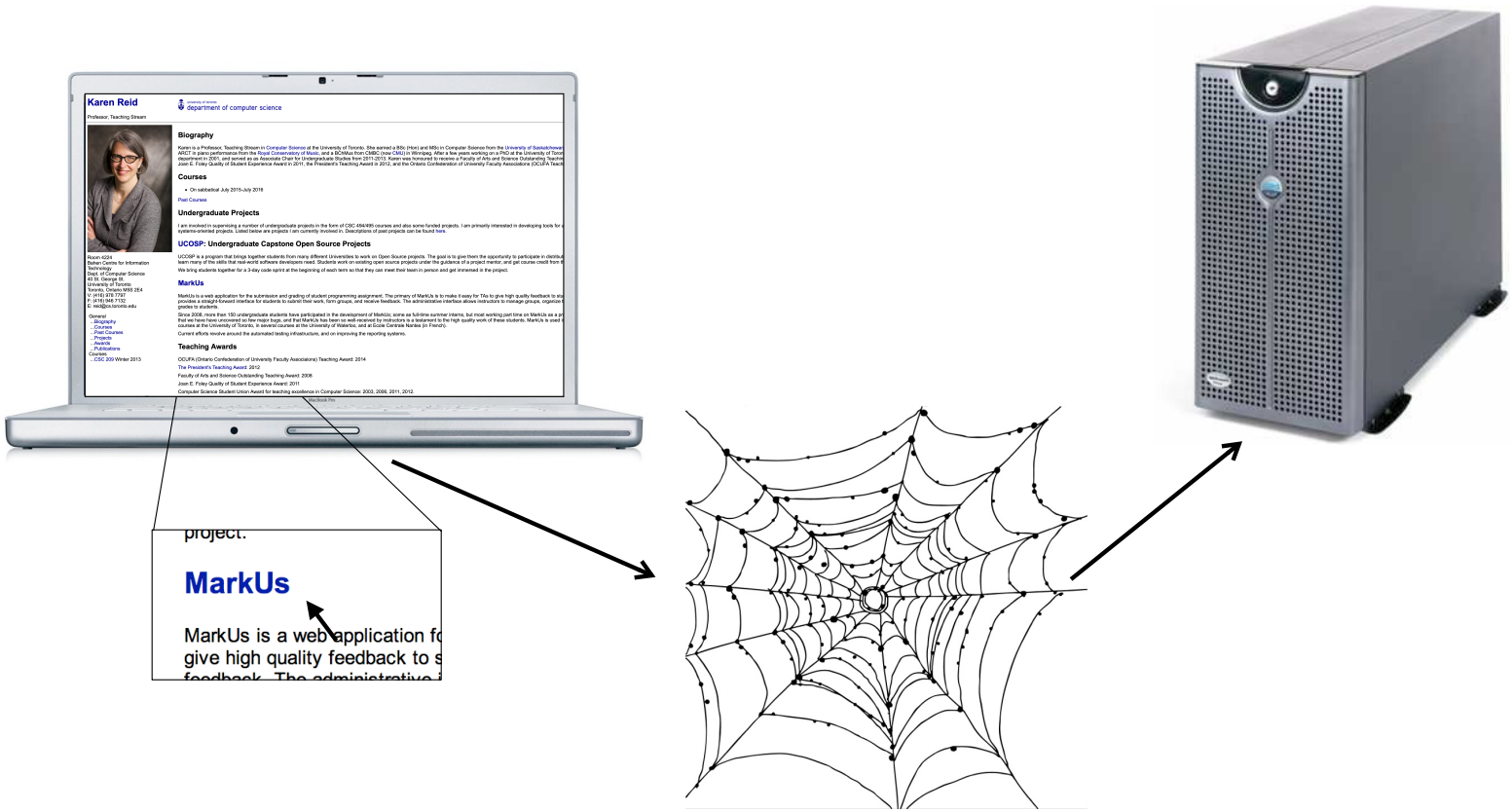
What makes the World Wide Web work?

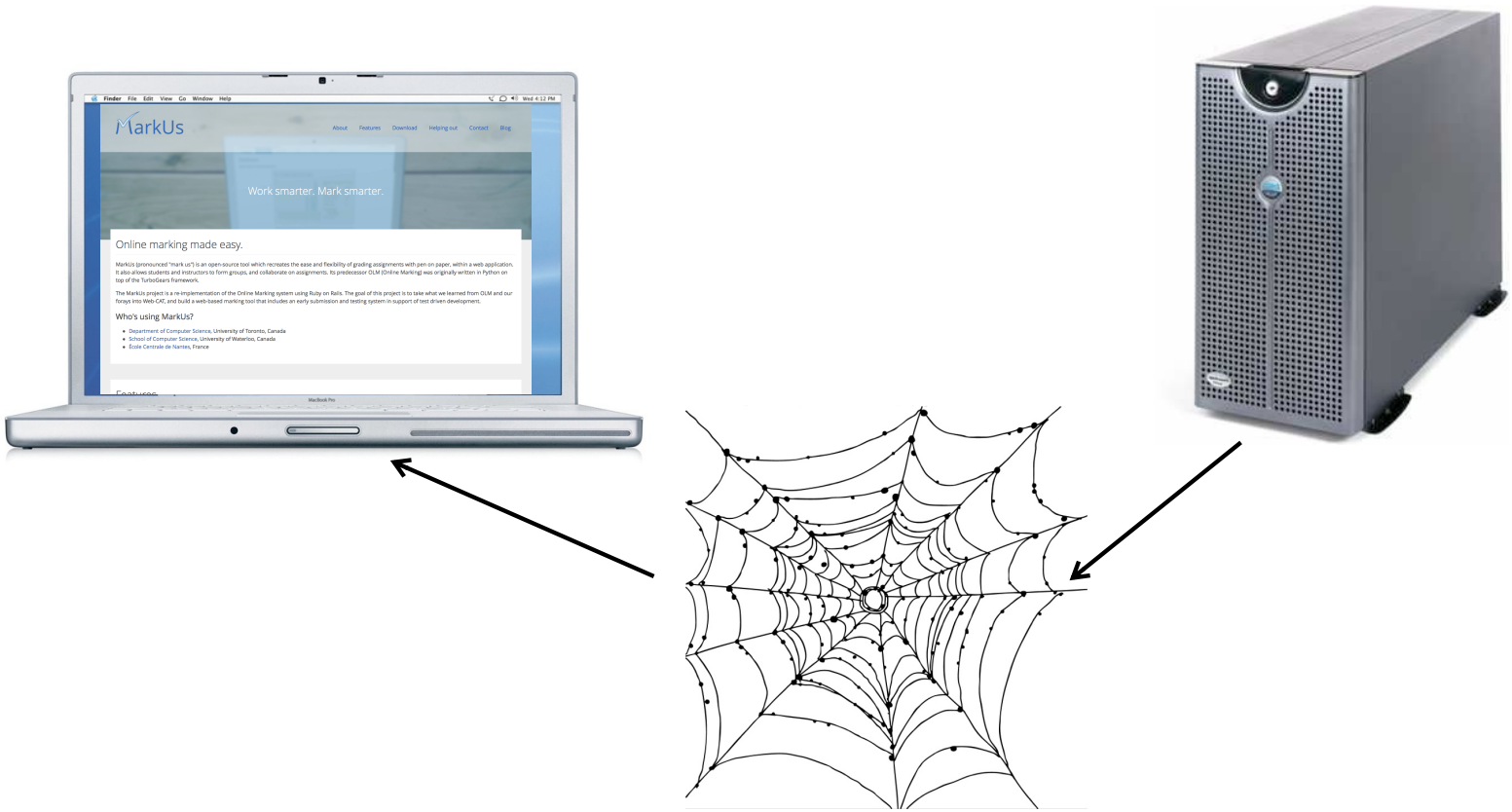
Karen Reid

image:Peter Werkman (www.peterwerkman.nl)

Simple Web Request



Response


MarkUs
About
Features
Download
Helping out
Contact
Blog
Work smarter. Mark smarter.
Online marking made easy.
MarkUs (pronounced "mark us") is an open-source tool which recreates the ease and flexibility of grading assignments with pen on paper, within a web application. It also allows students and instructors to form groups, and collaborate on assignments. Its predecessor OLM (Online Marking) was originally written in Python on top of the TurboGears Framework.
The MarkUs project is a re-implementation of the Online Marking system using Ruby on Rails. The goal of this project is to take what we learned from OLM and our forays into Web-CAT, and build a web-based marking tool that includes an early submission and testing system in support of test driven development.
Who's using MarkUs?
Department of Computer Science, University of Toronto, Canada
School of Computer Science, University of Waterloo, Canada
École Centrale de Nantes, France

The Request

- How do we tell the web server what we want?
- How do we even find the web server?
- How do the web server and browser talk to each other?

HTTP Request

request

```
GET / HTTP/1.1
Host:markusproject.org
...
```

reply

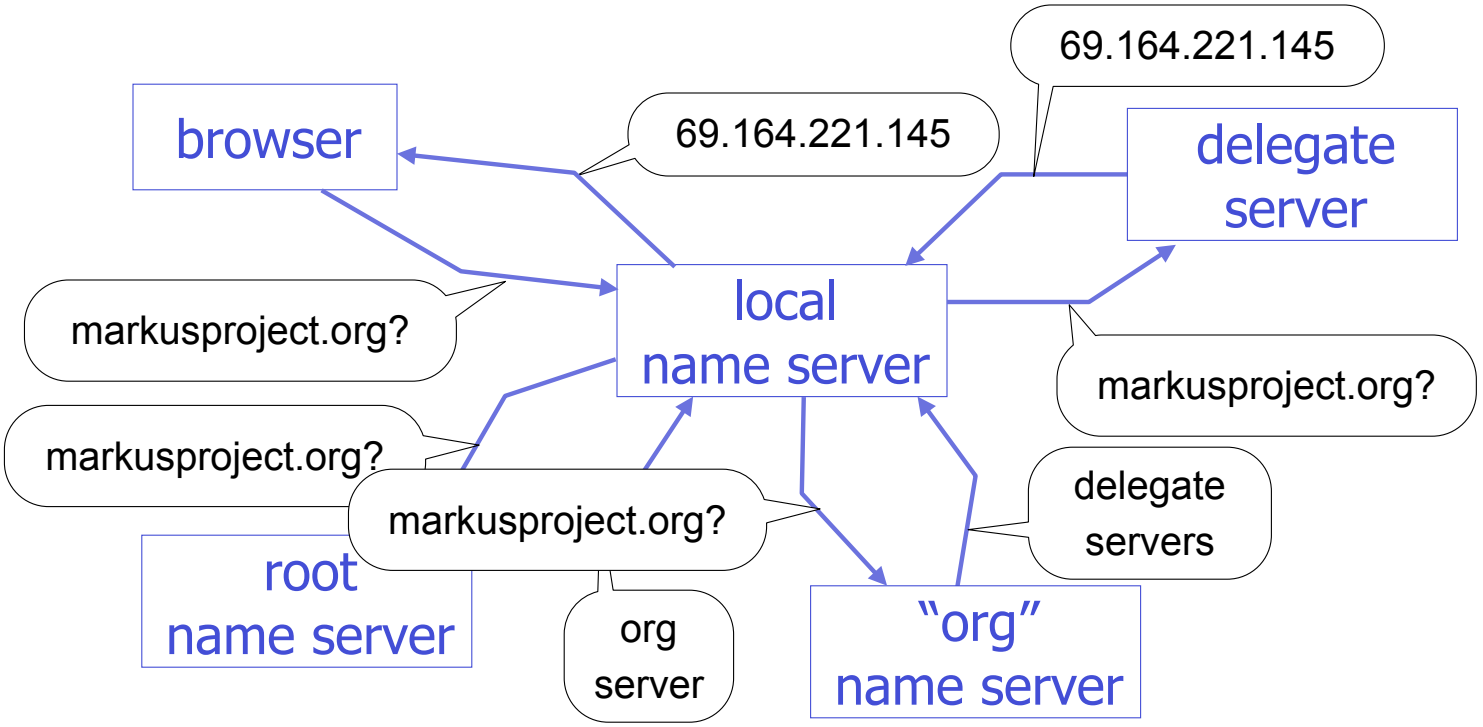
```
HTTP/1.1 200 OK
Date: Tue, 13 Mar 2017
Server: Apache/2.2.22(Debian)
Content-Type: text/html
```

How do we find the server?

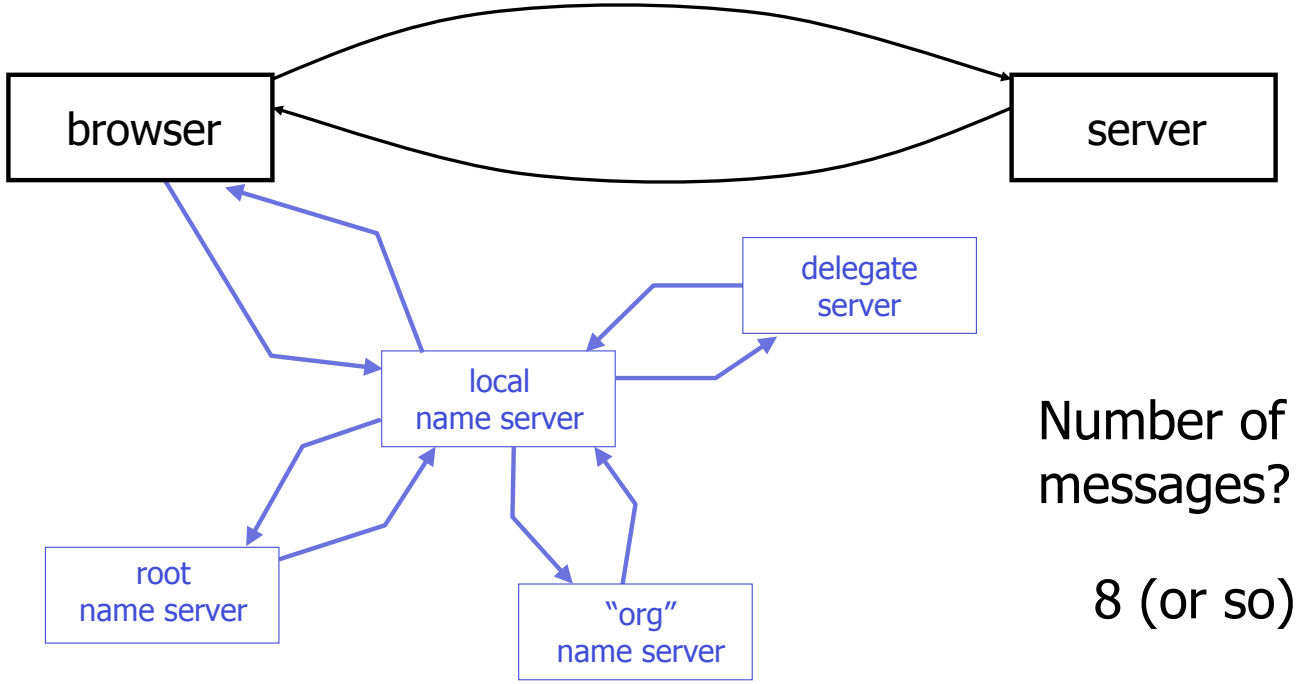
- Every computer on the Internet has an Internet address.
- Called an IP address (Internet Protocol)
- An IP address is 4 numbers separated by dots.

markusproject.org = 69.164.221.145

Domain Name Servers



This is getting complicated!



Number of messages?

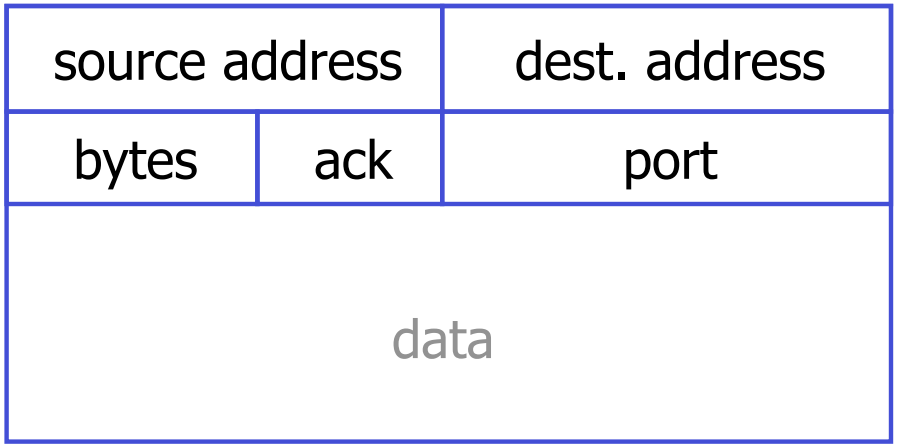
8 (or so)

Now what?

- Okay, we have the address.
- What do we do with it?
- Let's look at how two computers communicate.
- HTTP is a high-level protocol
- HTTP is specific to the web.
- Computers communicate for many reasons.
  - Network needs to support many different protocols

TCP/IP

- Transmission Control Protocol.
- Tells us how to package up the data.



try nslookup

Addresses and Ports

- A socket pair is the two endpoints of the connection.
- An endpoint is identified by an IP address and a port.
- IPv4 addresses are 4 8-bit numbers:
  - 128.100.31.200 = wolf
- Ports
  - because multiple processes can communicate with a single machine we need another identifier.

Think of a port like a room number in a building

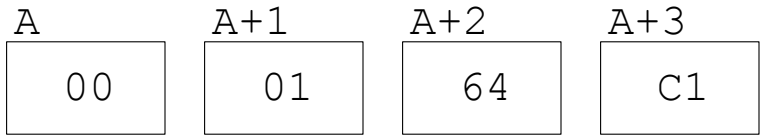
More on Ports

- Well-known ports: 0-1023
  - 80 = http
  - 22 = ssh
  - 23 = telnet
  - 21 = ftp
  - 25 = smtp (mail)
  - 194 = irc
- Registered ports: 1024-49151
  - 666 = Doom (first online first-person shooter)
  - 26000 = quake
  - 19132 = minecraft: bedrock edition (unofficial)
- Dynamic (private) ports: 49152-65535

Network byte order?

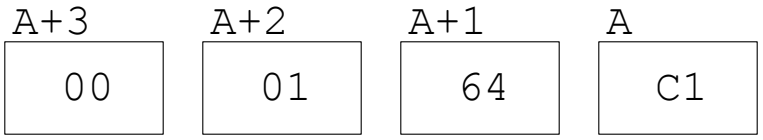
- Big-endian

91,329 =



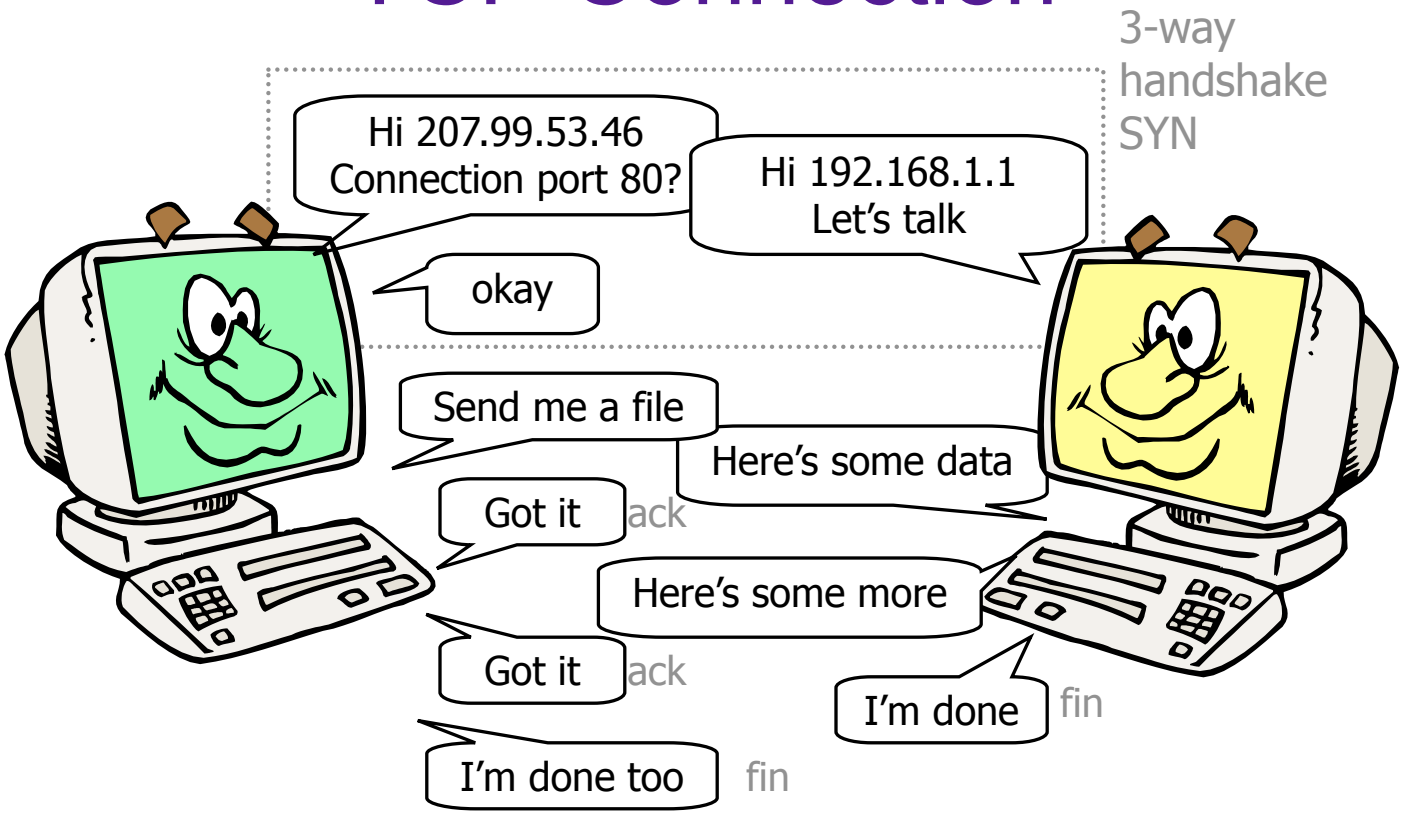
- Little-endian

91,329 =



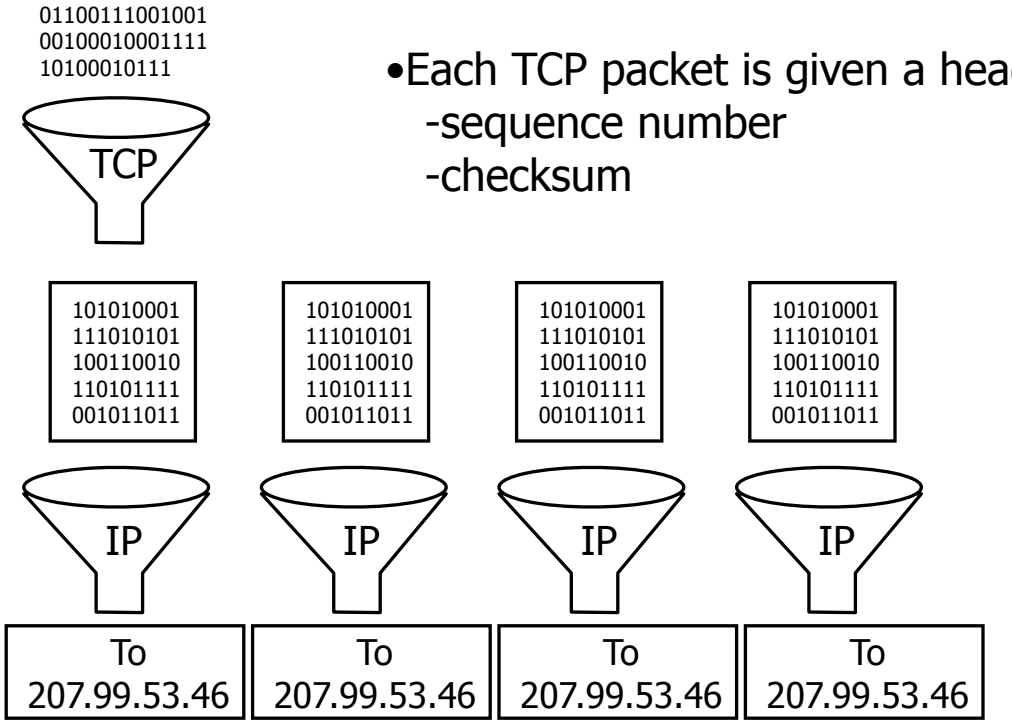
- Intel is little-endian, and Sparc is big-endian
- Many modern chips are bi-endian, but are most commonly used as little-endian.

TCP Connection



Packaging up the data

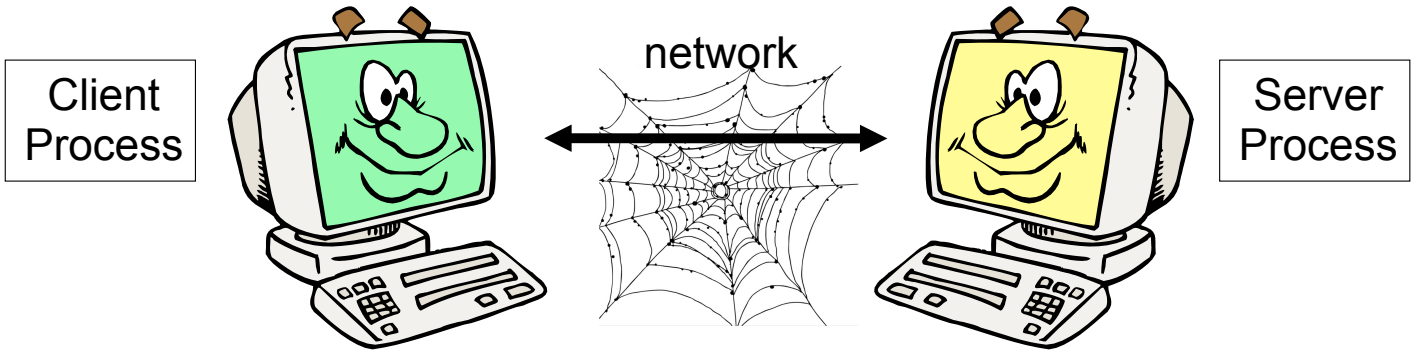
- make packets



- Each TCP packet is given a header
  - -sequence number
  - -checksum

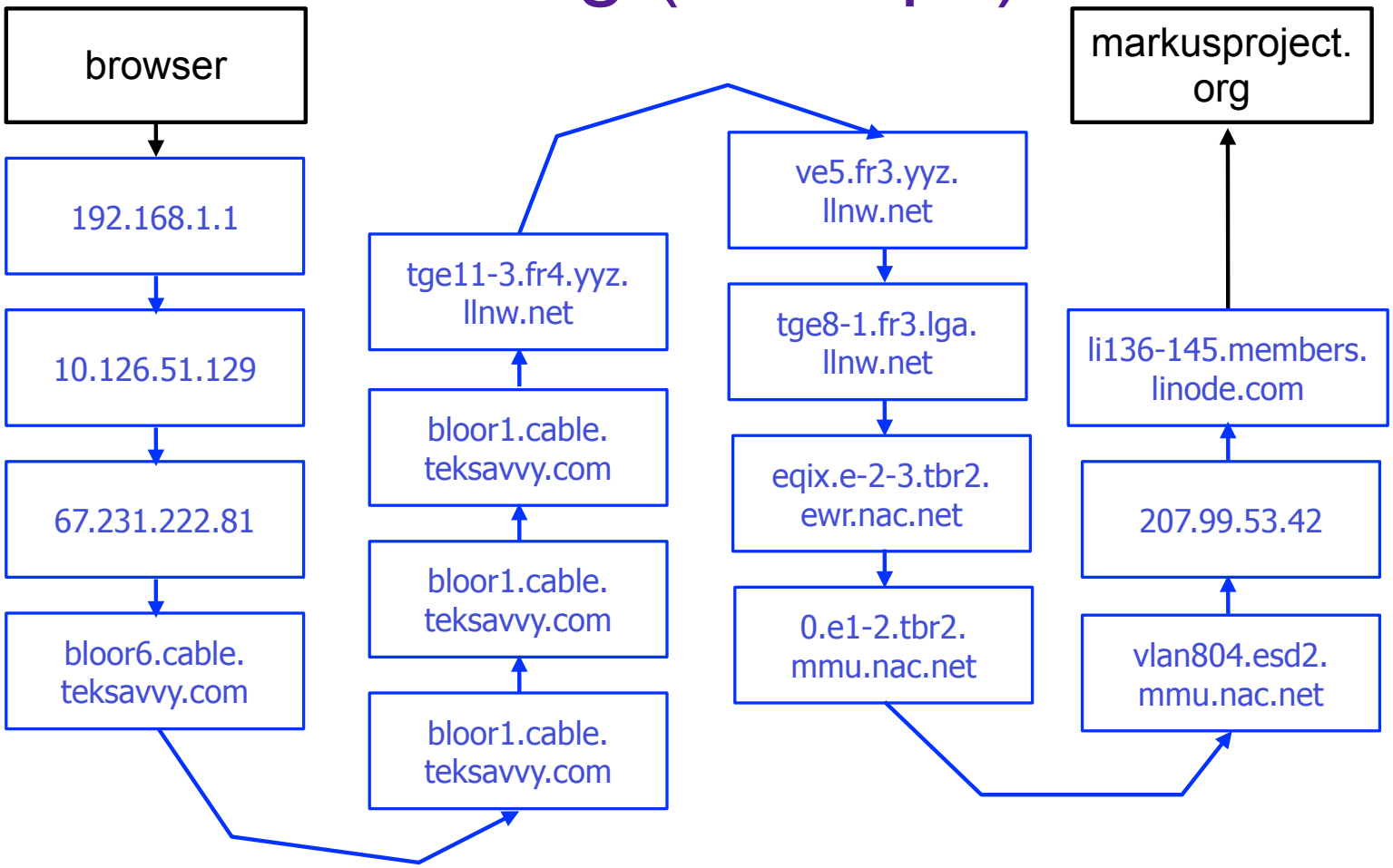
- put in an IP envelope with another header

The Big Picture

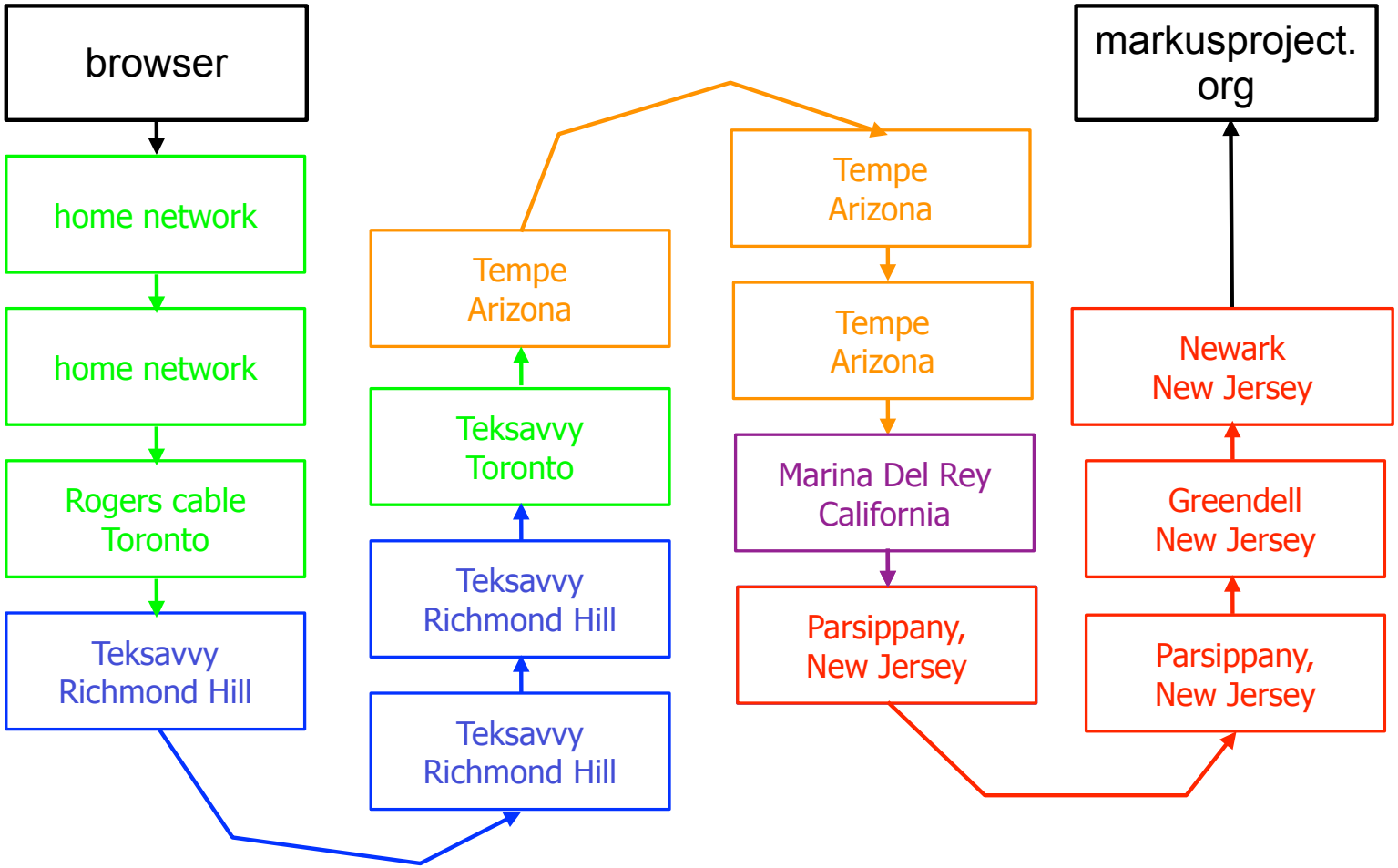


- **Client-Server model**: a client process wants to talk to a server process
- Client must find server - **DNS lookup**
- Client must find process on server - **ports**
- Finally **establish a connection** so two processes can talk

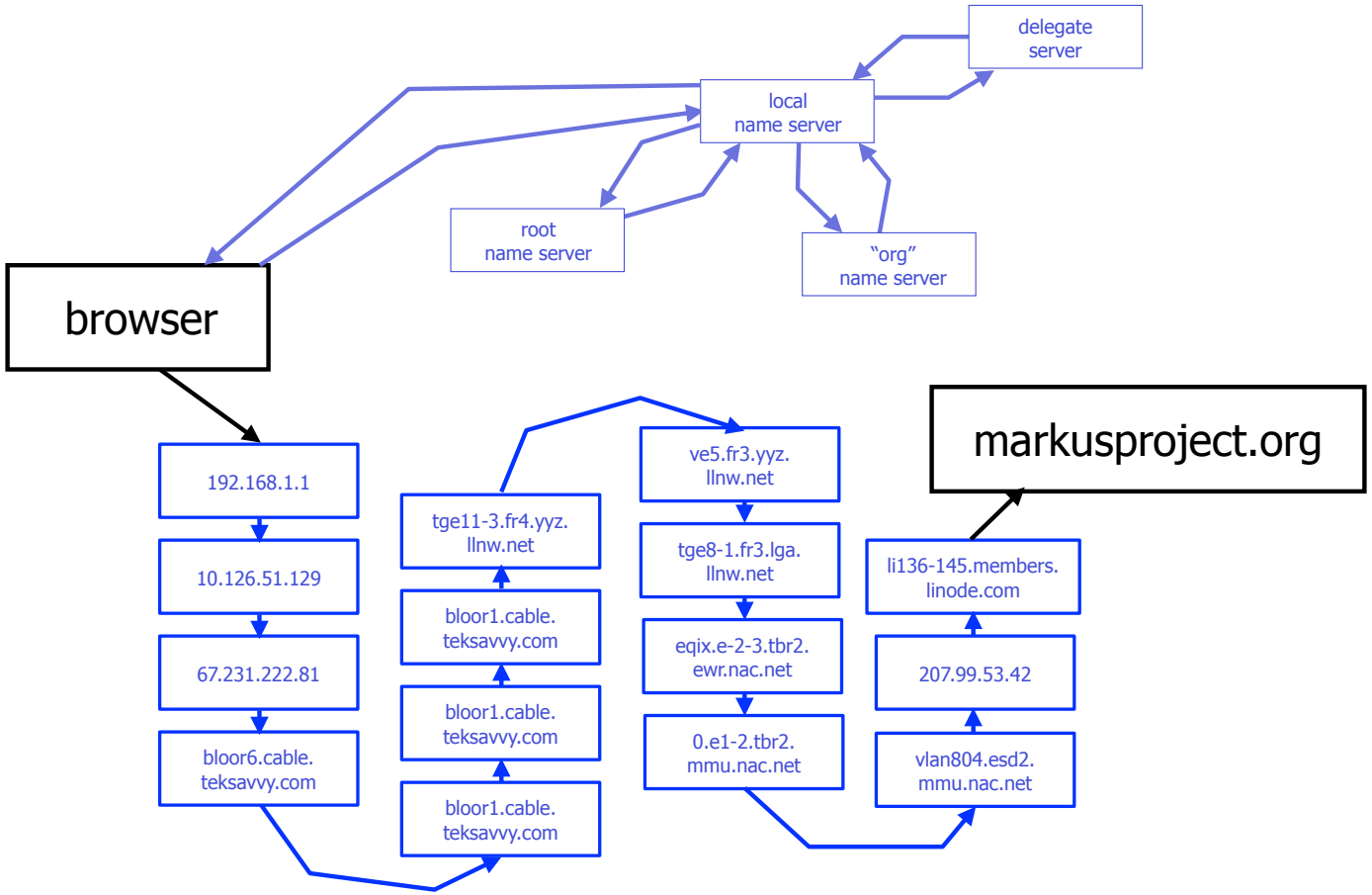
Routing (15 hops)

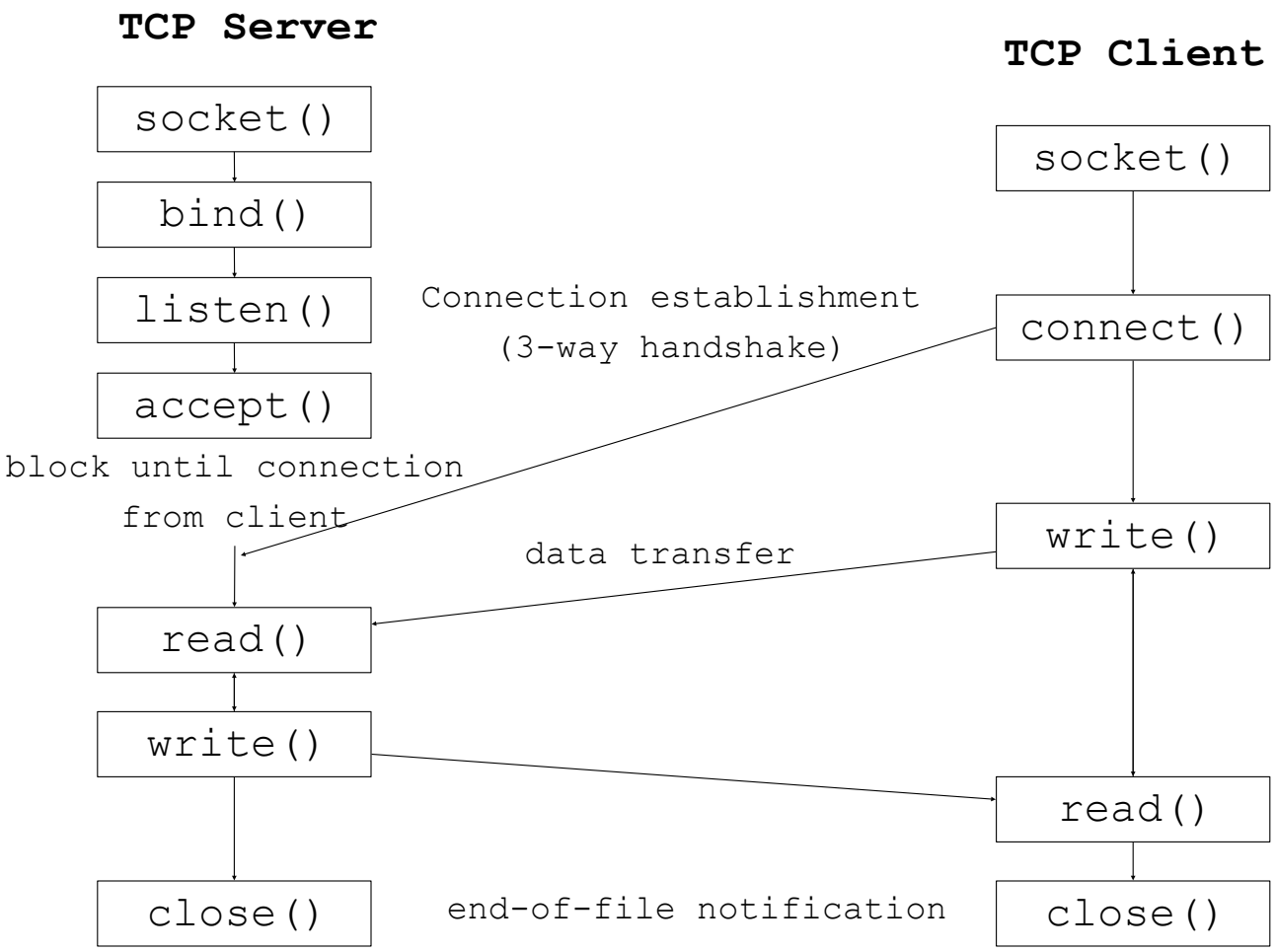


7 cities, 5 states/prov, 2 countries



Putting it together

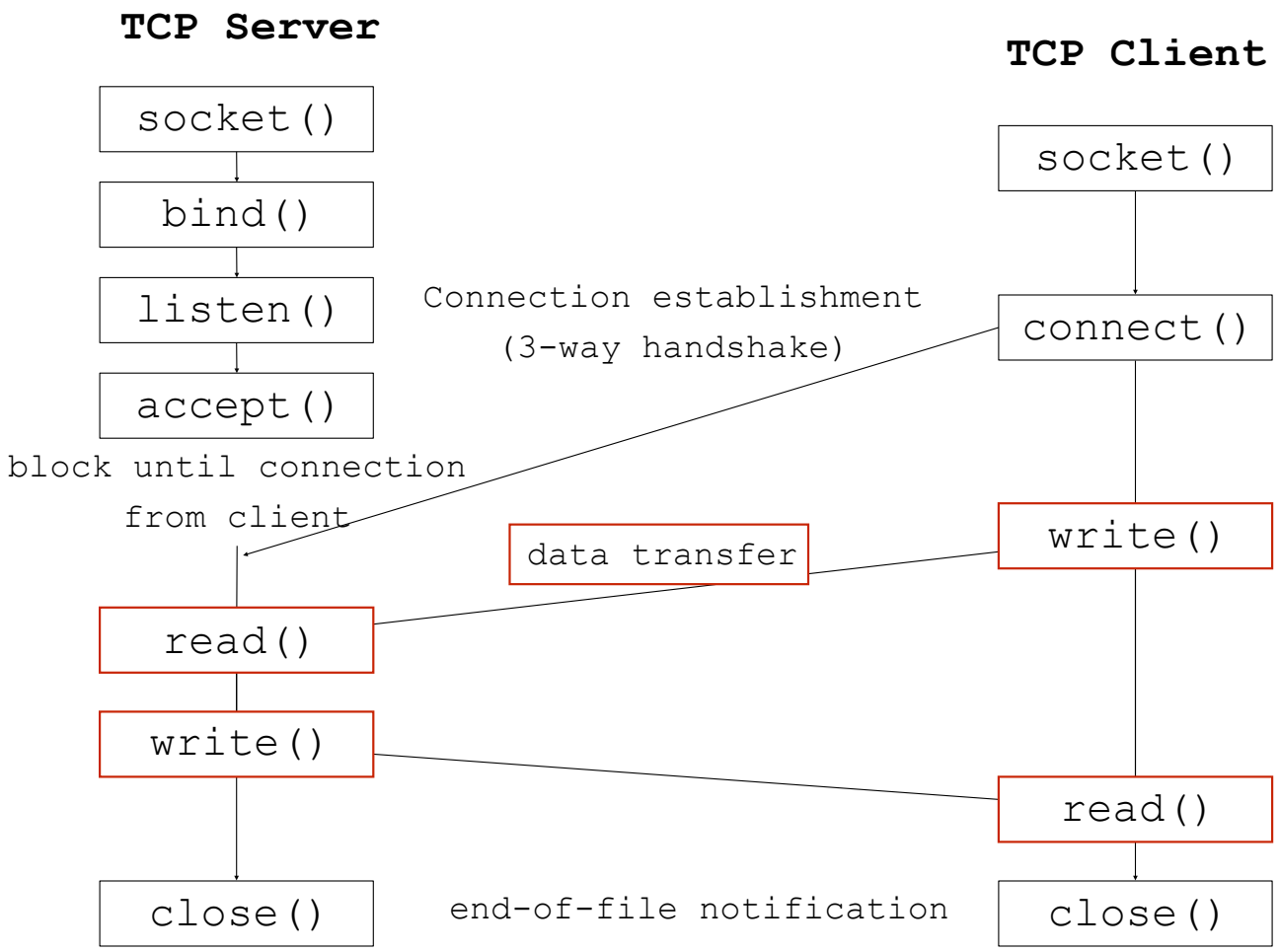



TCP Server
TCP Client
socket()
bind()
listen()
accept()
block until connection
from client
socket()
connect()
write()
read()
write()
close()
Connection establishment
(3-way handshake)
data transfer
end-of-file notification
read()
close()

HTTPS (TLS)
Transport Layer Security

Protocol layer on top of TCP to negotiate encryption protocols and keys

Means all data for TCP connection is encrypted




All we did was click on a link...
https://www.kaspersky.com/blog/amazing-internet-maps/10441/

Take aways

- The web today is made up of complex layers of software
- No one person, organization, or company could have created it in isolation
- We can understand it because we can study one layer at a time
- We can create new things by building on top of existing layers