OSI Model
The Open Systems Interconnection (OSI) model is a 7-layer view of networking that abstracts and encapsulates the functionality of each component of networking.

OSI Layer 1: ________________

OSI Layer 2: ________________

OSI Layer 3: ________________

IPv4, Packet Length: 0x00c6 (198 bytes); Source IP: ac.16.b4.a3 (172.22.180.163); Destination IP: 12.dc.95.a6 (18.220.149.166)

OSI Layer 4: ________________

Port :0xbafa (47866) connecting to Port :0x0050 (80); Checksum 0x079e; Timestamp: 0x814630a0 (2168860832)

OSI Layer 5, 6, and 7: ________________

GET / HTTP/1.1\r\nUser-Agent: Wget/1.20.3
(\n"linux-gnu")\r\nAccept: */\r\nAccept-Encoding: \r
identity\r\nHost: waf.cs.illinois.edu\r\nConnection: Keep-Alive\r\n\r
Full Packet Journey
Consider an HTTP request you are making from your browser to waf.cs.illinois.edu (just as I did using tcpdump). Assuming we start from your laptop on a WiFi network:

Network Layer (Layer 3) Protocol: Internet Protocol (IP)
- The network layer provides __________ communication.
- When on the Internet, every host relies on the IP protocol:
  - IP (IPv4) Address:
  - IPv6 Addresses:
Transport Layer (Layer 4) Protocols:
Two protocols for ______________________ communications:

1. 

2. 

Features of TCP and UDP:

<table>
<thead>
<tr>
<th>Feature / Metric</th>
<th>TCP</th>
<th>UDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication Between Processes: HTTP Web Services</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

One of the primary ways that processes will communicate is via “web services” -- applications that communicate using the HTTP protocol.

The HTTP protocol has two parts:

Protocol Part #1: 

```
1 POST /extract HTTP/1.1\r\n2 Host: localhost:5000\r\n3 User-Agent: curl/7.68.0\r\n4 Accept: */*\r\n5 Content-Length: 3046796\r\n6 \r\n7 { 3,046,796 bytes payload } 
```

Protocol Part #2: 

```
1 HTTP/1.0 200 OK\r\n2 Content-Length: 3044143\r\n3 Content-Type: image/gif\r\n4 Last-Modified: Mon, 28 Sep 2021 21:16:13\r\n5 Cache-Control: public, max-age=43200\r\n6 Expires: Tue, 24 Mar 2022 09:16:12 GMT\r\n7 ETag: “1601327773.0845277-3044143-32865”\r\n8 Server: Werkzeug/0.16.1 Python/3.8.2\r\n9 Date: Mon, 28 Sep 2020 21:16:12 GMT\r\n10 \r\n11 { 3,044,143 bytes of content } 
```

Request Packet Organization:
- Line Delineation:
- Start Line (Line 1): HTTP method (verb), target, and version
- Request Headers (Lines 2+):
- Payload (or sometimes just the “Data”):

In general, the request and response follows the same format with only one major exception:
- Response “Status Line” (Line 1):

```
1xx Informational  2xx Success  3xx Redirection  4xx Client Error  5xx Server Error
100: Continue     200: OK        304: Not Modified  400: Bad Request  500: Internal
201: Created     304: Not Modified  408: File Not Found  500: Server Error
```