A Client-Server Transaction

Every network application is based on the client-server model:

- A server process and one or more client processes
- Server manages some resource.
- Server provides service by manipulating resource for clients.

Note: clients and servers are processes running on hosts (can be the same or different hosts).

Computer Networks

A network is a hierarchical system of boxes and wires organized by geographical proximity

- LAN (local area network) spans a building or campus.
  - Ethernet is most prominent example.
- WAN (wide-area network) spans country or world.
  - Typically high-speed point-to-point phone lines.

An internetwork (internet) is an interconnected set of networks.

- The Global IP Internet (uppercase “I”) is the most famous example of an internet (lowercase “i”)

Let’s see how we would build an internet from the ground up.
Lowest Level: Ethernet Segment

Ethernet segment consists of a collection of *hosts* connected by wires (twisted pairs) to a *hub*.

Spans room or floor in a building.

![Diagram of an Ethernet segment with a collection of hosts connected to a hub via twisted pairs.]

**Operation**
- Each Ethernet adapter has a unique 48-bit address.
- Hosts send bits to any other host in chunks called *frames*.
- Hub slavishly copies each bit from each port to every other port.
  - Every host sees every bit.

Next Level: Bridged Ethernet Segment

Spans building or campus.

Bridges cleverly learn which hosts are reachable from which ports and then selectively copy frames from port to port.

![Diagram of a bridged Ethernet segment with hosts connected to a hub and a bridge.]

Conceptual View of LANs

For simplicity, hubs, bridges, and wires are often shown as a collection of hosts attached to a single wire:

![Diagram showing a conceptual view of LANs with hosts connected to a single wire.]

Next Level: Internets

Multiple incompatible LANs can be physically connected by specialized computers called *routers*.

The connected networks are called an *internet*.

![Diagram showing an internetwork with LANs connected by routers.]

LAN 1 and LAN 2 might be completely different, totally incompatible LANs (e.g., Ethernet and ATM).
The Notion of an internet Protocol

How is it possible to send bits across incompatible LANs and WANs?

Solution: *protocol software* running on each host and router smoothes out the differences between the different networks.

Implements an *internet protocol* (i.e., set of rules) that governs how hosts and routers should cooperate when they transfer data from network to network.

- TCP/IP is the protocol for the global IP Internet.

What Does an internet Protocol Do?

1. Provides a naming scheme
   - An internet protocol defines a uniform format for *host addresses*.
   - Each host (and router) is assigned at least one of these internet addresses that uniquely identifies it.

2. Provides a delivery mechanism
   - An internet protocol defines a standard transfer unit (*packet*).
     - *Packet* consists of *header* and *payload*.
       - Header: contains info such as packet size, source and destination addresses.
       - Payload: contains data bits sent from source host.

Transferring Data Over an internet protocol

Other Issues

We are glossing over a number of important questions:

- What if different networks have different maximum frame sizes? (segmentation)
- How do routers know where to forward frames?
- How are routers informed when the network topology changes?
- What if packets get lost?

These (and other) questions are addressed by the area of systems known as *computer networking*. 
Global IP Internet

Most famous example of an internet.

Based on the TCP/IP protocol family

- **IP** (Internet protocol):
  - Provides basic naming scheme and unreliable delivery capability of packets (datagrams) from host-to-host.

- **UDP** (Unreliable Datagram Protocol)
  - Uses IP to provide unreliable datagram delivery from process-to-process.

- **TCP** (Transmission Control Protocol)
  - Uses IP to provide reliable byte streams from process-to-process over connections.

Accessed via a mix of Unix file I/O and functions from the sockets interface.

Basic Internet Components

An *Internet backbone* is a collection of routers (nationwide or worldwide) connected by high-speed point-to-point networks.

A **Network Access Point (NAP)** is a router that connects multiple backbones (sometimes referred to as *peers*).

**Regional networks** are smaller backbones that cover smaller geographical areas (e.g., cities or states).

A **point of presence (POP)** is a machine that is connected to the Internet.

**Internet Service Providers** (ISPs) provide dial-up or direct access to POPs.

The Internet Circa 1993

In 1993, the Internet consisted of one backbone (NSFNET) that connected 13 sites via 45 Mbs T3 links.

- **Merit (Univ of Mich)**, **NCSA (Illinois)**, **Cornell Theory Center**, **Pittsburgh Supercomputing Center**, **San Diego Supercomputing Center**, **John von Neumann Center (Princeton)**, **BARRNet (Palo Alto)**, **MidNet (Lincoln, NE)**, **WestNet (Salt Lake City)**, **NorthwestNet (Seattle)**, **SESQUINET (Rice)**, **SURANET (Georgia Tech)**.

Connecting to the Internet involved connecting one of your routers to a router at a backbone site, or to a regional network that was already connected to the backbone.
In the early 90’s commercial outfits were building their own high-speed backbones, connecting to NSFNET, and selling access to their POPs to companies, ISPs, and individuals.

In 1995, NSF decommissioned NSFNET, and fostered creation of a collection of NAPs to connect the commercial backbones.

Currently in the US there are about 50 commercial backbones connected by ~12 NAPs (peering points).

Similar architecture worldwide connects national networks to the Internet.

---

**Network Access Points (NAPs)**

Private “peering” agreements between two backbone companies often bypass NAP

---

**Source:** Boardwatch.com

Note: Peers in this context are commercial backbones...
MCI/WorldCom/UUNET Global Backbone

A Programmer’s View of the Internet

1. Hosts are mapped to a set of 32-bit IP addresses.
   - 128.2.203.179

2. The set of IP addresses is mapped to a set of identifiers called Internet domain names.
   - 128.2.203.179 is mapped to www.cs.cmu.edu

3. A process on one Internet host can communicate with a process on another Internet host over a connection.

1. IP Addresses

32-bit IP addresses are stored in an IP address struct
- IP addresses are always stored in memory in network byte order (big-endian byte order)
- True in general for any integer transferred in a packet header from one machine to another.
  - E.g., the port number used to identify an Internet connection.

```c
/* Internet address structure */
struct in_addr {
    unsigned int s_addr; /* network byte order (big-endian) */
};
```

Handy network byte-order conversion functions:
- htonl: convert long int from host to network byte order.
- htons: convert short int from host to network byte order.
- ntohl: convert long int from network to host byte order.
- ntohs: convert short int from network to host byte order.

Dotted Decimal Notation

By convention, each byte in a 32-bit IP address is represented by its decimal value and separated by a period
- IP address 0x8002C2F2 = 128.2.194.242

Functions for converting between binary IP addresses and dotted decimal strings:
- inet_aton: converts a dotted decimal string to an IP address in network byte order.
- inet_ntoa: converts an IP address in network byte order to its corresponding dotted decimal string.
- “n” denotes network representation. “a” denotes application representation.
2. Internet Domain Names

Domain Naming System (DNS)

The Internet maintains a mapping between IP addresses and domain names in a huge worldwide distributed database called DNS.

- Conceptually, programmers can view the DNS database as a collection of millions of host entry structures:

```c
/* DNS host entry structure */
struct hostent {
    char *h_name; /* official domain name of host */
    char **h_aliases; /* null-terminated array of domain names */
    int h_addrtype; /* host address type (AF_INET) */
    int h_length; /* length of an address, in bytes */
    char **h_addr_list; /* null-terminated array of in_addr structs */
};
```

Functions for retrieving host entries from DNS:

- `gethostbyname`: query key is a DNS domain name.
- `gethostbyaddr`: query key is an IP address.

A Program That Queries DNS

```c
int main(int argc, char **argv) { /* argv[1] is a domain name */
    char **pp; /* or dotted decimal IP addr */
    struct in_addr addr;
    struct hostent *hostp;

    if (inet_aton(argv[1], &addr) != 0)
        hostp = gethostbyaddr((const char *) &addr, sizeof(addr), AF_INET);
    else
        hostp = gethostbyname(argv[1]);

    printf("official hostname: \%s\n", hostp->h_name);

    for (pp = hostp->h_aliases; *pp != NULL; pp++)
        printf("alias: \%s\n", *pp);

    for (pp = hostp->h_addr_list; *pp != NULL; pp++) {
        addr.s_addr = *((unsigned int *) pp);
        printf("address: \%s", inet_ntoa(addr));
    }
}
```
Querying DNS from the Command Line

Domain Information Groper (dig) provides a scriptable command line interface to DNS.

```
linux> dig +short kittyhawk.cmcl.cs.cmu.edu
128.2.194.242
linux> dig +short -x 128.2.194.242 KITTYHAWK.CMCL.CS.CMU.EDU.
linux> dig +short aol.com
205.188.145.215
205.188.160.121
64.12.149.24
64.12.187.25
linux> dig +short -x 64.12.187.25 aol-v5.websys.aol.com.
```

3. Internet Connections

Clients and servers communicate by sending streams of bytes over connections:
- Point-to-point, full-duplex (2-way communication), and reliable.

A **socket** is an endpoint of a connection
- Socket address is an IPAddress:port pair

A **port** is a 16-bit integer that identifies a process:
- **Ephemeral port**: Assigned automatically on client when client makes a connection request
- **Well-known port**: Associated with some service provided by a server (e.g., port 80 is associated with Web servers)

A connection is uniquely identified by the socket addresses of its endpoints (**socket pair**)
- (cliaddr:cliport, servaddr:servport)

Putting it all Together: Anatomy of an Internet Connection

Client socket address
128.2.194.242:51213

Server socket address
208.216.181.15:80

Client host address
128.2.194.242

Server host address
208.216.181.15

Next Time

How to use the sockets interface to establish Internet connections between clients and servers

How to use Unix I/O to copy data from one host to another over an Internet connection.