TCP/IP sockets, HTTP, DNS

(at least we’re going to try...)

10/19/15
What is the difference between a client and a server in TCP networking?

A. Which party opens a connection
B. Which party sends the first data
C. Which party provides a service to the other
D. Which party the programmer or user designates as which
E. Titles are basically meaningless and we should all just get past the need to label everything
What is the difference between a client and a server in TCP networking?

A. Which party opens a connection
B. Which party sends the first data
C. Which party provides a service to the other
D. Which party the programmer or user designates as which
E. Titles are basically meaningless and we should all just get past the need to label everything
What is a networking socket?

A. Representation of a physical network interface (e.g. Ethernet port or wireless card)

B. Representation of a software communication endpoint (e.g. a port and protocol on a machine)

C. OS buffer to store data between receipt and application request

D. Buffer on the networking card managed by hardware rather than the OS

E. Simple server used for testing
What is a networking socket?

A. Representation of a physical network interface (e.g. Ethernet port or wireless card)

B. Representation of a software communication endpoint (e.g. a port and protocol on a machine)

C. OS buffer to store data between receipt and application request

D. Buffer on the networking card managed by hardware rather than the OS

E. Simple server used for testing
What is the type of a socket?

A. Integer
B. Struct defined in a header file
C. Pointer to a struct defined in a header file
D. Can’t really tell from how it’s used
E. None of the above
What is the type of a socket?

A. Integer
B. Struct defined in a header file
C. Pointer to a struct defined in a header file
D. Can’t really tell from how it’s used
E. None of the above
Using TCP

**Client**
- socket
- connect
- send & recv
- close

**Server**
- socket
- bind
- listen
- accept
- recv & send
- close
What should you think when you see capitalized options like AF_INET?

A. They are REALLY IMPORTANT
B. They are types defined in the include files
C. They are named constants
D. They are capitalized for historical reasons
E. Not exactly one of the above
What should you think when you see capitalized options like AF_INET?

A. They are REALLY IMPORTANT
B. They are types defined in the include files
C. They are named constants
D. They are capitalized for historical reasons
E. Not exactly one of the above
   (most are named constants, but DIR is an example of one that was a type)
The system call send takes a pointer to stuff to send and also a length. What is this length?

A. The size of the buffer storing stuff to send
B. The number of chars to send
C. The number of bytes to send
D. One less than the number of chars to send (allowing for \0)
E. One less than the number of bytes to send
The system call send takes a pointer to stuff to send and also a length. What is this length?

A. The size of the buffer storing stuff to send
B. The number of chars to send
C. The number of bytes to send
D. One less than the number of chars to send (allowing for \0)
E. One less than the number of bytes to send
The echo client receives the echoed string back using the following:

```
while(totalBytesRcvd < echoStringLen) {
    numBytes = recv(...);
    totalBytesRcvd += numBytes;
    ...
}
```

Why does this use a loop instead of just one call to recv?

A. In case the response is too large for a single call
B. In case part of the message is lost in transit
C. Because the client should work with any echo server, including one that splits the bytes into multiple groups sent separately
D. Because bytes from a send can be split into multiple recv
E. Not exactly one of the above
The echo client receives the echoed string back using the following:

```c
while(totalBytesRcvd < echoStringLen) {
    numBytes = recv(...);
    totalBytesRcvd += numBytes;
    ...
}
```

Why does this use a loop instead of just one call to recv?

A. In case the response is too large for a single call
B. In case part of the message is lost in transit
C. Because the client should work with any echo server, including one that splits the bytes into multiple groups sent separately
D. Because bytes from a send can be split into multiple recvs
E. Not exactly one of the above
Why does the accept call return a socket rather than just activating the socket passed to it? 
clntSock = accept(servSock, ...);

A. So that the return value can be checked for errors
B. Because servSock could be NULL
C. The returned socket is different than the one passed to it
D. Historical reasons
E. Not exactly one of the above
Why does the accept call return a socket rather than just activating the socket passed to it?
clntSock = accept(servSock, ...);

A. So that the return value can be checked for errors
B. Because servSock could be NULL
C. The returned socket is different than the one passed to it
D. Historical reasons
E. Not exactly one of the above
Why require all the steps to set up a server rather than providing a single call (e.g. getConnection)?

A. So each step can take its own options (more customization)
B. So each step can generate its own errors (more fine-grained errors)
C. So each step can be run in parallel (more performance)
D. Historical reasons
E. To make the interface more complicated
Why require all the steps to set up a server rather than providing a single call (e.g. getConnection)?

A. So each step can take its own options (more customization)
B. So each step can generate its own errors (more fine-grained errors)
C. So each step can be run in parallel (more performance)
D. Historical reasons
E. To make the interface more complicated
Telnet

• Program that opens a TCP connection and lets you type into it:

\texttt{telnet \textit{host} [\textit{optional\_port\_num}]}

either hostname or IP address
Hyper-text transfer protocol (HTTP)
Requesting a webpage

• Open TCP connection on port 80:
  telnet cs.knox.edu 80
• Send the request:
  GET / HTTP/1.1
  Host: cs.knox.edu
  <blank line>
Requesting a webpage

- Open TCP connection on port 80: `telnet cs.knox.edu 80`
- Send the request:
```
GET / HTTP/1.1
Host: cs.knox.edu
```
And now I want interactive web pages and shopping carts...
Domain Name Service (DNS)

Converting hostnames into IP addresses
Getting address: Programmer’s perspective

int getaddrinfo(char* host, char* service,
struct addrinfo* hints, struct addrinfo** results);

- Read about it in Section 3.1 of the text
  - See Section 3.2.1 for utility function to make it easy