Lecture 1
Plan: Introduction/finish 1.1-1.3

1. Syllabus

- Goal: Learn principles of computer network technologies (**L3 and BigPipe Backbone**), algorithms used in computer networks, and network programming (don't confuse network programming with web programming)
- Big picture of this course: Starting from a network communication model (layered model), we go through each layer and study their functionalities and design issues. Focus on the network programming in the context of Internet. There are lots of terminologies/acronyms in Computer Networks. (**S1**, **acronyms**)
- Textbook required and we will use UNIX C/C++ programming environment. (ask about UNIX experience)
- Outline: First 7 chapters and go through the schedule in the end of the syllabus
- Office hours: Ask students if the times are good.
- Homework: Late work policy.
- Exams: 1 quiz, 2 exams and 1 final. The second part of the course is focused more on network programming. Allow you to bring a note.
- Web: Use ANGEL and will assist you if you need assistance.
- Honor code: No cheating on homework and exams. Must sign or state in the emails.

2. Introduction to computer networks
Computer networks combine the computational power of computers with the communication capabilities of communication networks.

- Def: A collection of autonomous computers interconnected by a single technology.
  This definition excludes
    Shared memory multiprocessors
    A central computer with time-sharing terminals

Q:/ What's the difference between a distributed system and a computer network
A:/Contrast this with **a "distributed system" which is a software system built upon a network**, presenting a "single machine" view to the user; that is, a transparent view of the network. e.g. WWW web
In a network, users log on to a local machine, then run applications which can communicate with applications running on other machines in the network. So no uniform interface is necessary.

This is still a wide-ranging definition, including Local Area Networks (LANs) like the one in the CS labs and Wide Area Networks (WANs).

- What do we use networks for?
  - resource sharing -- sharing hardware (e.g., printers), sharing files (i.e., information)
  - communication -- e-mail, instant messaging, videoconferencing
  - e-commerce-- business-to-business, business-to-consumer, consumer-to-consumer
  - entertainment – VOD, gambling, gaming

So, what aspects of networks will we be studying in this course?
  - Hardware

- Software
- Applications

a. Network Hardware

Q:/ What hardware technologies are used for computers to communicate with each other?
A:/ More of an engineering question about data communication

- Communication Media: Copper based (twisted pair, coaxial cable), Optical fiber, Wireless (satellite, microwave) **S2**
- Communication technologies: Broadcast (Ethernet), Point to point
- Topology: Star, ring, bus, hierarchical
- Scale: LAN, MAN, WAN

Differences in distance lead to differences in which technologies are most effective. The result is that many different technologies are used for networking. In certain situations, different technologies work well. None is best for all situations. → which makes this course a little bit difficult because no single underlying principle exist.

- An internet is a collection of interconnected, autonomous networks. For example, the figure below illustrates a collection of LANs connected by a WAN.

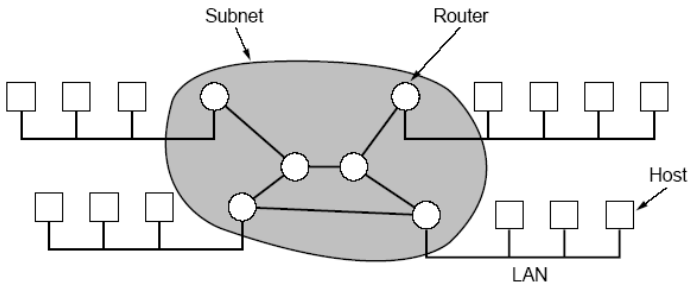


Fig. 1-9. Relation between hosts on LANs and the subnet.

The global Internet is an example of an internet. It encompasses all sorts of physical communication media. What ties it together is a set of standardized *protocols* agreed upon by everybody.
Protocols are rules for communication. A protocol is an agreed-upon sequence of messages. Like password, countersign. The protocol also includes the definition of the format of messages.
Network protocols may be implemented in hardware or software.

b. Network Software

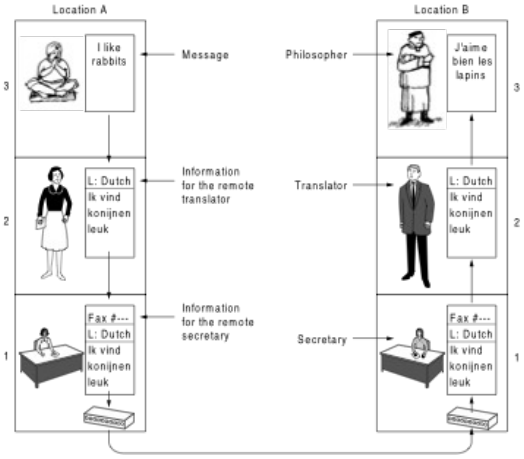
Q:/Why do we need network software
A:/ The ultimate goal is to write applications over the computer networks.
Note: It's better to use the abstract view of the network hardware so the software can be freed of the low level details.

- Basic problems that have been faced by network designers as networks evolved:

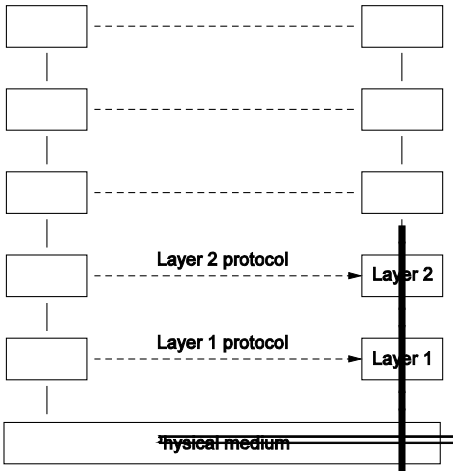
i.It's a long way from low-level hardware to the application view

ii. Need to standardize the application view, while hardware level involves different technologies, different vendors, etc.

Solution: Network software and hardware is organized in layers (protocol stack, **S3-4**)
example: philosopher example



- Communication via abstraction (protocol stack)



When a peer on host1 wants to talk to its peer on host2 at level 5, it follows the level 5 protocol. But what it really does is to make calls to the level 4 service routines on its own machine.
The level 4 software then follows the level protocol to transfer information to the level 4 peer on host2. But it does this not by talking to the level 4 peer directly, but by making calls to level 3 on its own machine, etc.
Eventually, communication is made through whatever physical medium is provided. The software on host2 gets the transmitted data and passes it up through the higher levels.
When layer 4 on host2 received data from layer 3 (on host2) it will appear to come from its layer 4 peer.

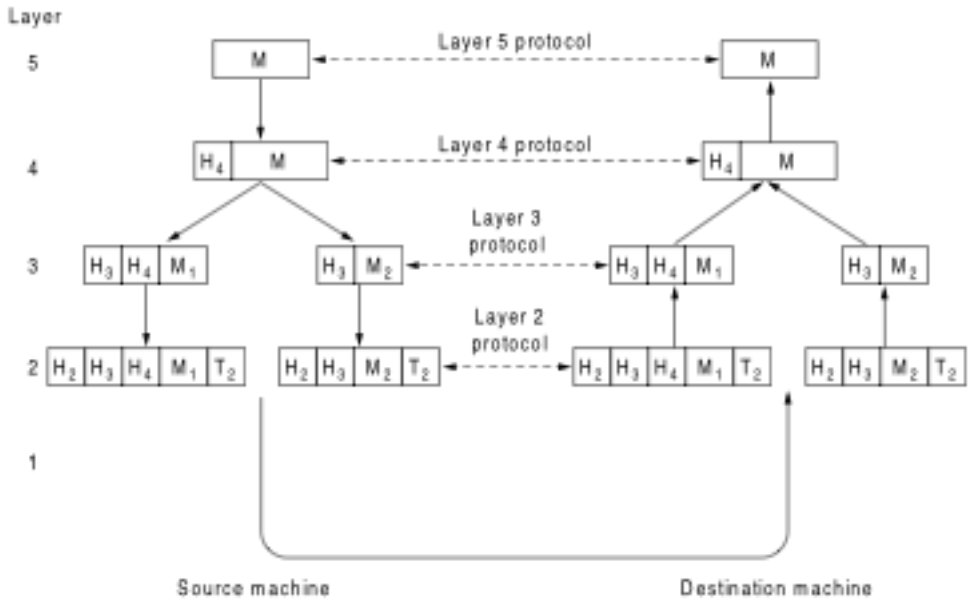
Campus mail system example

Many college campuses have their own mail delivery system, to send correspondence between departments.

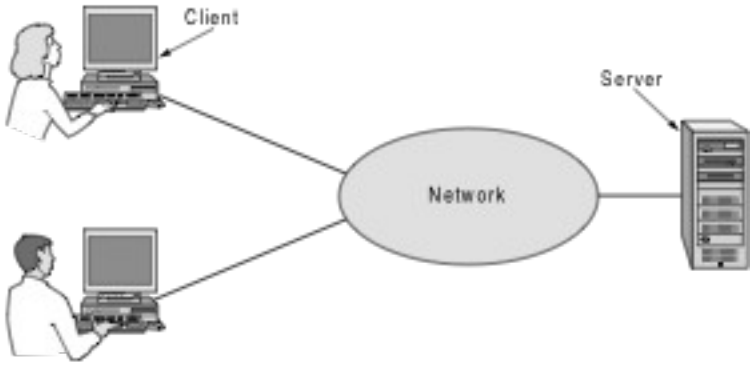
Suppose that Ashland University decides to open a branch campus in say, Las Vegas. Some faculty have offices there, some here in Ohio.
I want to send a copy of my latest paper to Professor X in room 200 in the Newton Bldg in Las Vegas. So I put the paper in an envelope with address, and drop it in a campus mailbox.
When it is picked up and taken to the campus P.O., the clerk sorts mail into local, L.V., US Mail.

- local mail --> direct delivery
- US mail --> deliver to US P.O.
- L.V. mail --> what to do?
  - Put all LV mail in a box and send it through FedEx to LV
  - Put in an envelope, affix stamp, send to LV PO.
  - Open it and fax it to LV PO, which prints and delivers it.

In general, at each layer, header and footer data is added to the actual communication. This is analogous to the extra envelope in the LV example. Level i receives message M from level i+1. It does not care about the contents, it is just a string of bits. It packages it in the format required by level i-1 and passes it down.



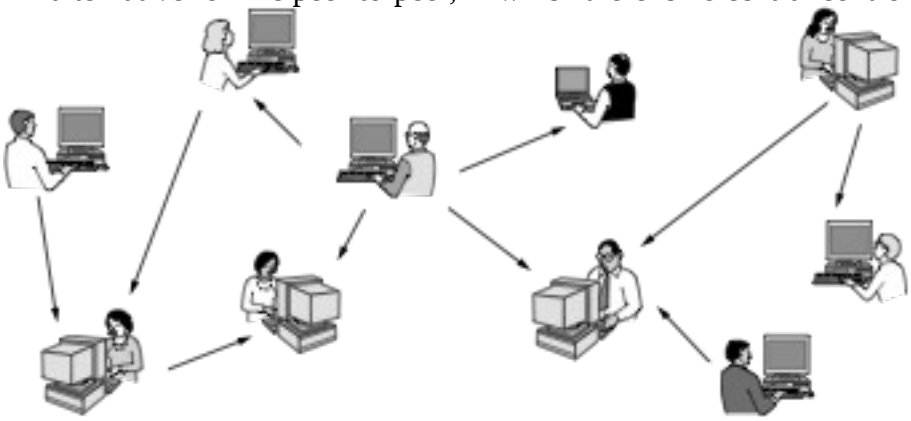
One other comment: Current network applications are mostly based on the "client-server" model of communication:



Each client-server application has its own protocol, consisting of a set of request messages that a client can send to a server, and the corresponding replies.

example http (hypertext transfer protocol) used by the Web.

An alternative form is peer-to-peer, in which there is no central control:



So, what sort of services are provided by these prototcol layers?

1. Naming/addressing. Identifying senders and receivers.
2. Routing. Determining a path from sender to receiver.
3. Error control. Detection/correction of errors.
4. Flow control. Handle communication between fast and slow devices.
5. Multiplexing. Using some physical channel for multiple communications simultaneously.
6. Sequence control. Preserve message ordering.
7. Encryption.

Okay, we have seen the idea of a layered architecture and protocol stacks. What sort of layering is done in practice?

There are two models:

- The OSI Reference Model -- gives a good theoretical picture of what needs to be done.
- TCP/IP -- became the de facto Internet standard. evolved on Internet from ARPANET.