Networking

CS 217

Client/Server

- Server: process that provides a service
  - e.g., file server, web server, mail server
  - called a passive participant: waits to be contacted
- Client: process that requests a service
  - e.g., web browser, mail reader
  - called an active participant: initiates communication

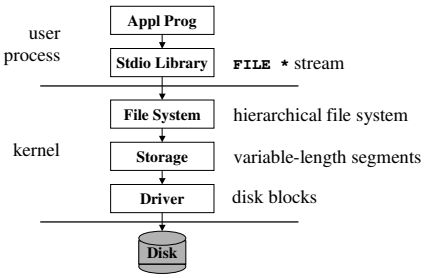
Names and Addresses

- Host name
  - e.g., **`www.cs.princeton.edu`**
- Host address
  - e.g., **`128.112.92.191`**
- Process id
  - indirectly, through a port (mailbox); 0-64k
  - servers assigned a well-known port
  - clients assigned a port "on demand"

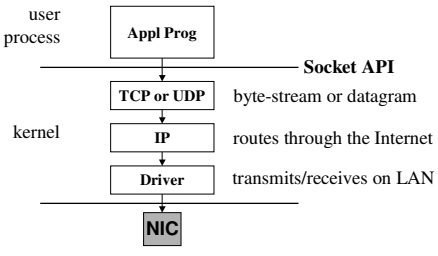
Communication Semantics

- Reliable Byte-Stream (like a pipe): TCP
- Unreliable Datagram: UDP
- Remote Procedure Call (RPC): SunRCP

File System



Network Subsystem



Socket API

- Socket Abstraction
  - end-point of a network connection
  - treated like a file
- Creating a socket

```
int socket(int domain, int type,
           int protocol)
domain = PF_INET, PF_UNIX
type = SOCK_STREAM, SOCK_DGRAM,
       SOCK_RAW
```


Sockets (cont)

- Passive Open (on server)

```
int bind(int socket,
         struct sockaddr *addr,
         int addr_len)
int listen(int socket, int backlog)
int accept(int socket,
           struct sockaddr *addr,
           int addr_len)
```


Sockets (cont)

- Active Open (on client)

```
int connect(int socket,
            struct sockaddr *addr,
            int addr_len)
```

- Sending/Receiving Messages

```
int send(int socket, char *buf,
         int blen, int flags)
int recv(int socket, char *buf,
         int blen, int flags)
```

Trivia Question

- How many messages traverse the Internet when you click on a link?
- DNS: root (2), site (2)
- TCP connection setup: 3
- HTTP request/reply with ACKS: 4
- TCP connection tear-down: 4
- Total: 15