Overview of Network Software

CS158a

Chris Pollett

Jan 31, 2007.

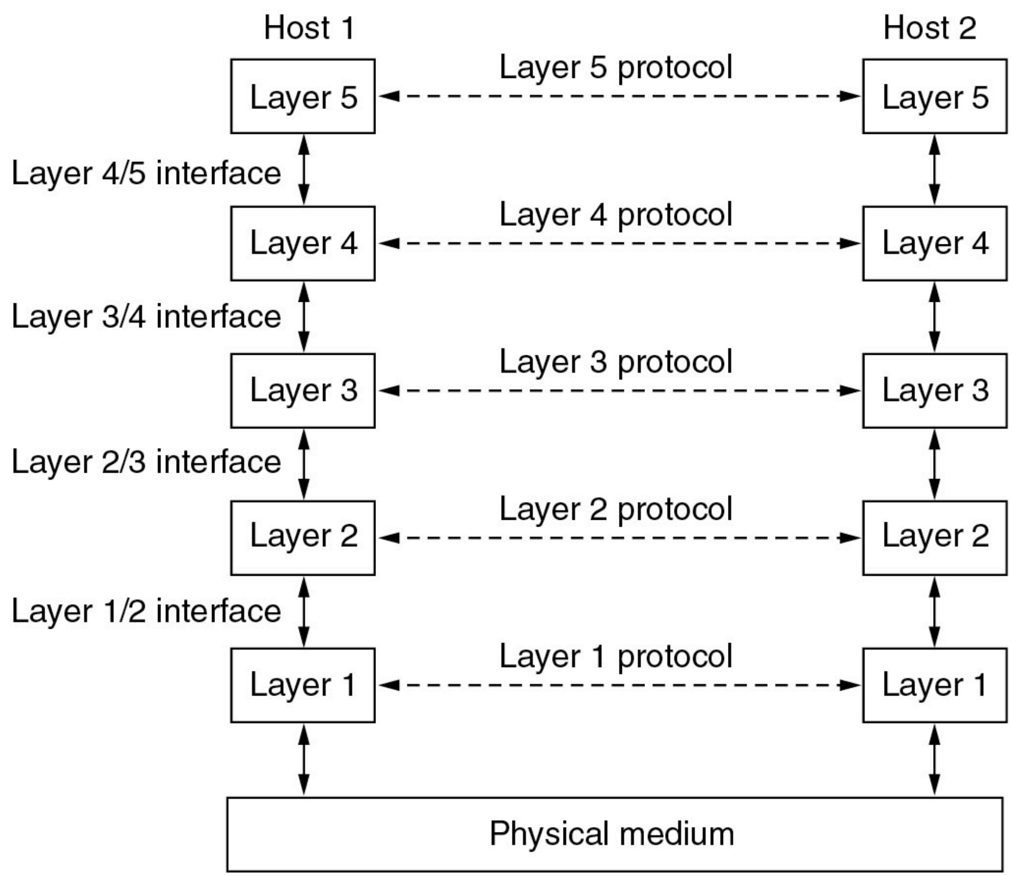
Outline

- Design Issues for Protocol Hierarchies
- Reference Models
- Example Networks

Protocol Hierarchies-Review

- To reduce design complexity this software is typically arranged into **layer**s or **levels**.
- Each levels is built on top of the layer below and offers services to the layer above.
- The rules and conventions **(interface)** for a given layer on one machine (a **peer)** to communicate with the same layer on another machine are called **protocols**.
- Below layer 1 is the physical medium through which communication occurs

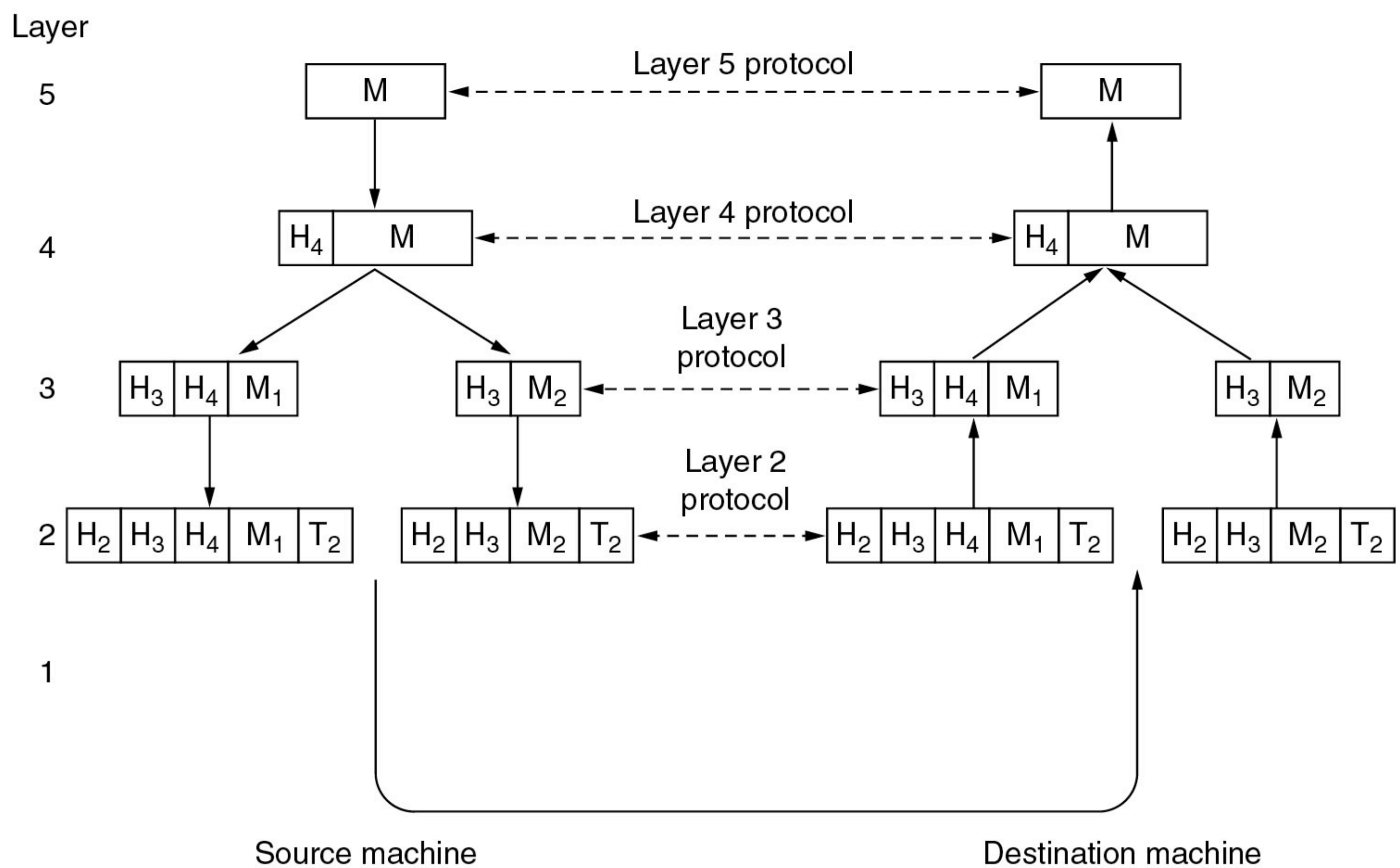
Example of Protocol Hierarchy



More on Protocol Hierarchies

- A set of layers and protocols is called a **network architecture**
- A list of protocols used by a certain system, one protocol per layer, is called a **protocol stack**.
- When the application layer, say layer 5, wants to send a message, it typically passes it to the layer below, layer 4. Layer 4 typically adds to the message some identifying information that it useful to transmit the message at the Layer 4 levels, say addressing info, then passes the message to layer 3 and so on.
- This info added by layer 4 is called a **header.**

Example of Adding Headers as we go to lower layers



Design Issues for Layers

- Each layer needs a mechanism for identifying the sender and the recipient. i.e., some form **addressing**. This identifying information might include the process on the recipient machine which should handle the message.
- **Error Control** -- need a mechanism to both detect and recover from errors.
- **Flow control** -- need a mechanism to handle mismatches between the rate the sender can send and the rate the receiver can receive.
- If several processes on two machines are communicating, it might make sense two parcel their messages together. This is called **multiplexing**. Extracting the separate messages at the other end is called **demultiplexing**.
- If there are multiple paths between the source and destination, a route must be chosen. The process of determining a route is called **routing**.

Types of Service a Layer can Provide

- **Connection-Oriented Service** -- (like a telephone call), the user establishes a connections, uses the connection (could send several messages), and releases the connection. Two flavor: message stream with fixed message sizes and byte stream. Examples: file transfer
- **Connectionless Service (datagram service)**-- (like mailing a letter) Each message has the full address and is sent independent of each other. Examples: DNS lookup, email, etc.
- **Quality of Service** -- some services promise never to lose data. This is usually implemented by having the receiver acknowledge receipt of messages.
  - Both connection-oriented and connectionless services can be **acknowledged services.**
- **Request-reply service** -- a connectionless service in which a single datagram request is made and the reply contains the answers. For example, network time.

Service Primitives

- A service is formally specified by giving a set of primitive operations (**primitives**) it supplies.
- For example, if a layer provides a connection-oriented service, it might provide the primitives:

Primitive	Meaning
LISTEN	Block waiting for an incoming connection
CONNECT	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Send a message to the peer
DISCONNECT	Terminate a connection

OSI Reference Model

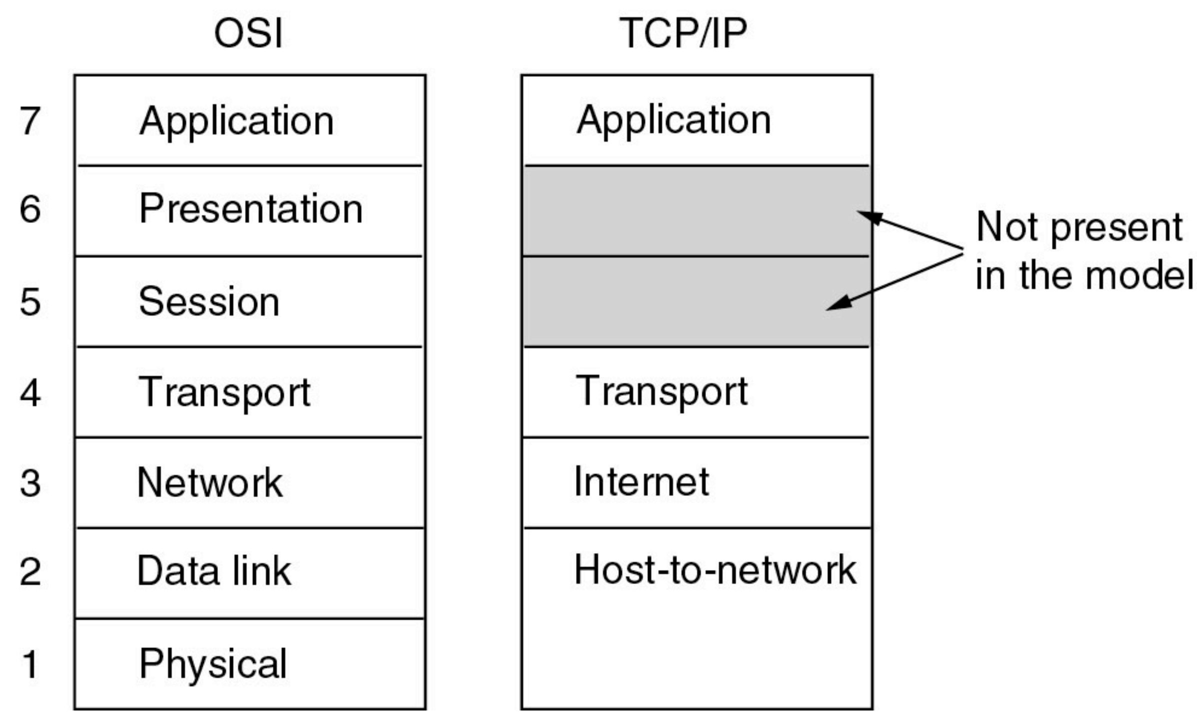
- We now begin to look at some real-world network architectures, starting with OSI
- Open Systems Interconnection (OSI), is an Internation Standards Organization (ISO), specifying how to connect computer systems which are open/available for communicating.
- It has seven layers.

Layers of the OSI Model

- Layer 1 (**Physical Layer**) -- concerned with transmitting 0s and 1s over the network. Design issues deal with mechanical, electrical, and timing interfaces as well as physical transmission medium.
- Layer 2 **(Data Link)** -- transforms a raw transmission facility into a line that appears free from undetected transmission errors and makes this available to the network layer. It breaks up the input data into **data frames**. If the service is reliable, the receiver acknowledges correct receipt of each frame with an **acknowledgment frame.**
- Layer 3 (**Network Layer**) -- controls operation of the subnet. It needs to be able to route messages from the source to destination. Routes can be either static or dynamic.
- Layer 4 (**Transport Layer**) -- receives data, split it into smaller units for the network layer. It adds sequencing information so that the original data can be reassembled at the other end.
- Layer 5 (**Session Layer**) -- allows users on different machines to establish sessions between them. This means it has a facility for saying who should transmit next. **(dialog control).**
- Layer 6 (**Presentation Layer**) -- used to specify the syntax of the data to be transmitted. You can think of this data as like specifying an XML language.
- Layer 7 **(Application Layer)** -- contains a variety of protocols which are needed by the end user. For example, FTP, HTTP, POP mail, etc

TCP/IP Reference Model

- Stands for Transmission Control Protocol/ Internet Protocol
- Developed out of one of the first networks around called ARPANET.
- Its layers correspond in the following way to the OSI model:



Hybrid Model from Book

5	Application layer
4	Transport layer
3	Network layer
2	Data link layer
1	Physical layer

Example Networks- ARPANET

- First set up in Dec, 1969.
- ARPA stands for Advanced Research Projects Agency.
- Subnet consisted on minicomputers called IMPs (interface message processors).
- A node in the network was in IMP and a host .
- Host could send IMPs up to 8063 bits, which would be split into packets of 1008 bits and sent to a target IMP.
- ARPANET quickly grew and experiments resulting from it led to the development of TCP/IP in 1974.
- The Domain Name System (DNS) was added to give human understandable names for machines in the 1980s.

NSFNET

- As to belong to ARPANET, a university needed to have a defense contract not every university and company could join.
- So NSFNET was created,
- The backbone of NSFNET were six supercomputer centers across the US. Regional networks were connected off of these.
- Initial speeds were 56kbps
- NSFNET was very popular and quickly grew becoming ANS so that for-profit companies could use the networks.
- Speeds increased around 1990 to 1.5 Mbps and then 45 Mbps.
- ANS was sold to AOL in 1995.
- By then commercial IP services were sprouting up and the government transition out of the networking business.
- To do this transition four regional NAPs (network access points) were created in SF, Chicago, Washington, and New Jersey
- Any operator that wanted to provide backbone service to the regional network had to connect to all the NAPs. This guaranteed interoperability.