# LECTURE 33

# NETWORK ARCHITECTURE

MCS 260 Fall 2021

Emily Dumas

# REMINDERS

- Homework 11 due tomorrow at 10am central
- Central time is now GMT-6
- Worksheet 12 available
- Project 4 description coming Wed, will involve topic selection. Will have 7 days to choose topic.
- No synchronous lecture Fri; will post lecture video instead

# NETWORKING

A **computer network** is a group of computers that are connected to one another in a way that allows them to exchange data.

There is a vast worldwide computer network called the **internet** to which we are all connected right now.

# SAMPLE INTERACTION

I open a web browser, enter

```
http://example.com/
```

and press Enter. Soon, a web page is displayed.

Today we'll discuss the layers of networking technology that are used in this process.

First we need some additional concepts and terminology.

# NETWORK CONCEPTS

Computers in a network are **hosts** or **nodes**.

Each host contains a device to send and receive data over the network. This is a **network interface controller** or **NIC**.

e.g. ethernet adapters (wired) and wifi adapters (wireless) are NICs.

Most modern networks (including the internet) are based on **packets**, i.e. groups of bits that move from one device to another as a unit.

The internet supports many ways to communicate between hosts for different purposes. These are **protocols**, i.e. rules defining how the communication takes place.

Protocols are a relatively high level concept. At a low level, everything comes down to packets that move from one NIC to another.

# URLS

The string `http://example.com/` is a URL or **Uniform Resource Locator**. It has several parts:

- **http** — the protocol (communication method) to use. Here it is the Hypertext Transfer Protocol (HTTP), which is the primary protocol for the web.
- **example.com** — the name of the host where this resource is located (a **web server**)
- The / at the end — the name of the resource we are requesting from example.com

# LOADING EXAMPLE.COM: OVERVIEW

The name example.com is looked up in a directory, yielding a 4-byte numeric **IP address** like 93.184.216.34 (byte values separated by dots).

My computer opens a channel to talk to 93.184.216.34 (called a **TCP connection**). The channel is a bit like a file.

By writing to this channel, my computer asks for "/".

Focus on one step where HTTP is used:

> *By writing to this channel, my computer asks for "/".*

*This is a complex operation. Let's dig into the details a bit more.*

# IN MORE DETAIL

HTTP is a protocol based on sending text commands. The command

```
GET /
```

will ask for the contents of `/` (which is really `http://example.com/` since we're talking to example.com.)

So we send this text over the channel.

*This is a complex operation. Let's dig into the details a bit more.*

`GET /` is translated into a packet of data to send to 93.184.216.34. The string itself is in this packet, along with a bunch of control data.

This packet is sent, it passes through a number of intermediate hosts along its way, and it is received by the web server at example.com.

(The web server sends a packet back to acknowledge receipt, so we determine there is no need to resend.)

*This is a complex operation. Let's dig into the details a bit more.*

A packet needs to go to IP address 93.184.216.34.

The NIC in my computer can only send packets to other NICs on the local network, each of which is identified by a **hardware address** (or MAC).

This IP address is not on our local network. Therefore, we send this packet to the **router**, whose MAC the OS knows.

The router will figure out what to do next (e.g. forward the packet to some other part of the internet).

*This is a complex operation. Let's dig into the details a bit more.*

We have a packet ready to go to the router.

It is a sequence of bytes, including lots of control data:

```
11 11 11 11 11 11 22 22 22 22 22 22 08 00 45 00   ......"""""".."E.
02 49 7e f8 40 00 40 06 79 c3 0a 00 00 21 5d b8   .I~.@.@.y....!].
d8 22 ba a4 00 50 90 19 f9 ae ae e9 d2 ea 80 18   ."...P..........
01 f6 42 2f 00 00 01 01 08 0a 03 1e 08 54 d9 98   ..B/.........T..
ab d5 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31   ..GET / HTTP/1.1
0d 0a 48 6f 73 74 3a 20 65 78 61 6d 70 6c 65 2e   ..Host: example.
63 6f 6d 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a   com..Connection:
20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 43 61 63    keep-alive..Cac
68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6d 61 78 2d   he-Control: max-
61 67 65 3d 30 0d 0a 44 4e 54 3a 20 31 0d 0a 55   age=0..DNT: 1..U
```

The NIC changes the voltage on certain wires of the network cable in a pattern corresponding to the bits of this packet. The router at the other end of the cable monitors these wires and reconstructs the packet.

# NETWORK LAYERS

- **Application layer**: Programs request operations that involve the network in some way.
- **Transport layer**: A communication channel is created between two hosts.
- **Network layer**: A packet moves from one device to another, possibly passing through many devices along the way. (IP based)
- **Link layer**: A packet moves from one device to another using a direct connection. (MAC based)
- **Physical layer**: Voltages on a wire, radio signals, etc.

# NETWORK LAYERS IN THIS EXAMPLE

- **Application layer**: Get http://example.com/ using HTTP.
- **Transport layer**: The text "GET /" is sent along a TCP channel to example.com.
- **Network layer**: A packet is sent to IP 93.184.216.34.
- **Link layer**: My computer's NIC sends a packet to the router's MAC. The router handles the next "hop".
- **Physical layer**: The NIC generates electrical signals on wires in the network cable.

# NEXT TIME

Application layer network operations in Python:

Making HTTP requests with the `urllib` module.

# REFERENCES


- Section 4.4 in Brookshear and Brylow discusses the layer model for internet protocols. They merge the physical and link layers into a single layer.


# REVISION HISTORY

- 2021-11-08 Initial publication
- 2021-11-08 Typos corrected