IP, TCP, and HTTP Protocols

IEEE Northern VA Section Hands-On Workshop Series

October 29, 2016 Montgomery College

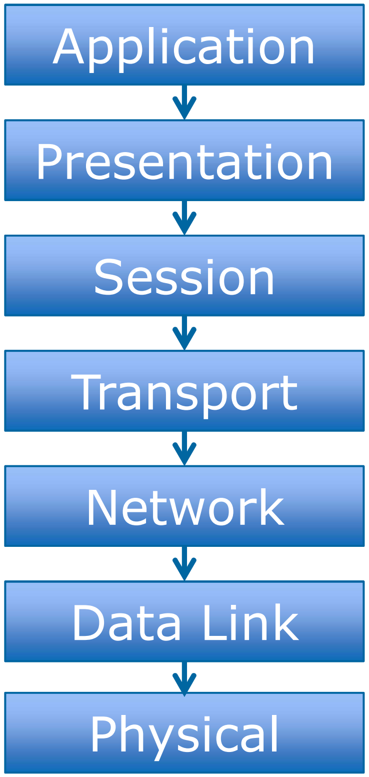
IEEE
Advancing Technology
for Humanity

Protocol Stacks

- Earlier we discussed architecture:
  - The major separable parts of a system and the way they communicate and control each other
- Sometimes network architectures are represented as a series of layers
  - Lower layers provide “services” to the upper layers
  - Some are implemented in hardware, some in software
  - Layers break up functionality into separable pieces, simplifying development and troubleshooting
  - Devices that communicate must have compatible corresponding layers

The ISO OSI 7 Layer Model

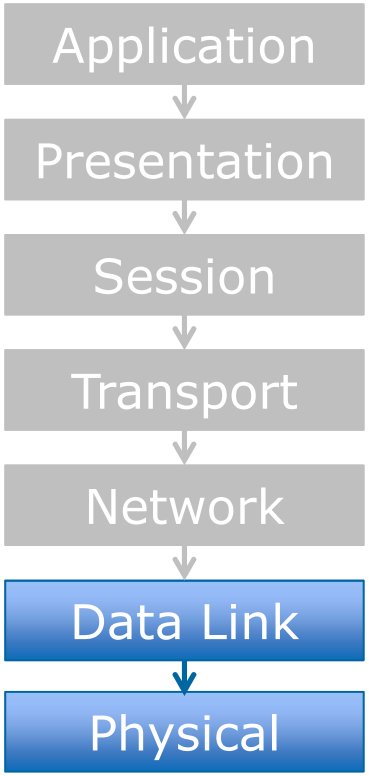
- The International Standards Organization’s Open Systems Interconnect Model defines seven layers



Application	Function: email, web, video chat, etc
Presentation	Character encoding, compression, encryption
Session	State that persists longer than individual exchanges
Transport	Multiplexing and optional ordering and reliability
Network	Addressing, routing, and fragmentation
Data Link	Symbol groups, error correction/detection
Physical	Amplitude and timing of voltage, radio, light

Physical and Data Link

- Data Link and Physical Layers are sometimes designed to work together



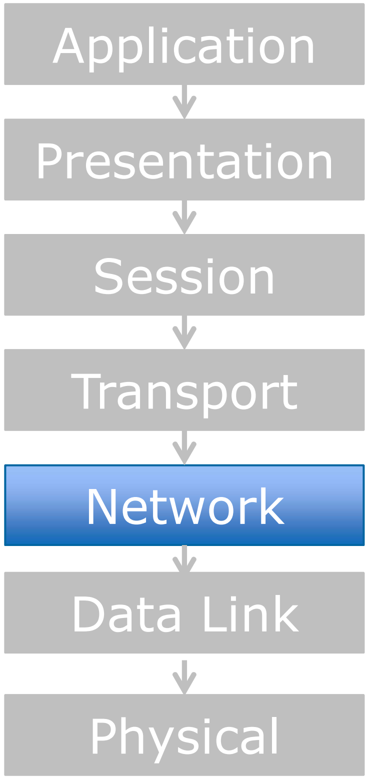
The most commonly used physical and data link protocols on the Internet are probably those based on the IEEE 802 family of LAN/MAN standards, including: 802.3 Ethernet (wired) and 802.11 "WiFi"

They define media access, the signaling, groups of bits called "frames", local addressing, and how to detect when there are errors.

Other examples include RS-232, RS-485 physical and HDLC data link layers.

Network Layer

- The Internet Protocol version 4 is the most widely used Network layer, though v6 is popular outside the US



Application

Presentation

Session

Transport

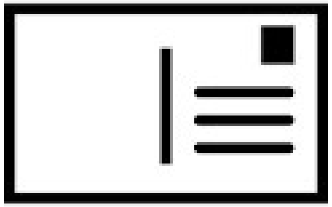
Network

Data Link

Physical

The network defines global addressing and how to break data into chunks called “packets”. Each is like a postcard – with a destination and return address and a maximum size. A lengthy message must be split into multiple cards.

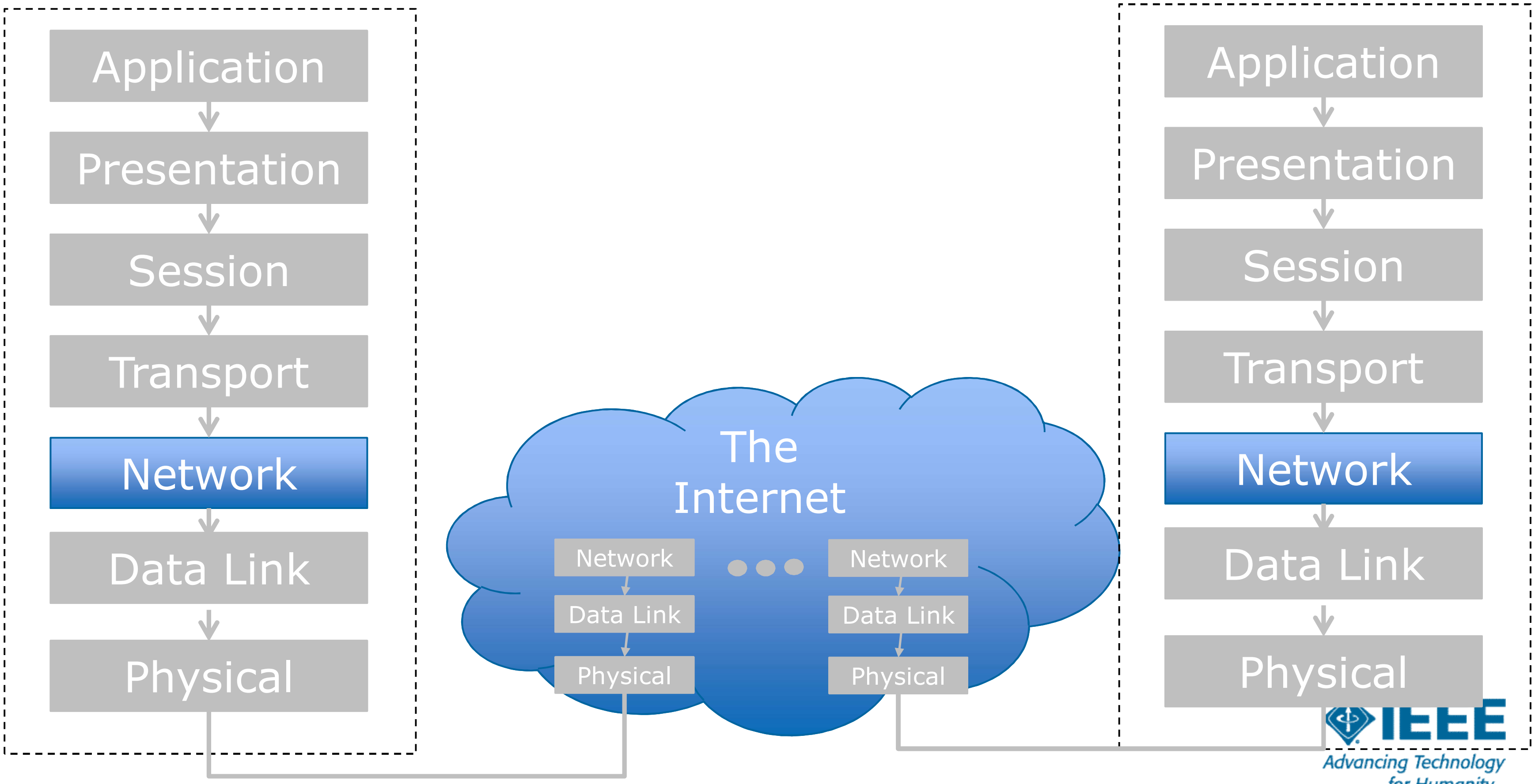
The Internet provides “best effort” delivery – there is no guarantee if or when a given packet will reach its destination.



IEEE
Advancing Technology
for Humanity

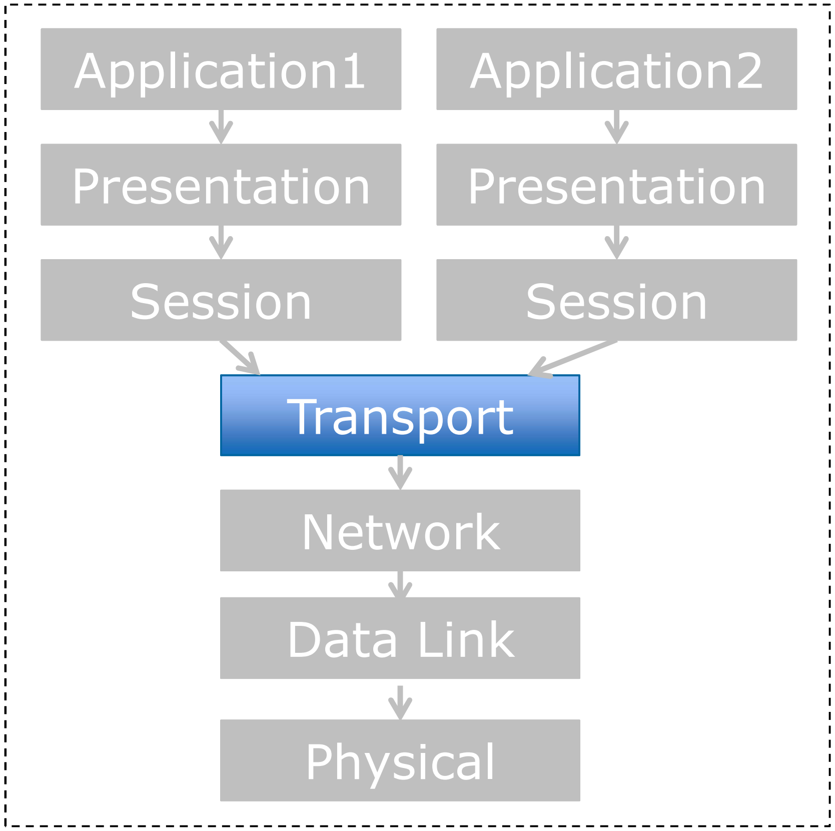
Network Layer

- The Internet works mostly at these three layers. Each router directs individual packets independently.



UDP Transport Layer

- Servers, personal computers, and even tablets and phones can run more than one application at a time.

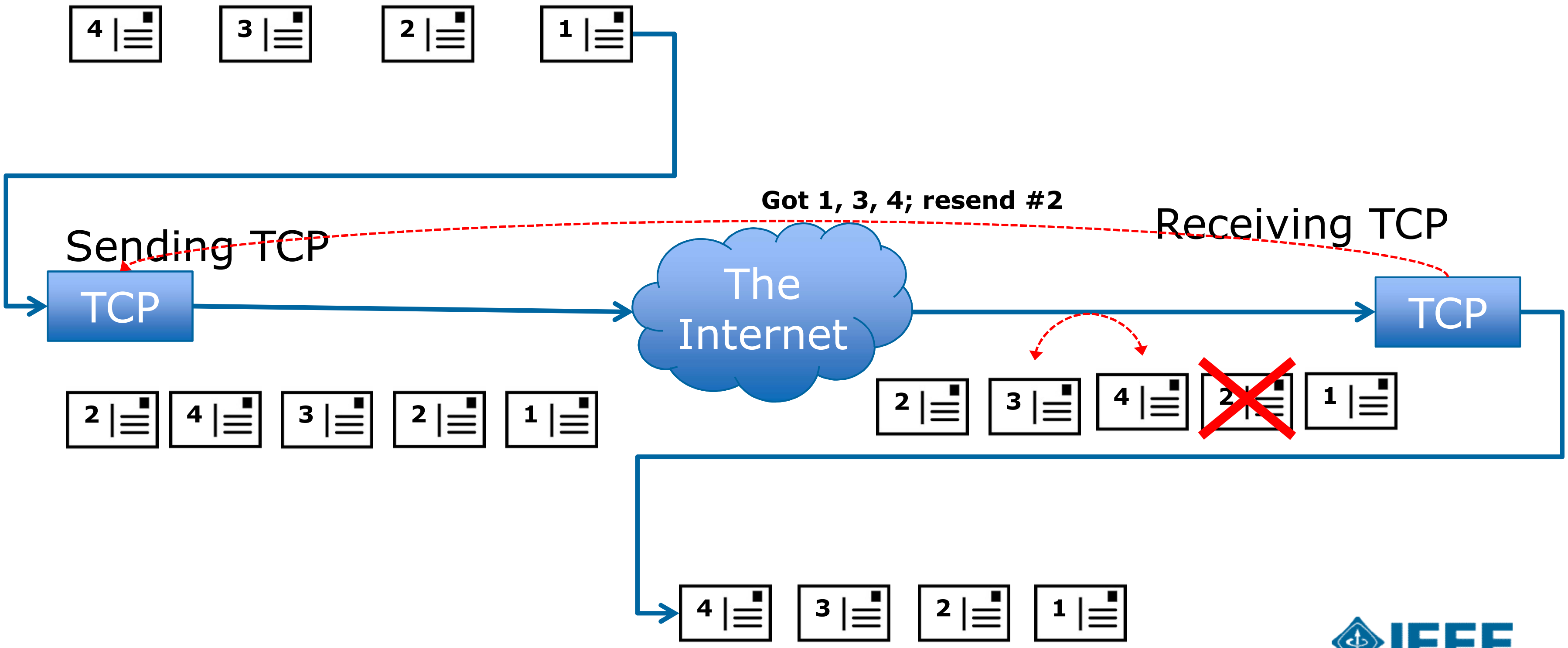


The User Datagram Protocol (UDP) allows multiple applications on a single host to share a network connection.

But packets may not get there, or not arrive in order, so it’s used for simple and/or time-sensitive applications.

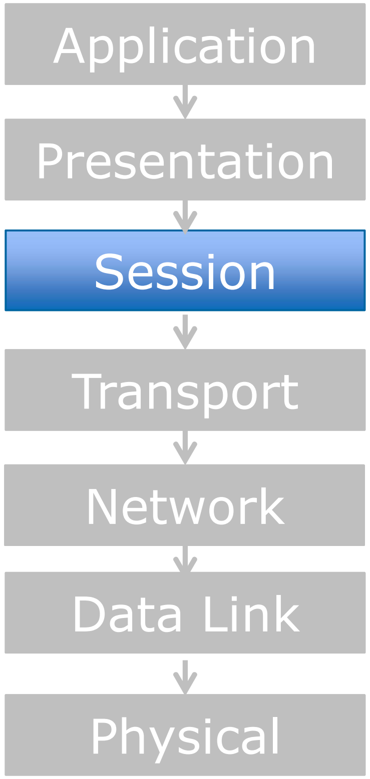
TCP Transport Layer

- TCP also multiplexes, plus it uses sequencing numbers and acknowledgements for reliable, ordered delivery



Session Layer

- Standards for this layer are not as common as those for others



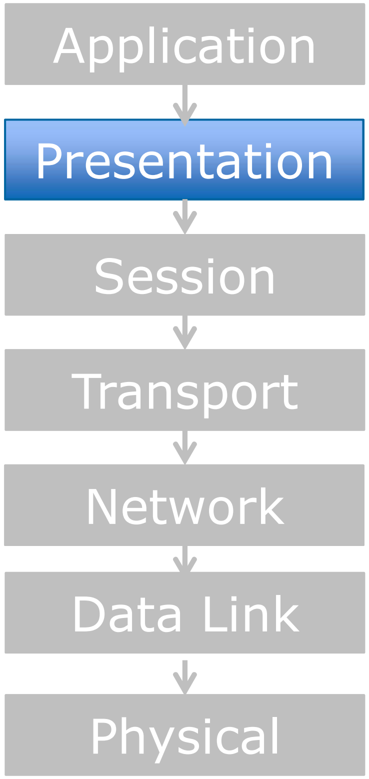
Many useful applications like chat consist of a series of short back-and-forth transmissions.

A session layer can maintain state across these sessions and resume in the case of a network interruption.

The applications we consider incorporate some of this state in other layers; there is no explicit session layer.

Presentation Layer

- There are many standards for data representation, compression, and encryption

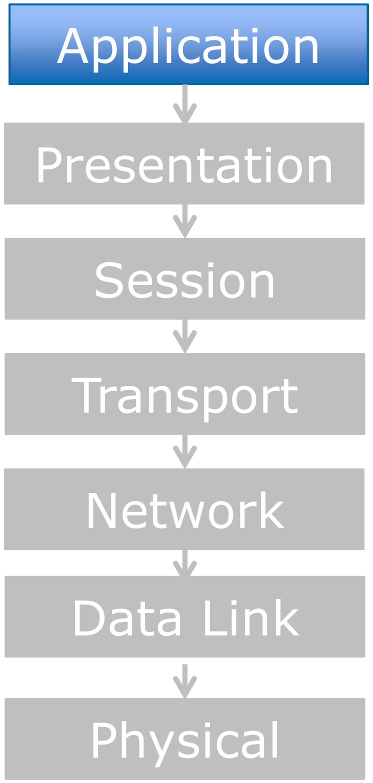


There are two general approaches for representing data: humanly readable “ASCII” vs. binary. The former is often easier to develop and troubleshoot, while the latter may be more efficient.

As with the session layer, the applications we consider often incorporate this into other layers with the exception of SSL/TLS widely used to encrypt web and email traffic.

Application Layer

- The rapid growth of Internet applications is attributed in part to the use of humanly readable protocols

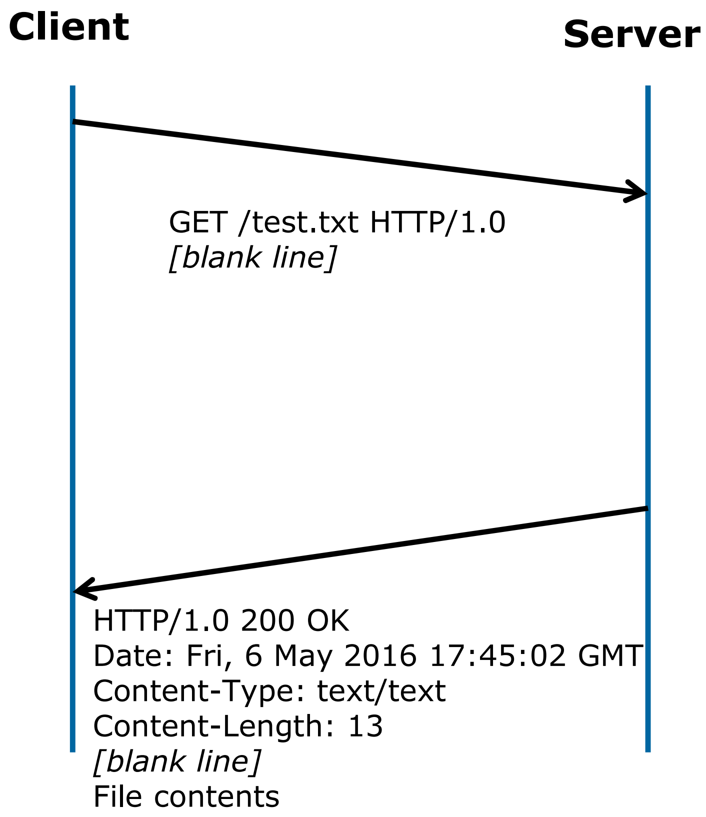


Think of the application protocol as the conversation you'd have on the phone to accomplish a task. Some like seeing if a store is open are very simple. Others like arranging a vacation with flights, rental cars, and hotels require a complicated exchange.

Internet applications use many different protocols, but some newer applications are able to reuse the popular Hypertext Transfer Protocol (HTTP) implemented by all web browsers.

IEEE Advancing Technology for Humanity

Minimal HTTP 1.0 GET Request



Client

Server

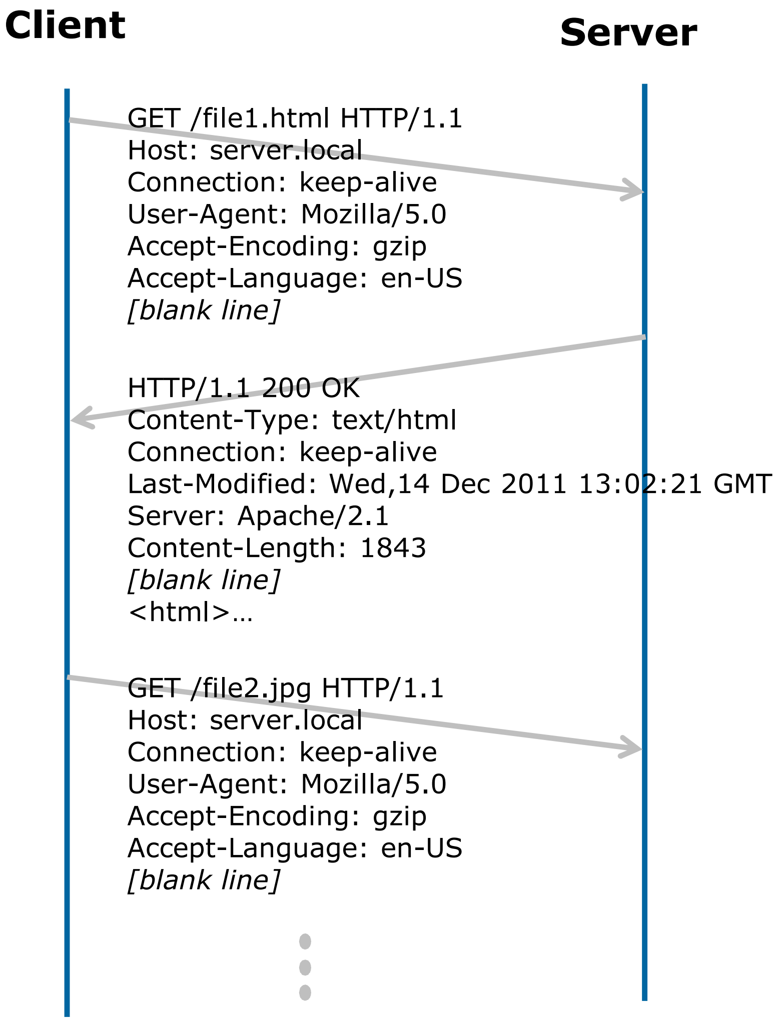
GET /test.txt HTTP/1.0
[blank line]

HTTP/1.0 200 OK
Date: Fri, 6 May 2016 17:45:02 GMT
Content-Type: text/text
Content-Length: 13
[blank line]
File contents

Timeline diagrams can illustrate application protocols. Time increases downward, and arrows point from sender to receiver.

HTTP uses TCP to provide reliable, ordered delivery. In version 1.0 there is a single request per connection.

HTTP 1.1 GET Request Example



```
GET /file1.html HTTP/1.1
Host: server.local
Connection: keep-alive
User-Agent: Mozilla/5.0
Accept-Encoding: gzip
Accept-Language: en-US
[blank line]
```

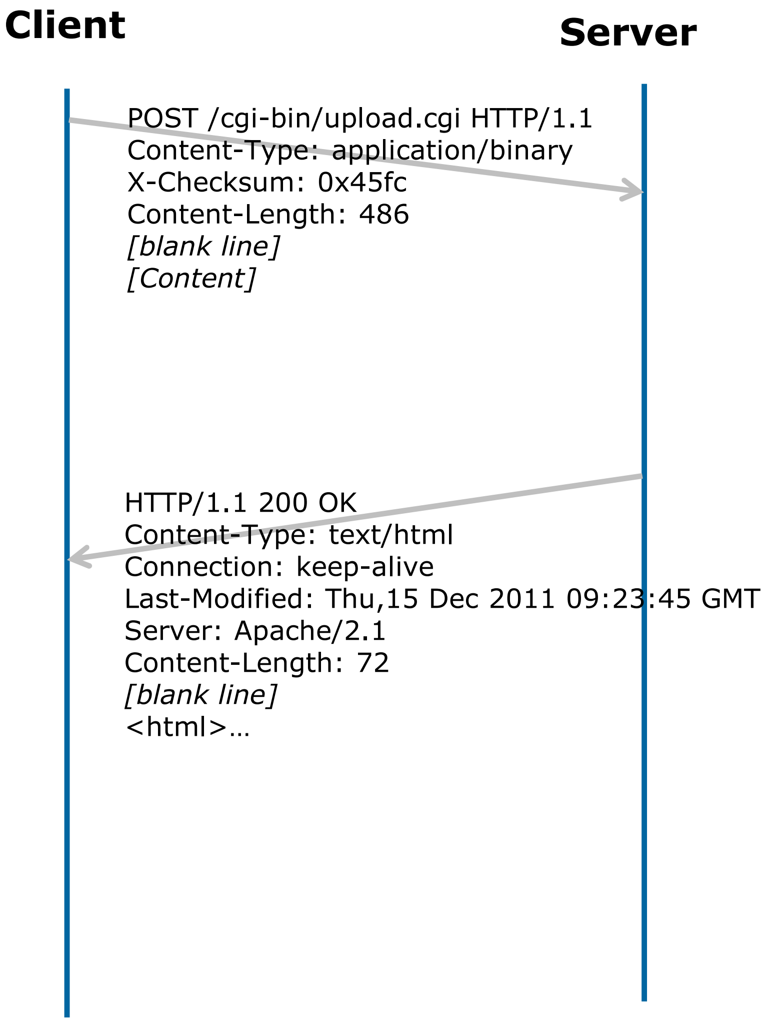
```
HTTP/1.1 200 OK
Content-Type: text/html
Connection: keep-alive
Last-Modified: Wed,14 Dec 2011 13:02:21 GMT
Server: Apache/2.1
Content-Length: 1843
[blank line]
<html>…
```

```
GET /file2.jpg HTTP/1.1
Host: server.local
Connection: keep-alive
User-Agent: Mozilla/5.0
Accept-Encoding: gzip
Accept-Language: en-US
[blank line]
```

As web pages became more complicated, incorporating content from multiple servers, it became very inefficient to retrieve only one file per TCP connection.

HTTP 1.1 allowed multiple requests per connection. Additional headers were defined to allow a single server to act as multiple virtual hosts.

HTTP 1.1 POST Request



Client

Server

```
POST /cgi-bin/upload.cgi HTTP/1.1
Content-Type: application/binary
X-Checksum: 0x45fc
Content-Length: 486
[blank line]
[Content]
```

```
HTTP/1.1 200 OK
Content-Type: text/html
Connection: keep-alive
Last-Modified: Thu,15 Dec 2011 09:23:45 GMT
Server: Apache/2.1
Content-Length: 72
[blank line]
<html>…
```

The examples so far show how clients retrieve information using GET requests.

Clients can also send files using POST requests and/or extension headers. This is how the “Upload” button usually works.

IoT Application Protocols

- IoT devices often have slower processors and limited memory
- Several efficient application protocols are intended for or used by IoT devices
  - CoAP: Binary over UDP
  - MQTT: Binary over TCP, publish/subscribe
  - AMQP: Binary over TCP, queue-oriented
  - XMPP: XML over TCP or HTTP

Why We Use HTTP

- HTTP already used browser to server
  - Server does not need to support additional protocol
- HTTP is easy to read and understand
  - Simple to develop and troubleshoot
- The ESP8266 has enough resources!
  - It’s a powerful little device that includes the HTTP and TCP protocol libraries
- So the Weather Station will use HTTP to send its measurements to the server

Questions?