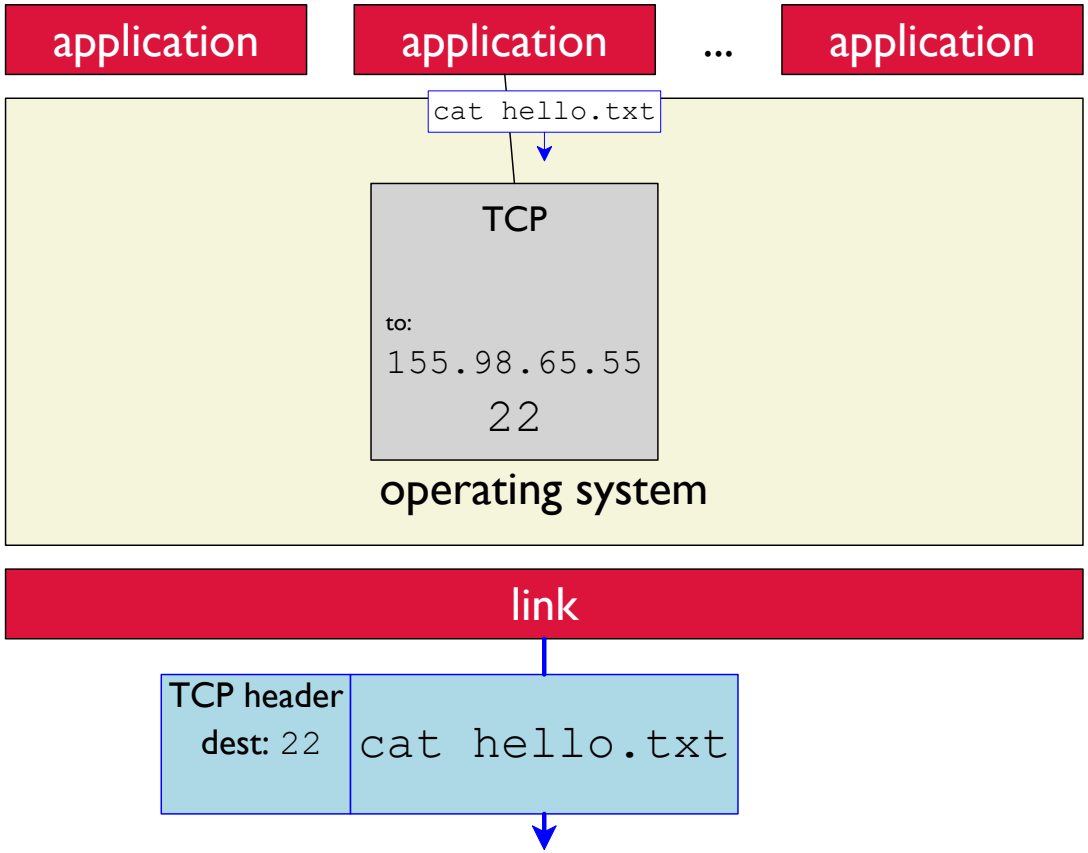
Using Sockets



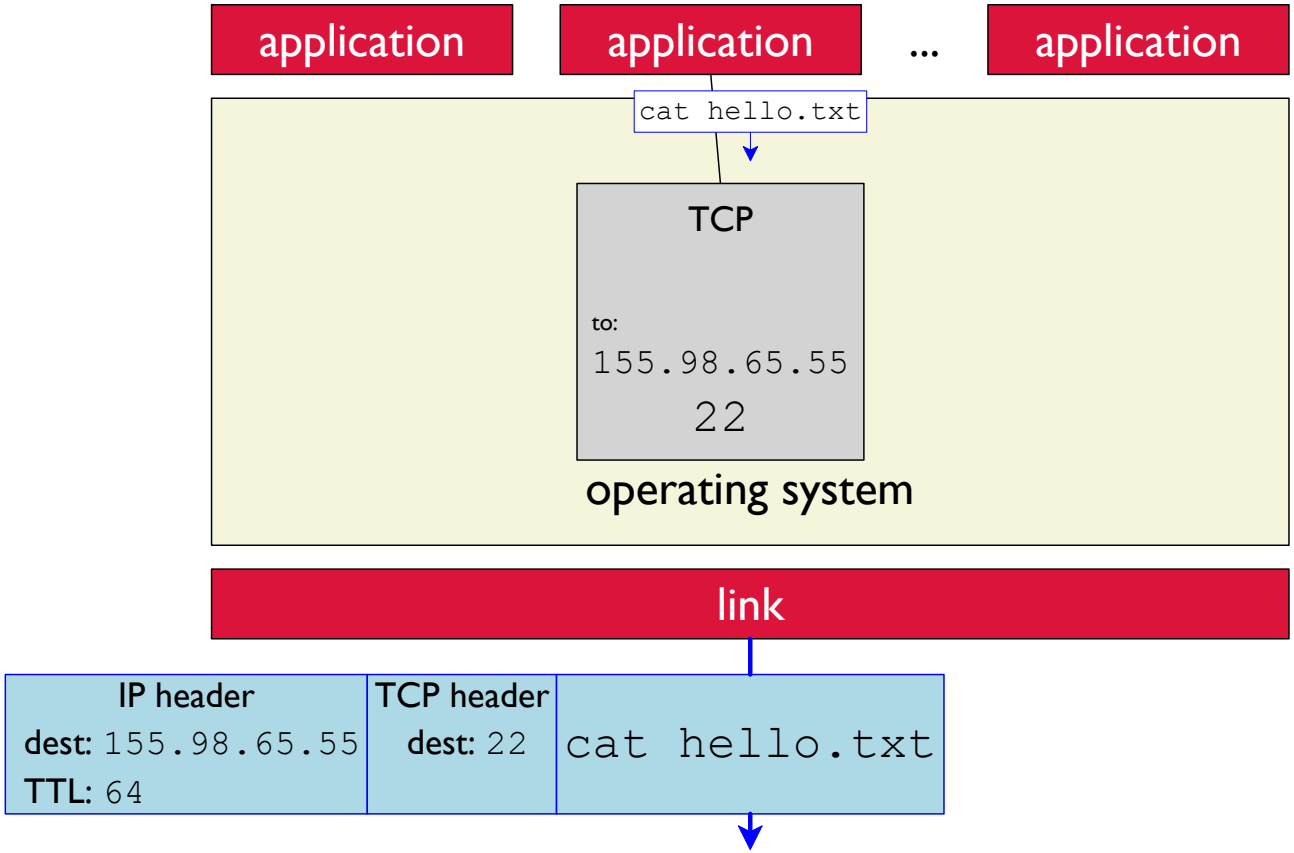
Using Sockets



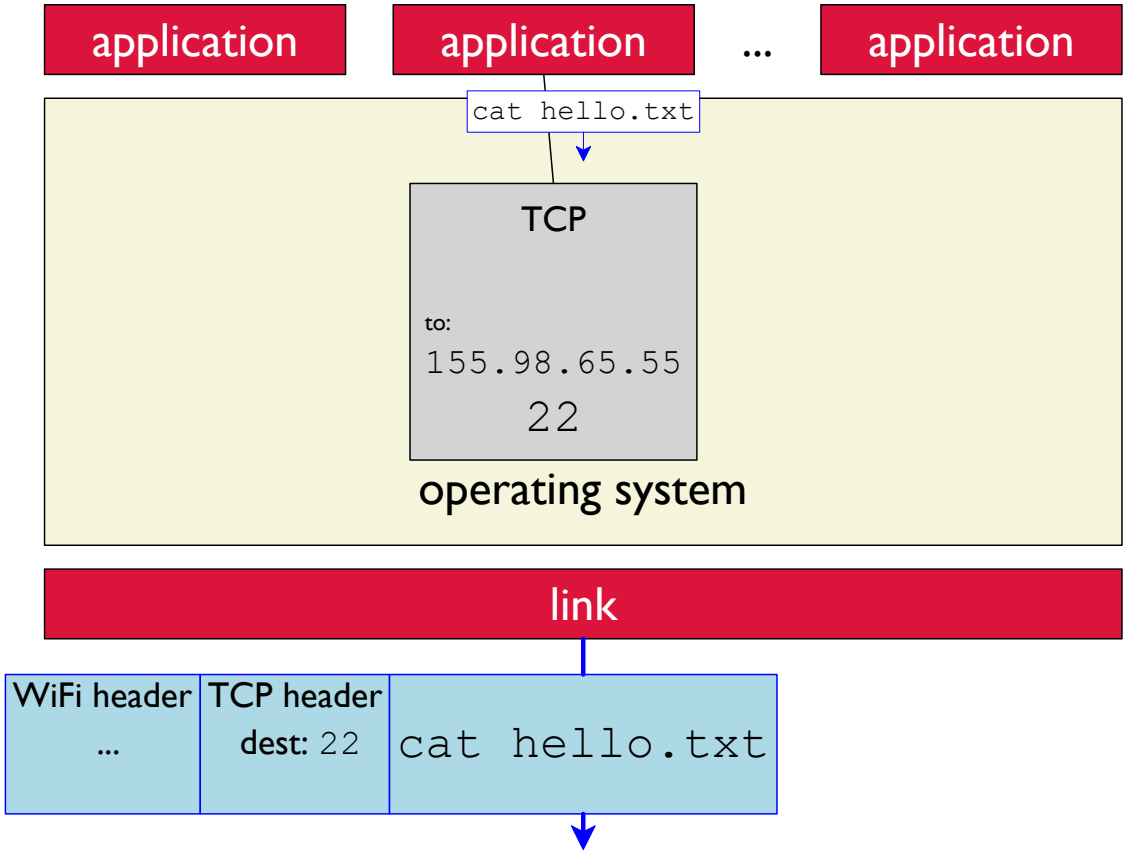
Using Sockets



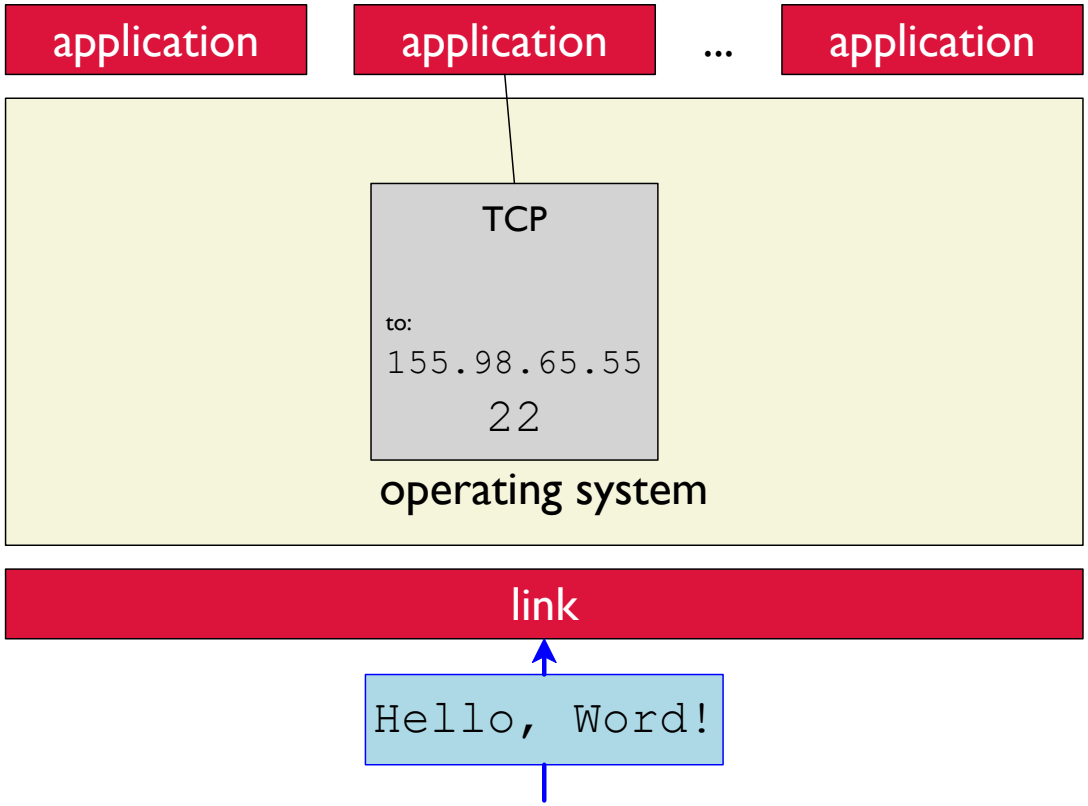
Using Sockets



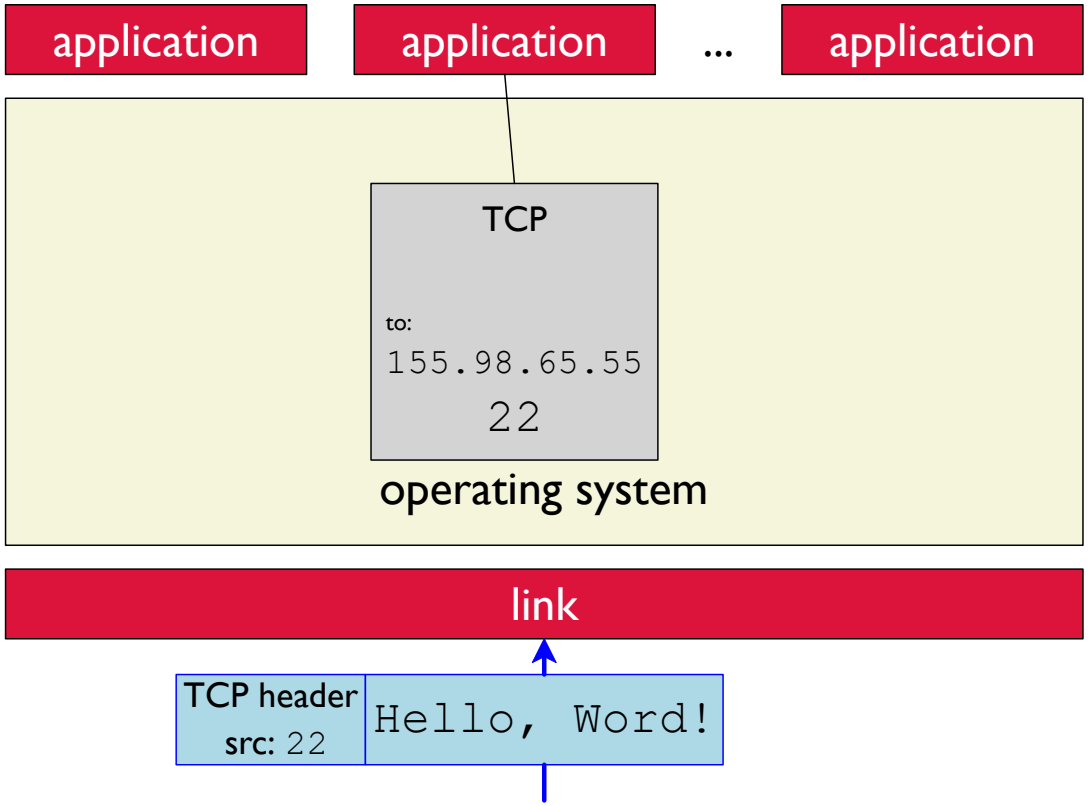
Using Sockets



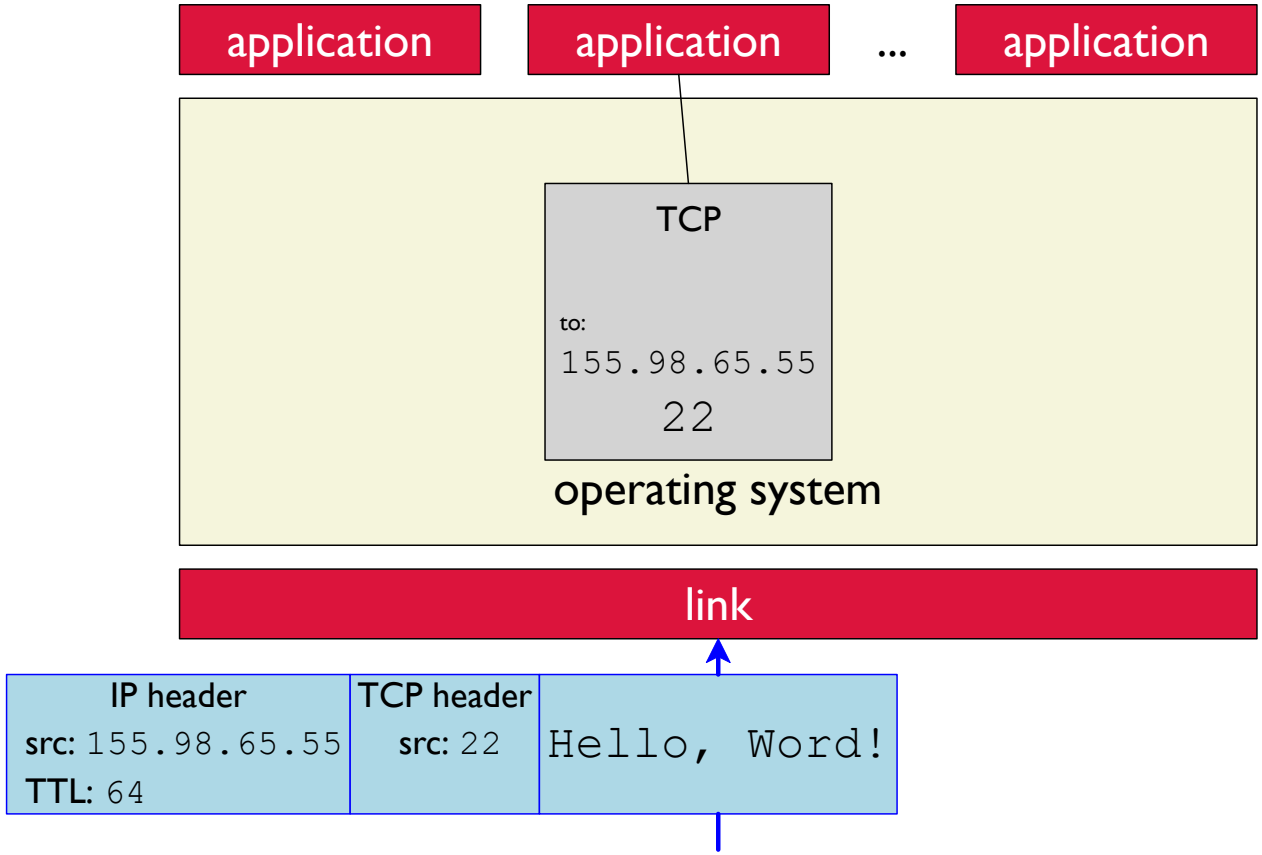
Using Sockets



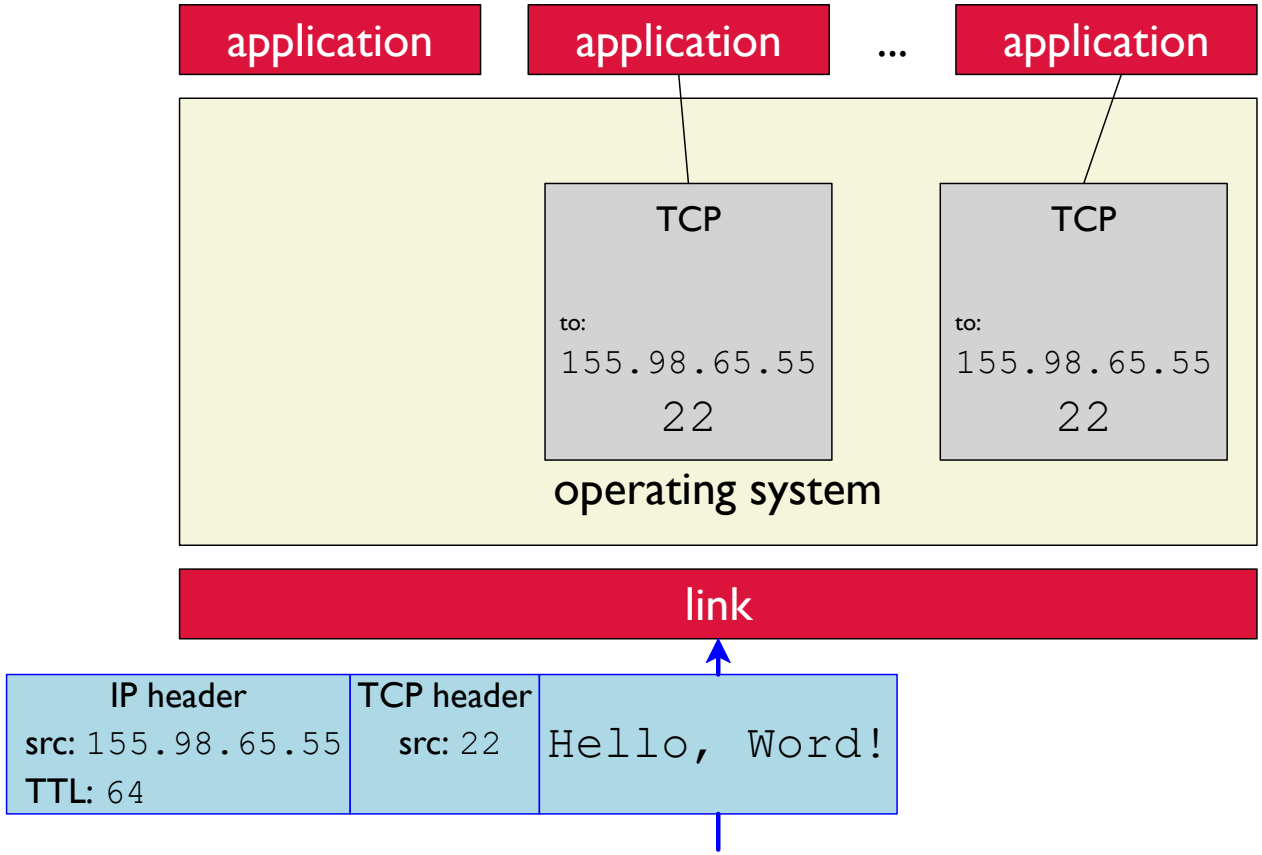
Using Sockets



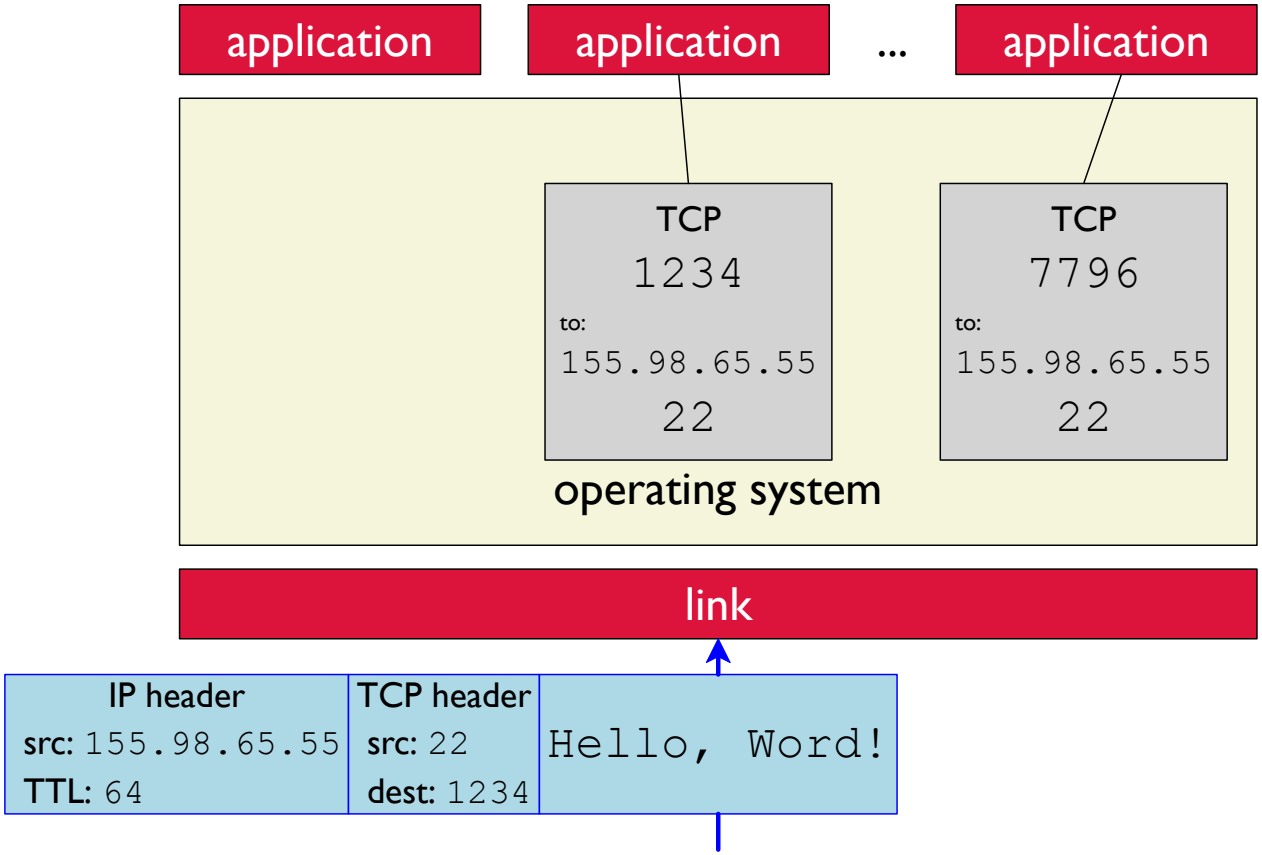
Using Sockets



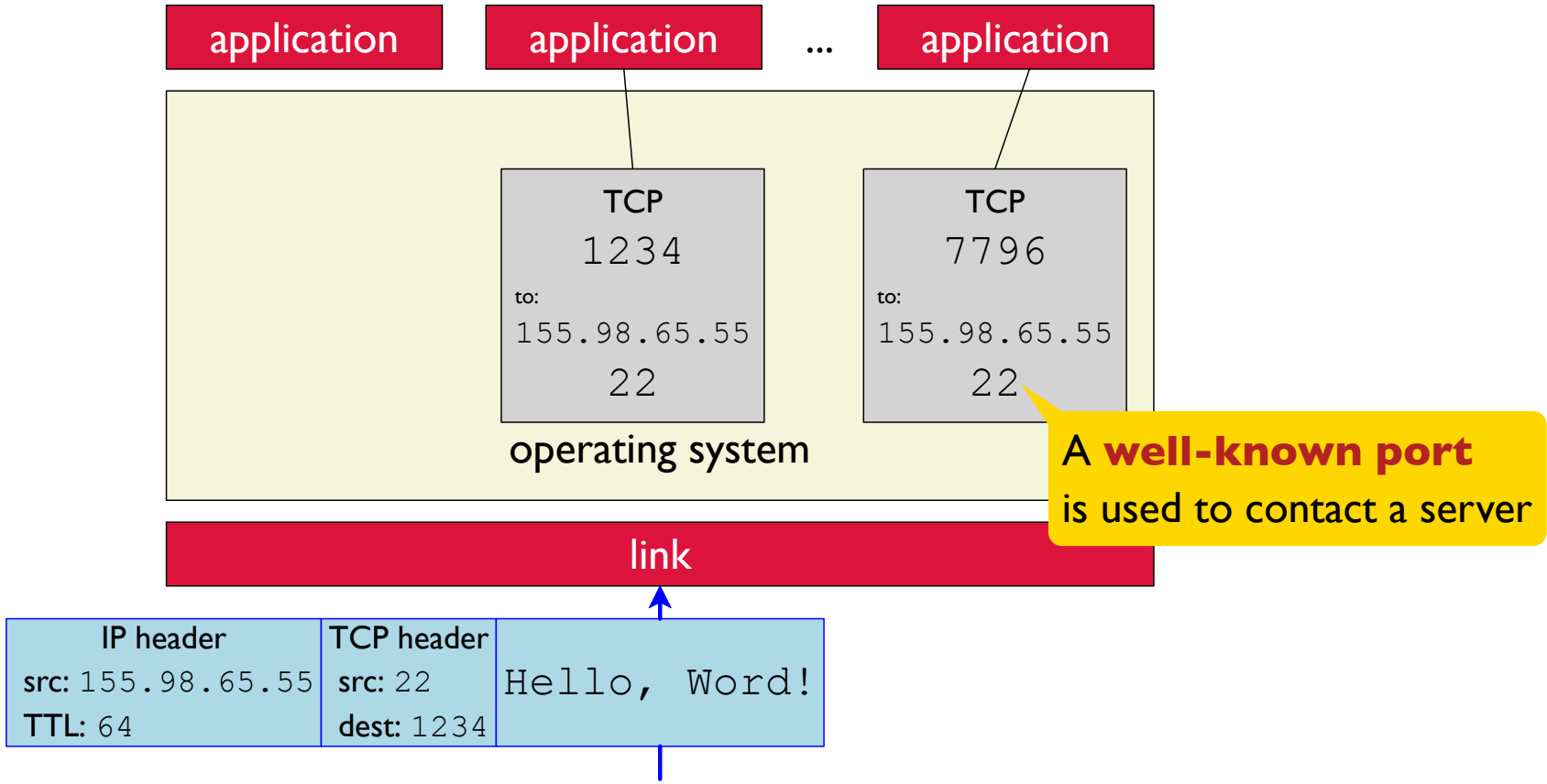
Using Sockets



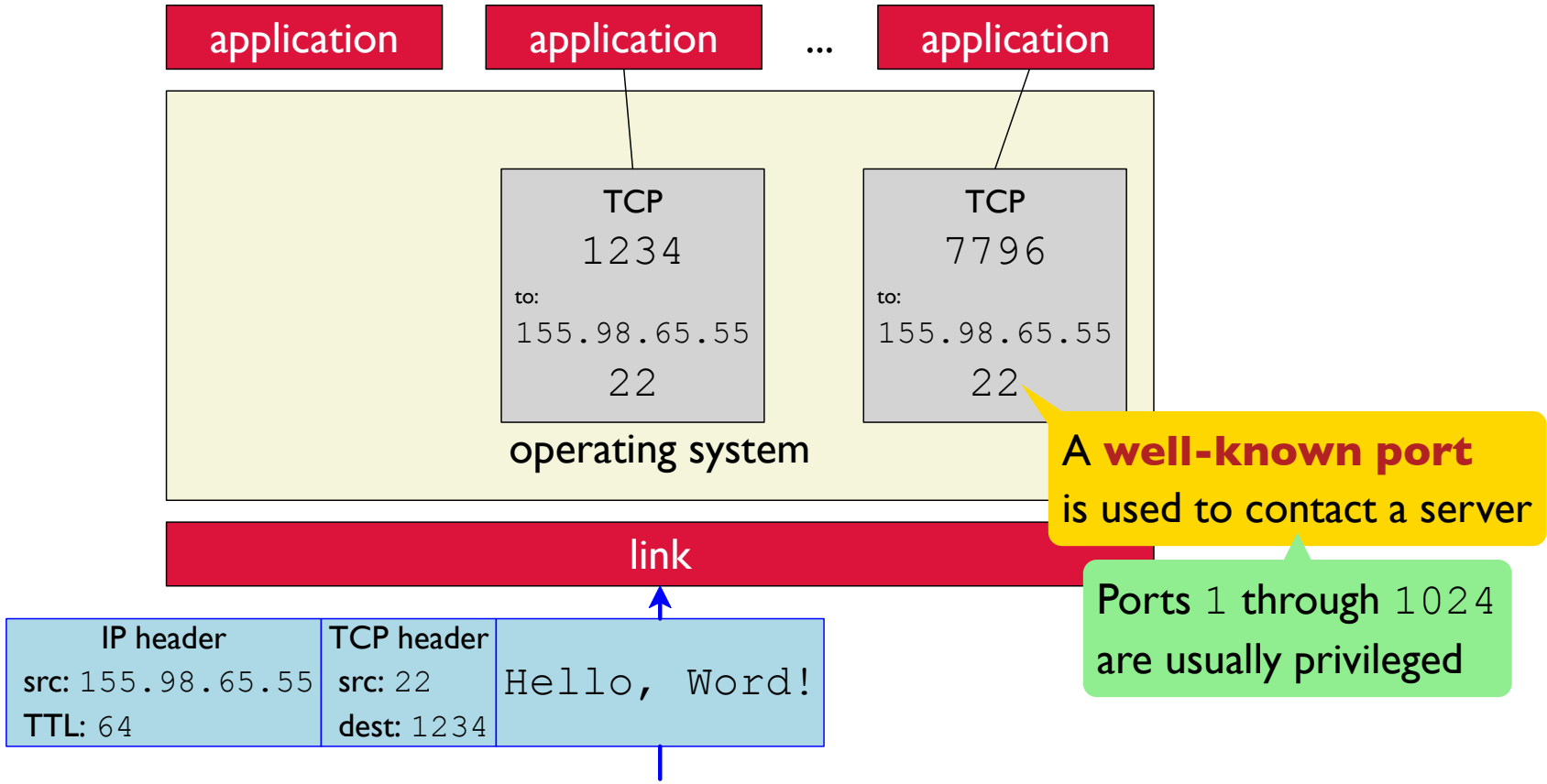
Using Sockets



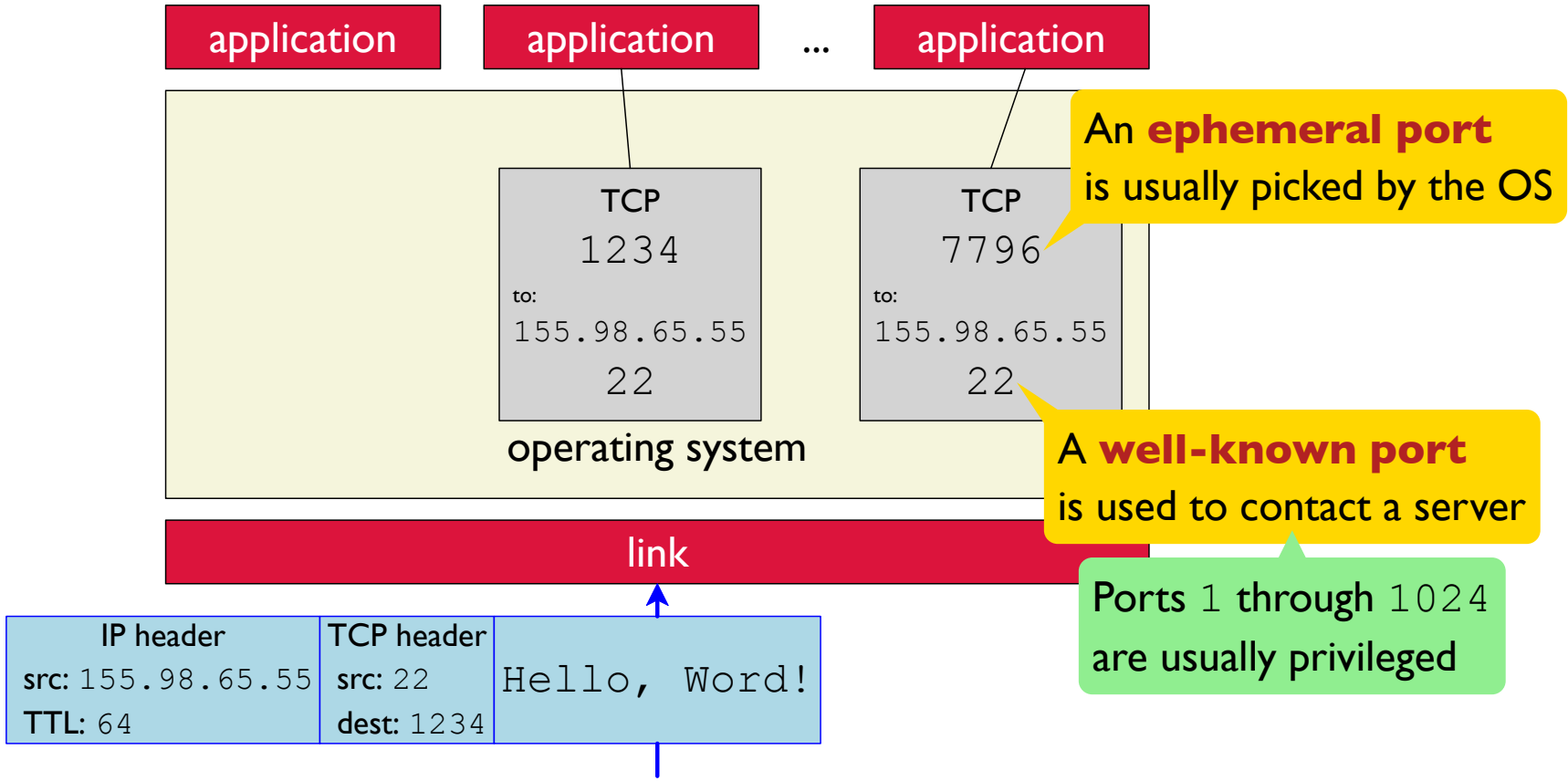
Using Sockets



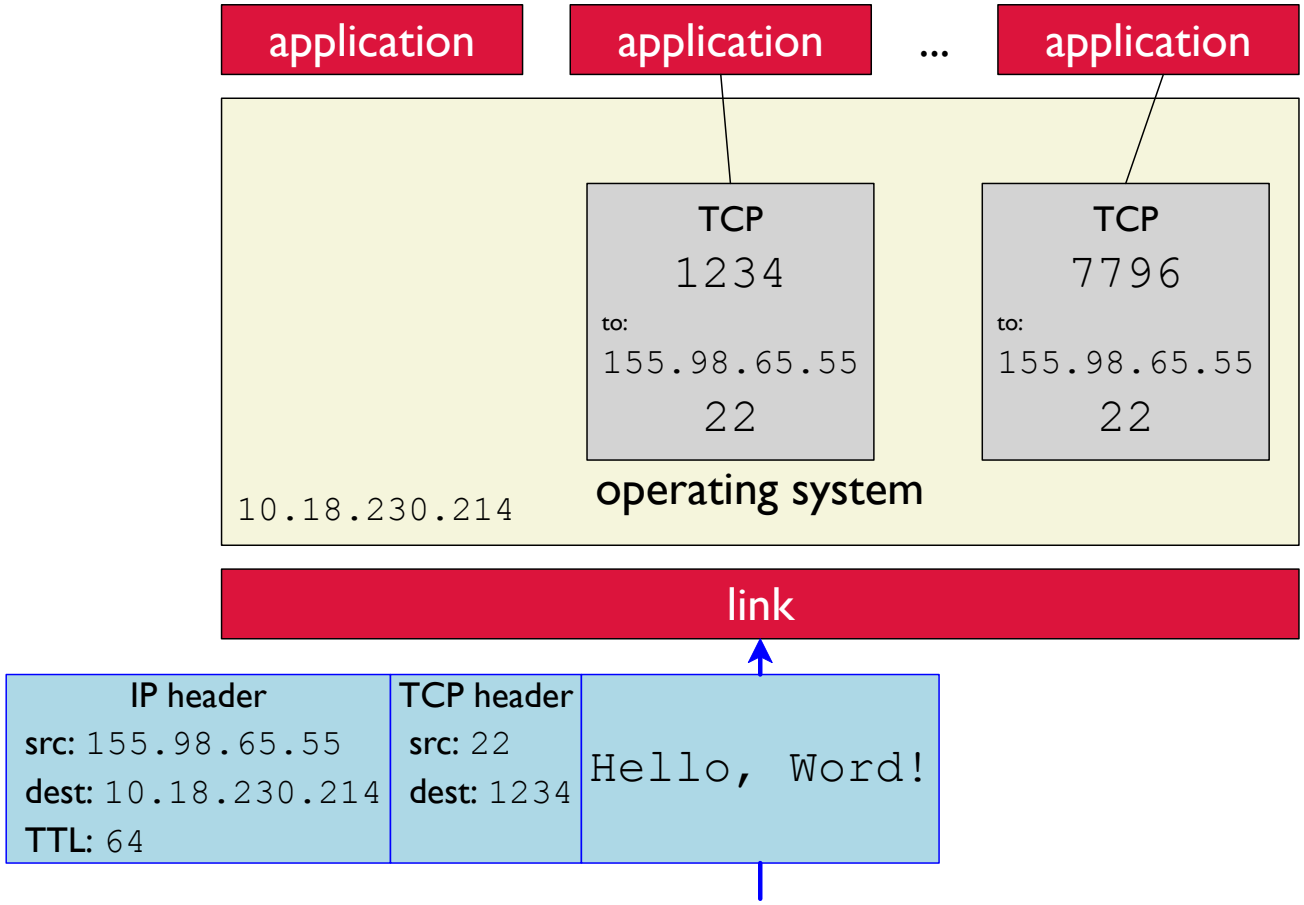
Using Sockets



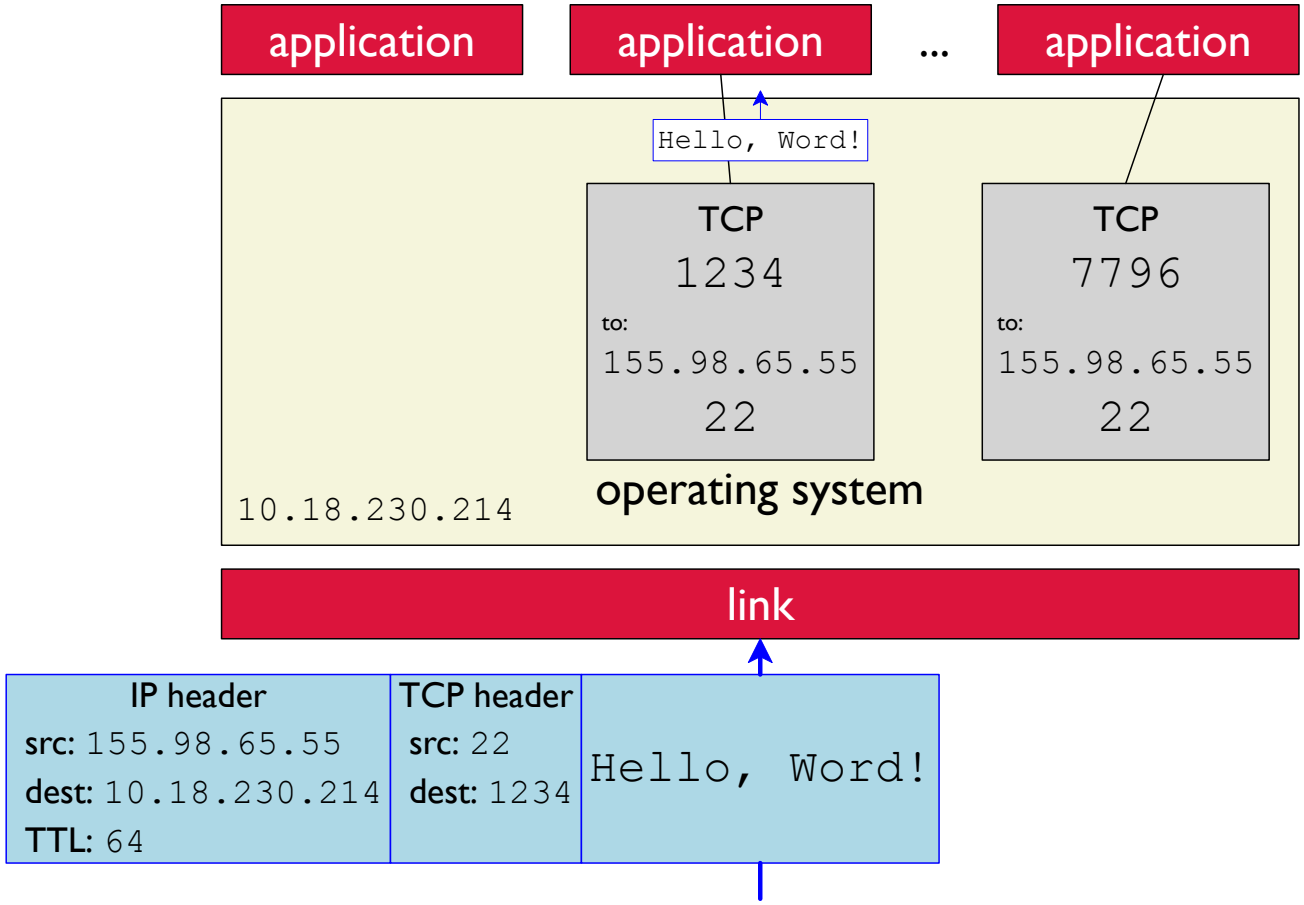
Using Sockets



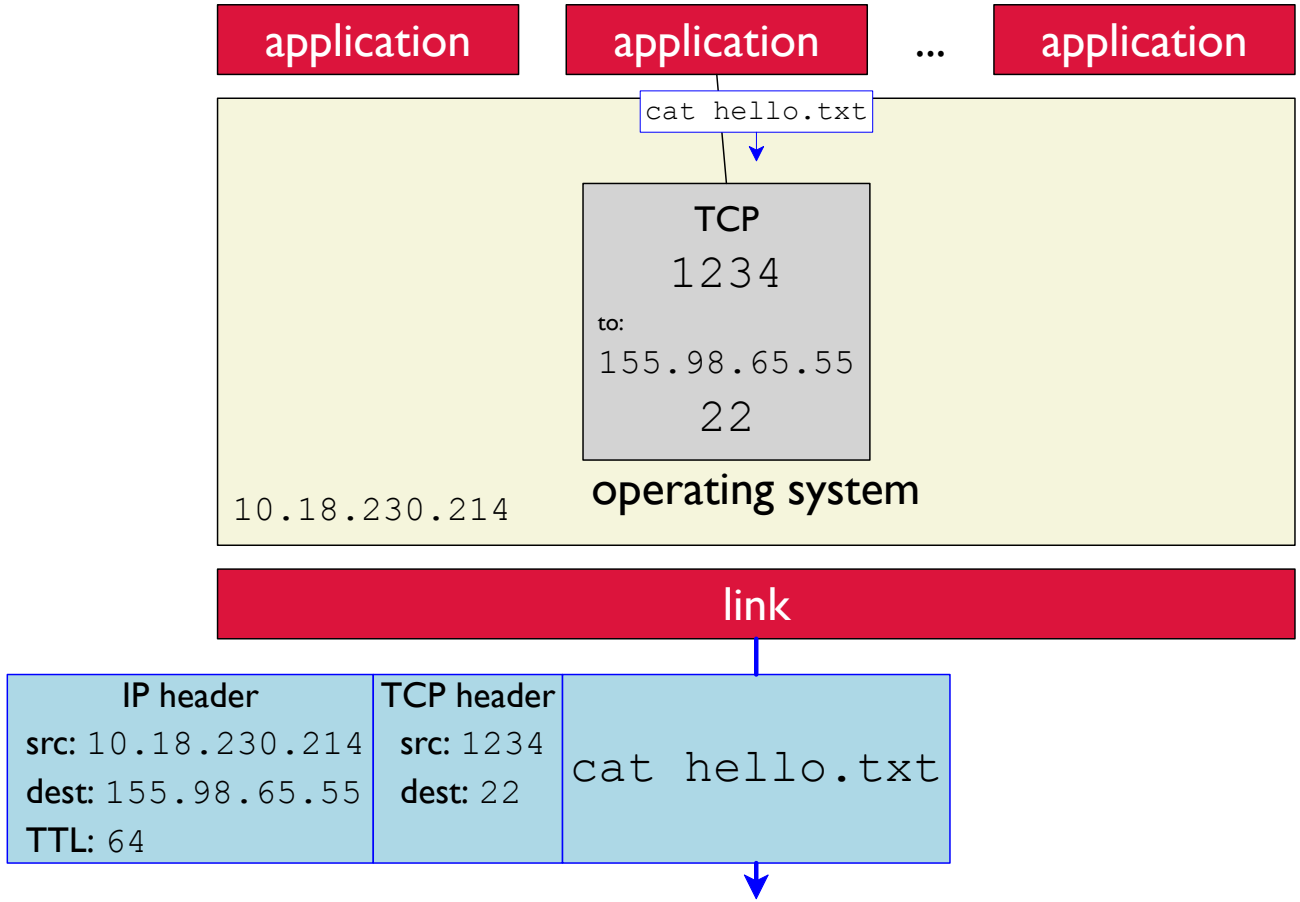
Using Sockets



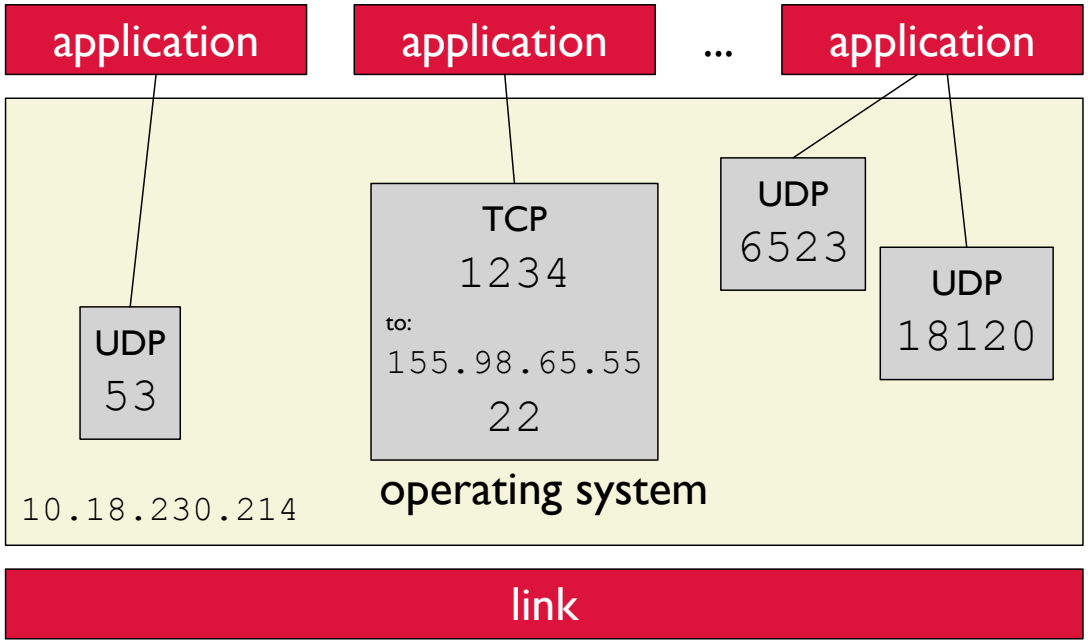
Using Sockets



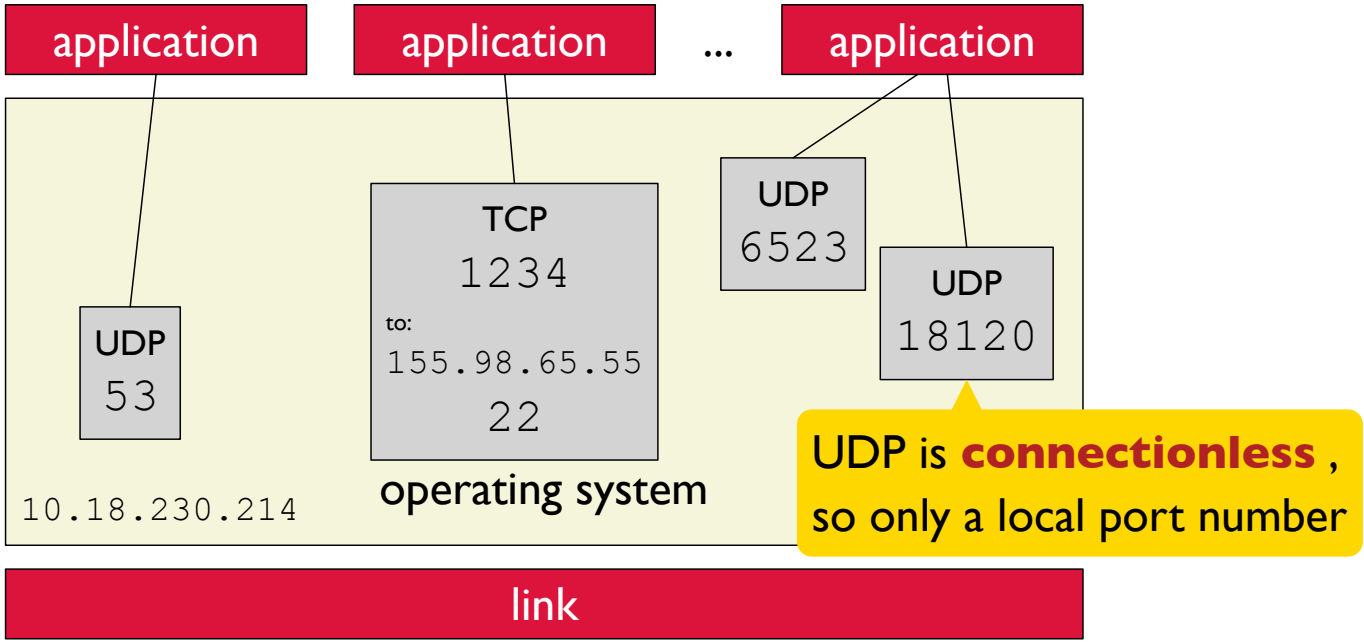
Using Sockets



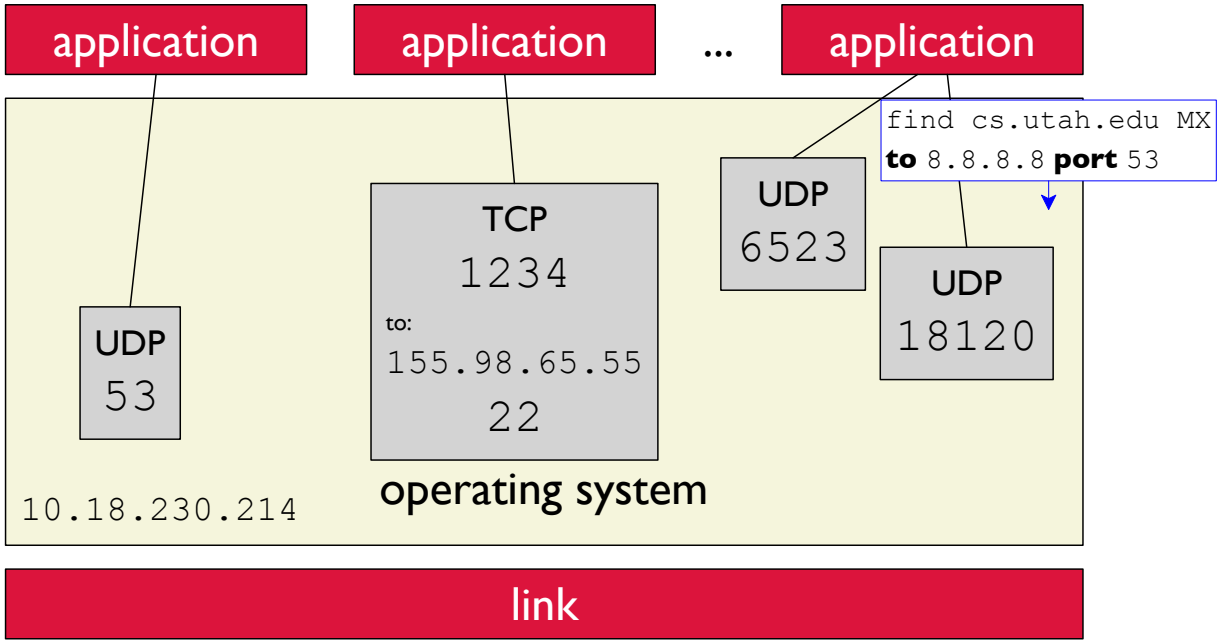
Using Sockets



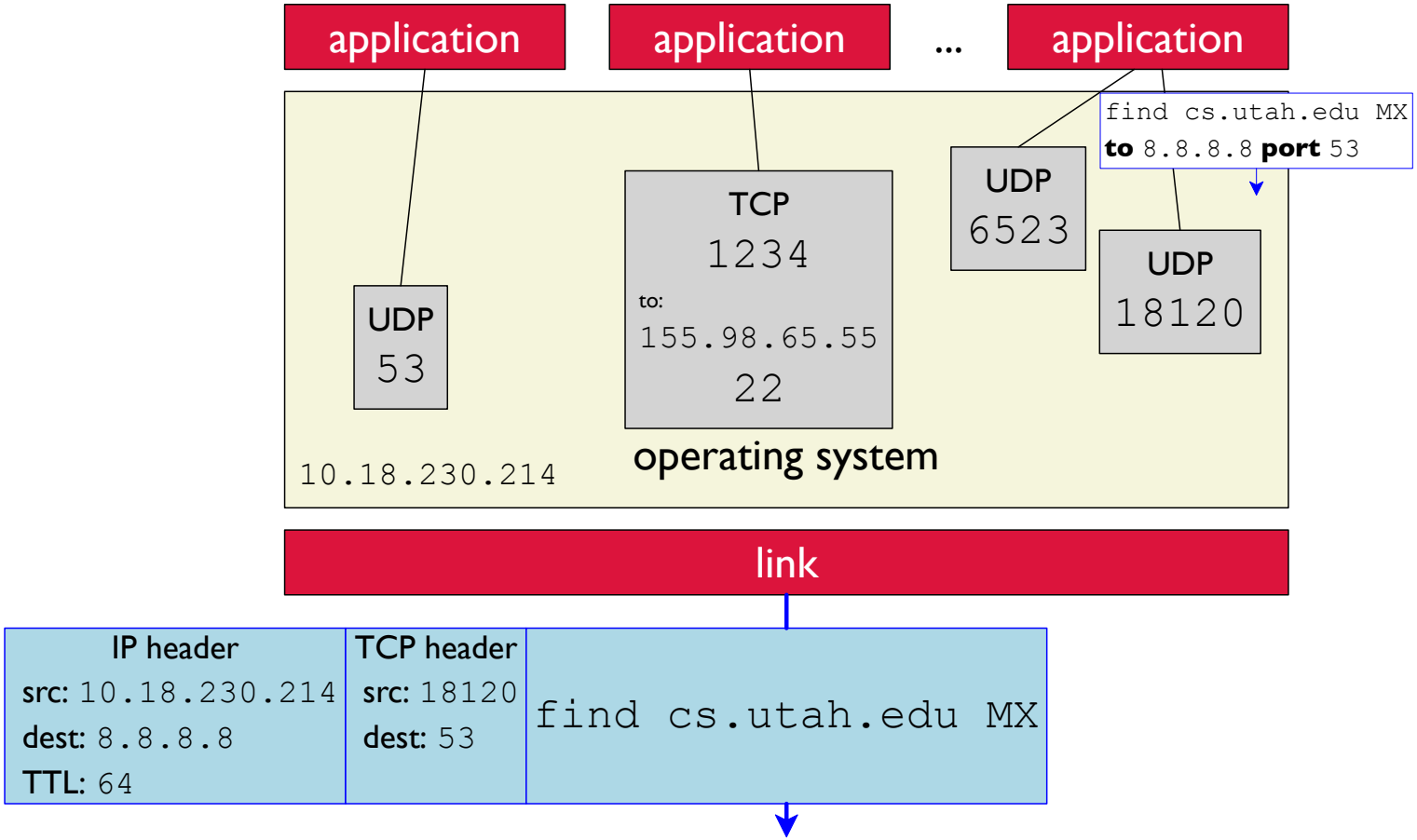
Using Sockets



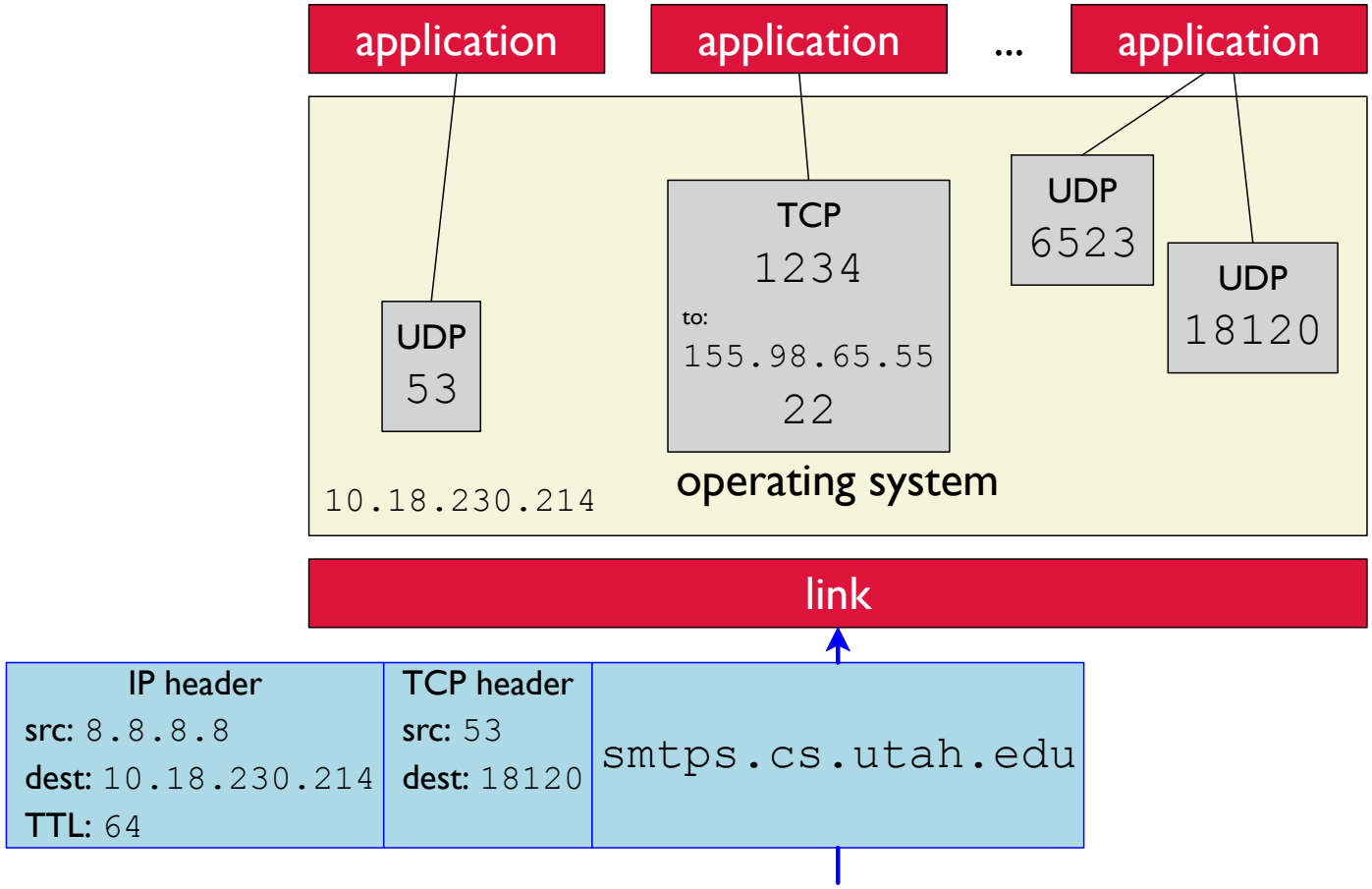
Using Sockets



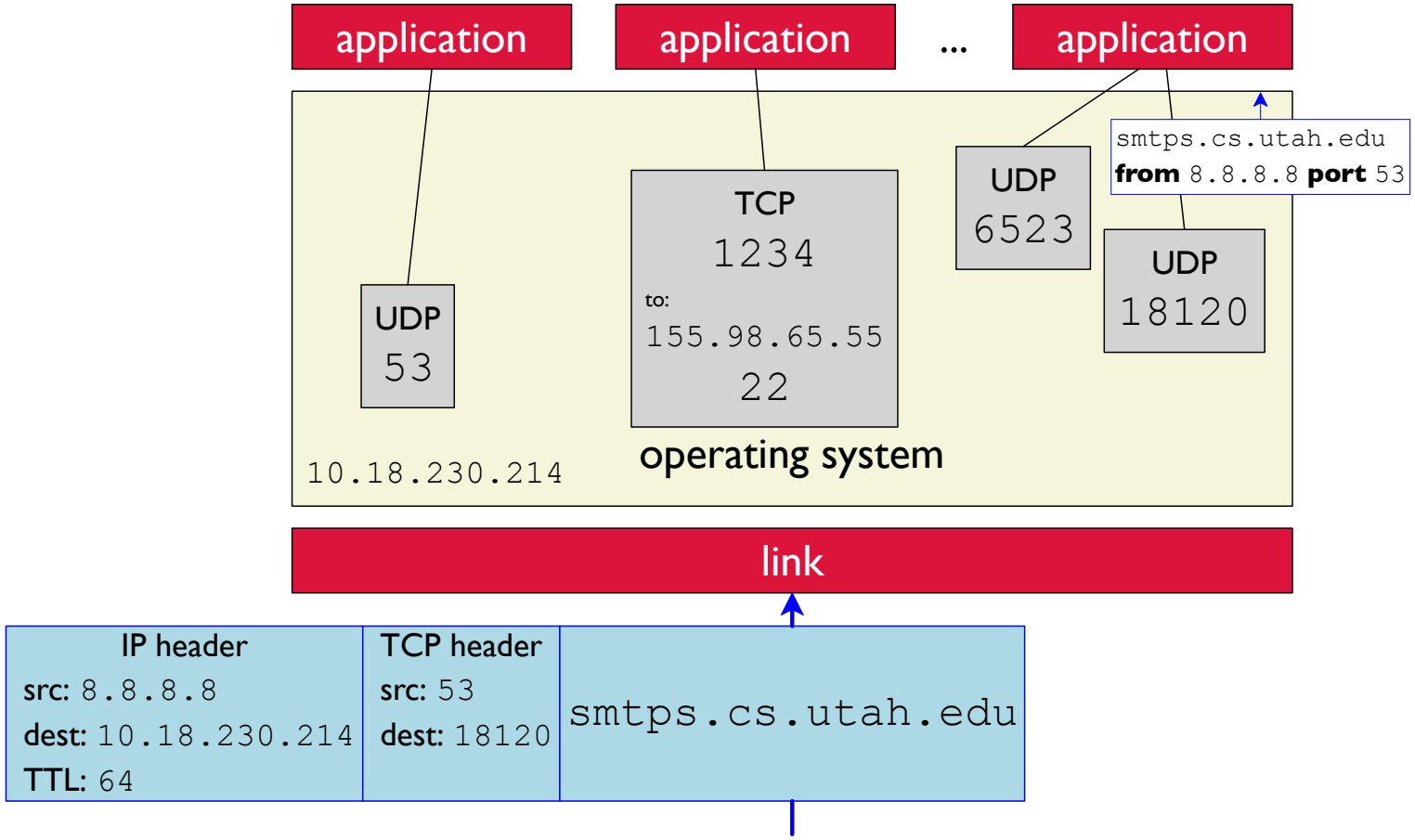
Using Sockets



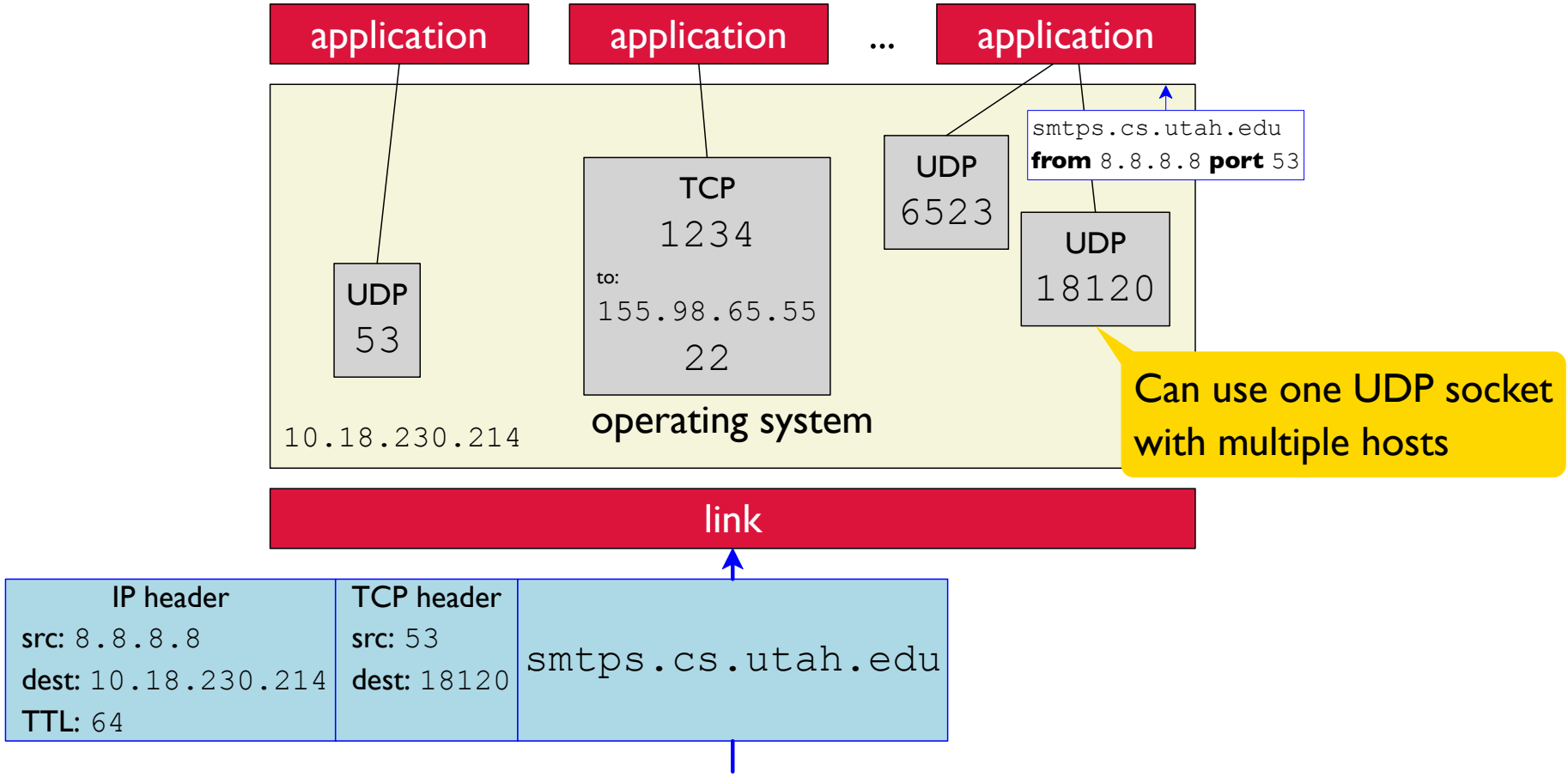
Using Sockets



Using Sockets



Using Sockets



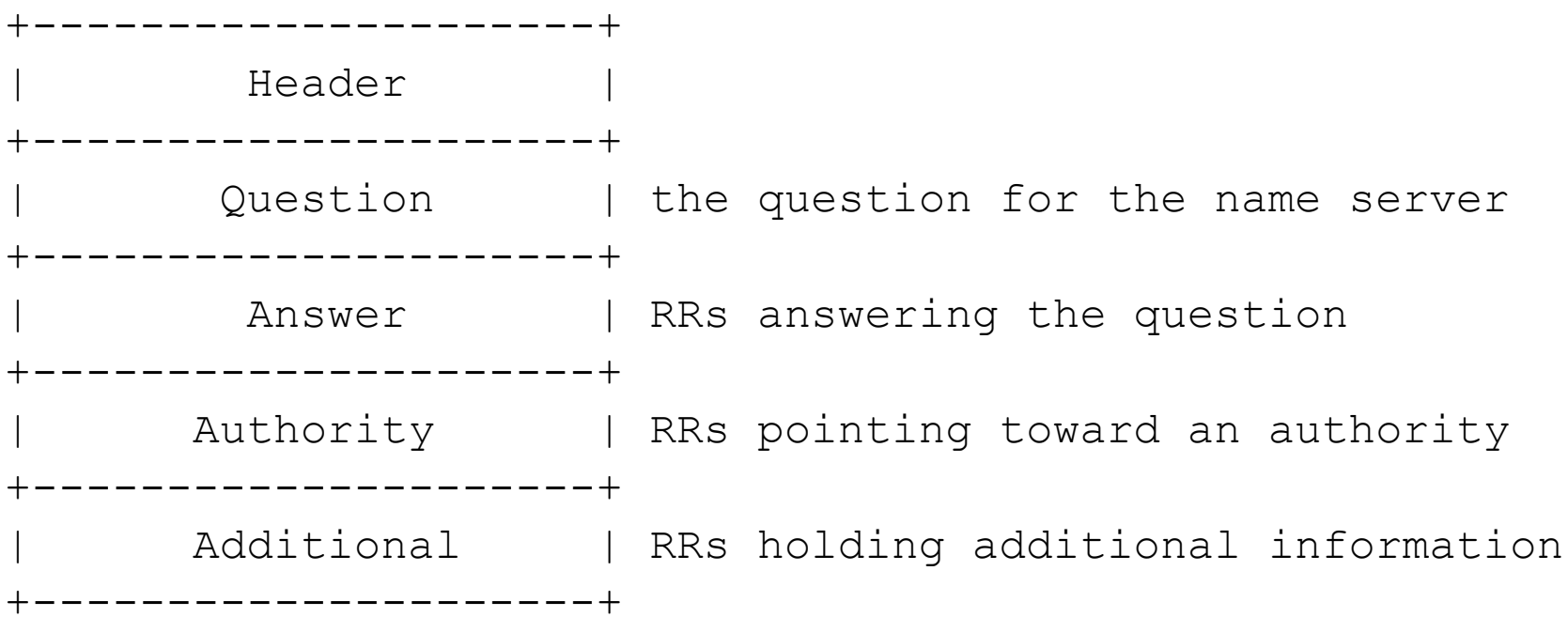
UDP Client

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class Main {
    public static void main(String[] args) throws IOException {
        int server_port = 5678;
        InetAddress server_host = InetAddress.getByName("localhost");
        System.out.println("Sending to " + server_host + " " + server_port);
        DatagramSocket socket = new DatagramSocket();

        System.out.println("I am " + socket.getLocalPort());

        byte[] data = new byte[3];
        data[0] = 10;
        data[1] = 20;
        data[2] = 30;
        DatagramPacket pkt = new DatagramPacket(data, data.length, server_host, server_port);
        socket.send(pkt);
    }
}
```

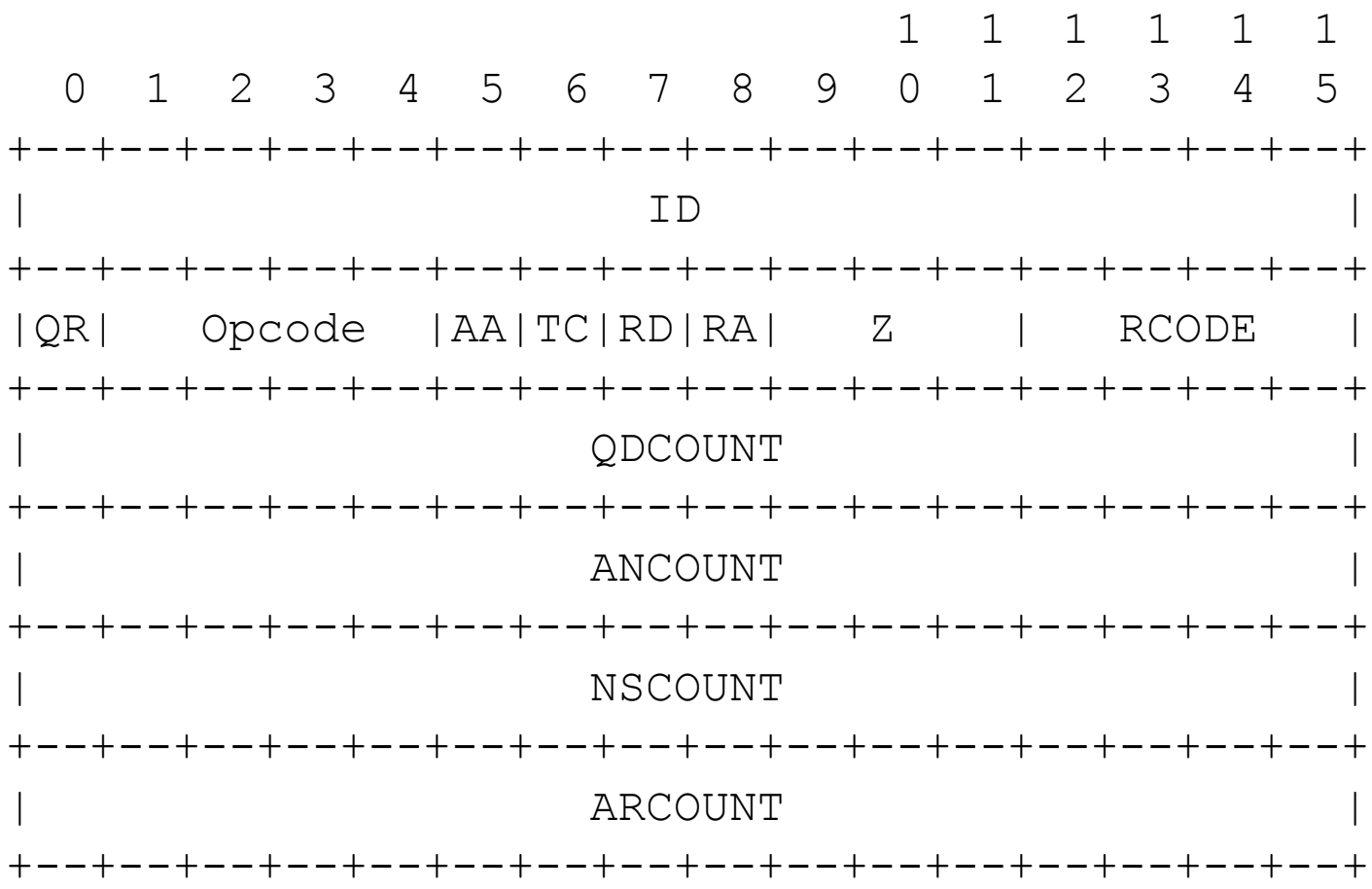
UDP Server

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;

public class Main {
    public static void main(String[] args) throws IOException {
        int server_port = 5678;
        System.out.println("Listening at " + server_port);
        DatagramSocket socket = new DatagramSocket(server_port);

        byte[] buffer = new byte[512];
        DatagramPacket pkt = new DatagramPacket(buffer, buffer.length);
        for (int count = 1; true; count++) {
            socket.receive(pkt); // <----- waits here
            System.out.println(count + " Heard from " + pkt.getAddress() + " " + pkt.getPort());
            for (int i = 0; i < pkt.getLength(); i++)
                System.out.printf(" %x", (int)buffer[i] & 0xFF);
            System.out.print("\n");
        }
    }
}
```

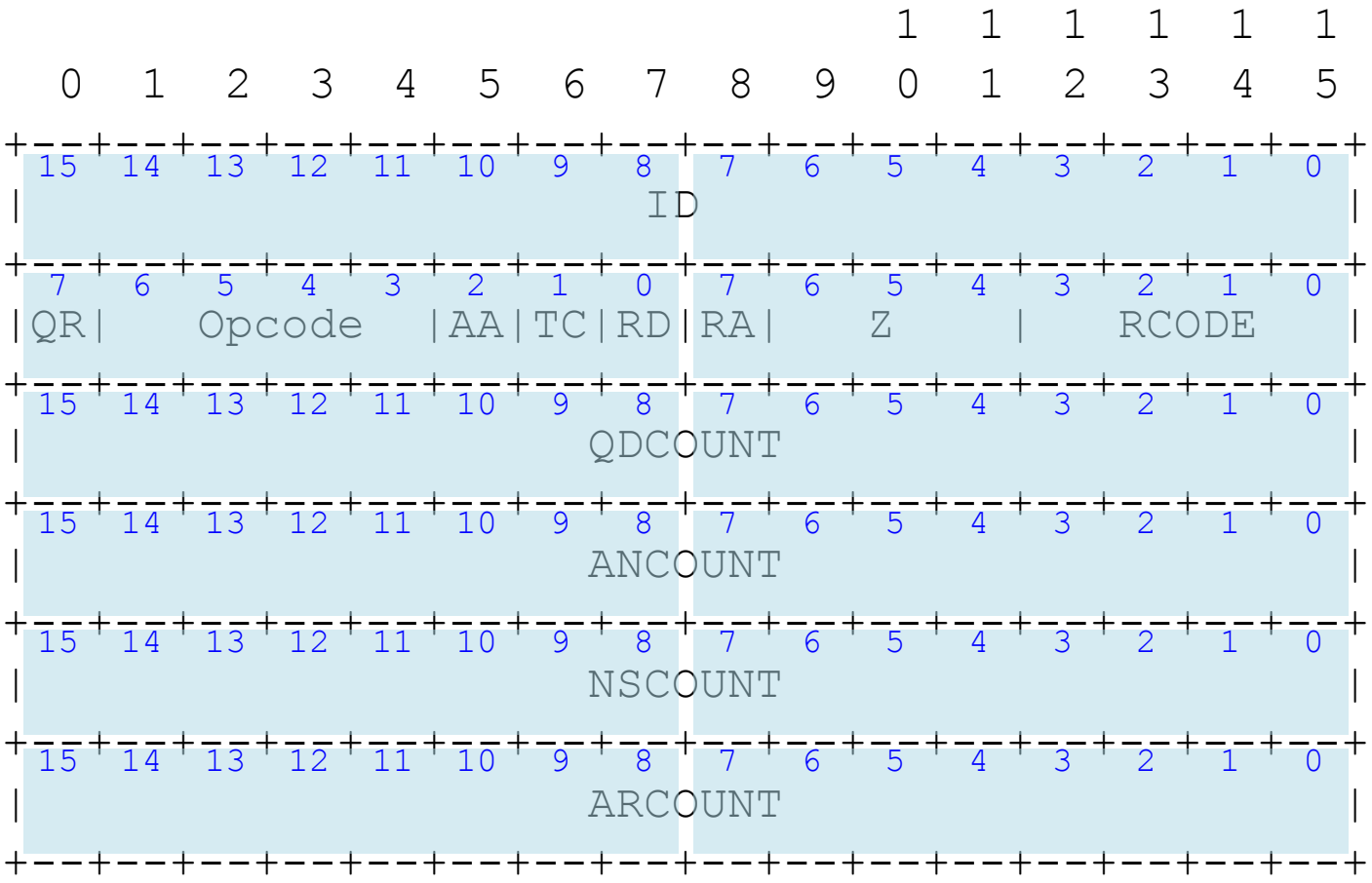
DNS Messages (RFC 1035)



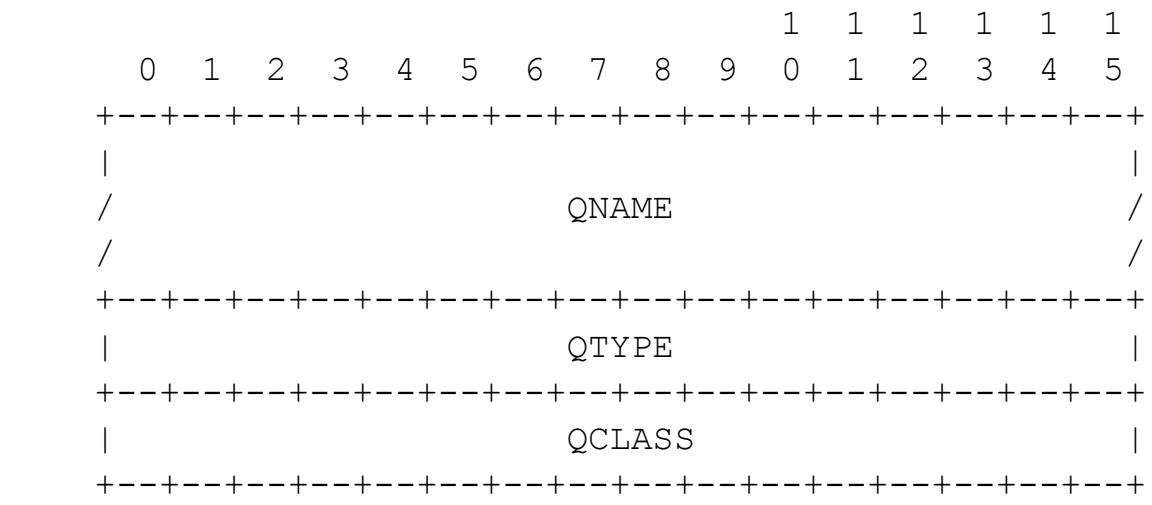
DNS Header (RFC 1035)



DNS Header (RFC 1035)



DNS Question (RFC 1035)

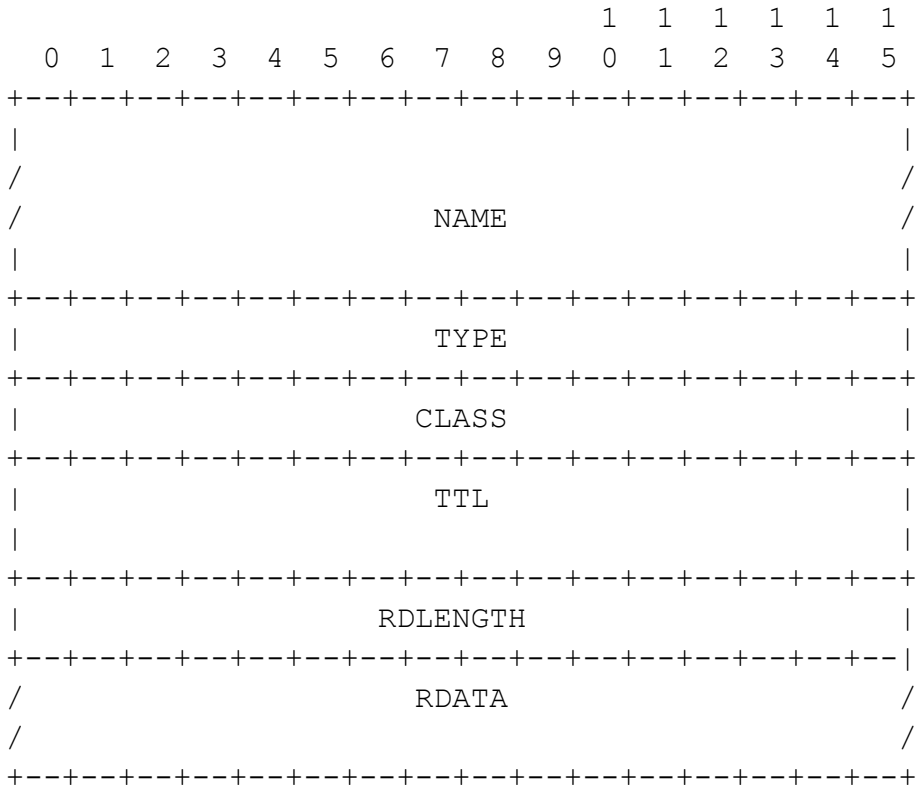


where:

QNAME a domain name represented as a sequence of labels, where each label consists of a length octet followed by that number of octets. The domain name terminates with the zero length octet for the null label of the root. Note that this field may be an odd number of octets; no padding is used.

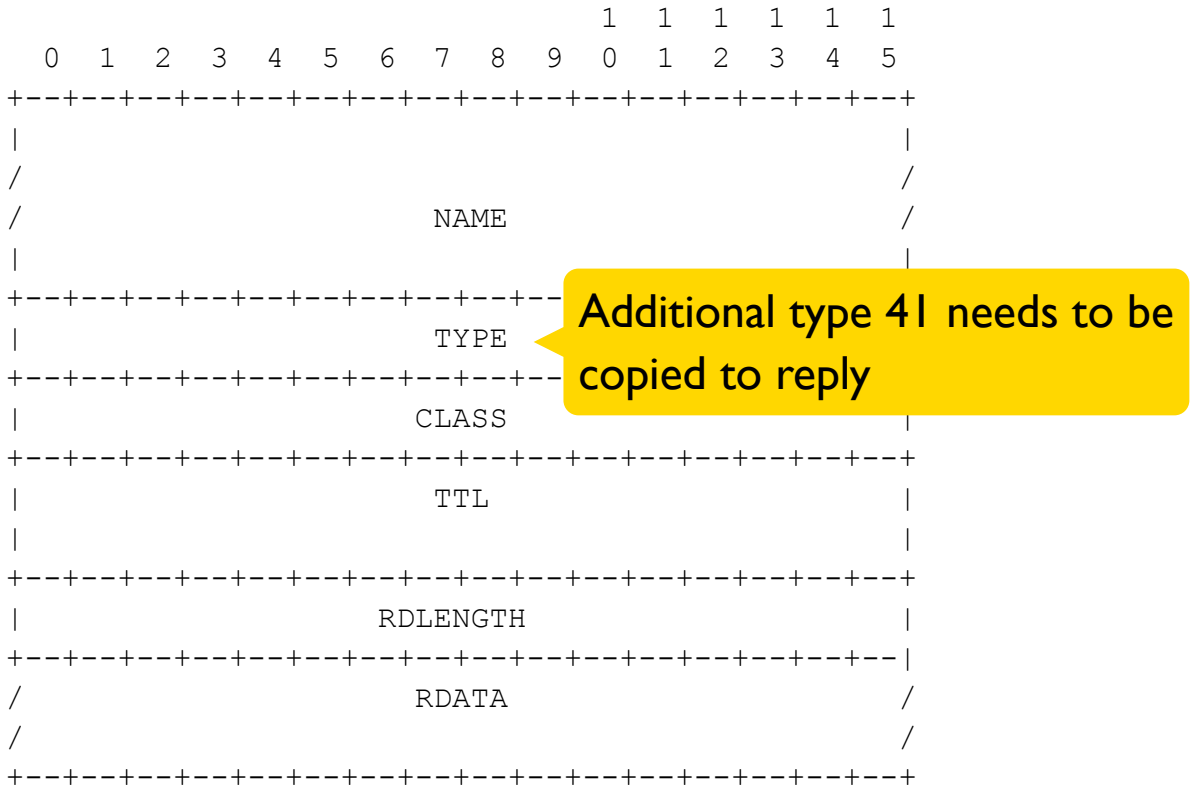
DNS Resource Record (RFC 1035)

Used for Answer, Authority, and Additional:



DNS Resource Record (RFC 1035)

Used for Answer, Authority, and Additional:



DNS Question (RFC 1035)

4.1.4. Message compression

In order to reduce the size of messages, the domain system utilizes a compression scheme which eliminates the repetition of domain names in a message. In this scheme, an entire domain name or a list of labels at the end of a domain name is replaced with a pointer to a prior occurance of the same name.

The pointer takes the form of a two octet sequence:

The first two bits are ones. This allows a pointer to be distinguished from a label, since the label must begin with two zero bits because labels are restricted to 63 octets or less.